Unprovabiliy of Continuation-Passing Style Transformation in Lambek Calculus*

Masaya Taniguchi[1,2]

[1] Japan Society for the Promotion of Science
[2] Japan Advanced Institute of Science and Technology
taniguchi@jaist.ac.jp

Abstract. Combinatory categorial grammar (CCG) has been able to accommodate various linguistic phenomena with added combinators to categorial grammar (CG). In particular, the type-raising rule realized by such a combinator in CCG and has solved the scope change of quantifiers, and the rule is generalized as continuation-passing style (CPS) transformation. However, there is concern that CPS may exceedingly accept ungrammatical sentences. In this paper, we analyze the expanded grammar rules using the Lambek calculus, that is a formal system of CG, to restrict CPS transformation. First, we show that Barker's CPS transformation is provable and Plotkin's CPS transformation is unprovable in Lambek calculus. Second, we show that a subset of Plotkin's CPS transformations which is provable in Lambek calculus. Due to the complexity of proving unprovability, we formalize the proof in Isabelle/HOL and verify it. We regard this subset represents the grammatical class in terms of Lambek calculus, and call it type-restricted CPS transformation.

Keywords: CPS transformation · Lambek Calculus · Isabelle/HOL

1 Introduction

The *ad-hoc* introduction of grammar rules was happened to deal with linguistic phenomena in categorial grammar (CG). Such haphazard introduction potentially gets over the original grammar class in Chomsky hierarchy, that is, the grammar rules may over-generate ungrammatical sentences. For example, the subordinate clause "that cat walks" is scrambled as "cat that walks" by the cross-composition rule [8].

$$\dfrac{\dfrac{\mathit{that} : \mathrm{SBAR/S} \qquad \dfrac{\mathit{cat} : \mathrm{NP} \qquad \mathit{walks} : \mathrm{NP\backslash S}}{\mathit{cat}\,\mathit{walks} : \mathrm{S}}<}{\mathit{that}\,\mathit{cat}\,\mathit{walks} : \mathrm{SBAR}}>$$

$$\dfrac{\mathit{cat} : \mathrm{NP} \qquad \dfrac{\mathit{that} : \mathrm{SBAR/S} \qquad \mathit{walks} : \mathrm{NP\backslash S}}{\mathit{that}\,\mathit{walks} : \mathrm{NP\backslash SBAR}}>Bx}{\mathit{cat}\,\mathit{that}\,\mathit{walks} : \mathrm{SBAR}}<$$

* This work was supported by Grant-in-Aid for JSPS Fellows Number 21J15207.

Hence, we verify the generative power using the formal method. Our targets are CPS transformation rules investigated by Plotkin [7] and Barker [1]. Note that there are slightly differences for generative power between them. The aim of this paper is to show that Plotkin's CPS transformation rule is unprovable in Lambek calculus [5], which is the mathematical formalization of categorial grammar. Further, we show the over-generation of this grammar rules from the unprovability.

There are two motivations for CPS transformation rule. One is to modify the scope of quantifiers, and another is to extend grammar rules for a specific linguistic phenomenon. Generally, the first motivation is not problematic because it does not violate the original syntax theory. However, the second motivation is disputable as to whether it solves only targeted phenomenon. For example, both of the type-raising rule and a cross-composition rule are well-known grammar rules in CG, however, actually, the former is provable in Lambek calculus while the latter is not. The following proof is a usage of the CPS transformation rule. In this paper, we regard that only those sentences provable by the calculus are grammatical.

$$\dfrac{\dfrac{cat : \mathrm{NP}}{cat : \mathrm{S}/(\mathrm{NP}\backslash\mathrm{S})}\,\mathrm{CPS} \qquad walks : \mathrm{NP}\backslash\mathrm{S}}{cat\, walks : \mathrm{S}}\,>$$

Section 2 introduces the basic concepts of continuations in computer science and Lambek calculus. Section 3 shows that the CPS transformation is unprovable in Lambek Calculus and the rule alternative to CPS transformation. Section 4 shows that the formalization of Lambek calculus in the proof assistant system Isabelle/HOL and give the formal proof of all theorems in the present paper.

2 Preliminaries

2.1 Continuations in Lambda Calculus

First, we work with untyped and simply-typed lambda calculus, as defined in [3]. A usual computer program implicitly executes a code sequentially step by step, however, in some cases, the order of execution may change. The remained schedule after the preempted execution is called a *continuation*. Then, a computer program with an explicitly-mentioned continuation is said to be in continuation-passing style (CPS).

Definition 1 (Plotkin's CPS transformation [7]). $[\![\cdot]\!]$ *is recursively defined as a map from a DS lambda term to a CPS lambda term, where* x *is a variable, and* M *and* N *are meta variables. Further,* m, n, k *represent the continuations of each term.*

$$[\![x]\!] \equiv \lambda k.kx \quad [\![\lambda x.M]\!] \equiv \lambda k.k(\lambda x.[\![M]\!]) \quad [\![MN]\!] \equiv \lambda k.[\![M]\!](\lambda m.[\![N]\!](\lambda n.mnk))$$

The original CPS transformation in Definition 1 is defined in untyped-lambda calculus. Here, we consider the type of the transformation in simply-typed lambda calculus as follows.

Definition 2 (Type of Plotkin's CPS transformation [2]). *$\langle\!\langle\cdot\rangle\!\rangle$ is mutually defined with $\langle\cdot\rangle$, where capital letters are types of simply-typed lambda calculus. Further, A is the answer type uniquely determined in the global context. $\langle\!\langle\cdot\rangle\!\rangle$ is corresponding to $[\![\cdot]\!]$ in Definition 1.*

$$\langle\!\langle X\rangle\!\rangle_A = (\langle X\rangle_A \to A) \to A \quad \langle X \to Y\rangle_A = \langle X\rangle_A \to \langle\!\langle Y\rangle\!\rangle_A \quad \langle Z\rangle_A = Z \,(Z \text{ is atomic})$$

In combinatory categorial grammar (CCG), we say that a grammar rule is CPS transformation, if the category in the rule corresponds to Definition 2. In addition to Plotkin's work, the following is another transformation motivated by the linguistic observation.

Definition 3 (Barker's CPS transformation [1]). *$\{\!|\cdot|\!\}$ is recursively defined as a map from a DS lambda term to a CPS lambda term, where a is an arbitrary constant, and M and N are meta variables. Further, m, n, k represent the continuations of each term.*

$$\{\!|a|\!\} \equiv \lambda k.ka \qquad \{\!|MN|\!\} \equiv \lambda k.\{\!|M|\!\}(\lambda m.\{\!|N|\!\}(\lambda n.kmn))$$

In Definition 3, the lambda abstraction from the CPS transformation is omitted and also the continuation of N is rearranged from $\lambda n.mnk$ to $\lambda n.kmn$. As a result, types in the transformation are simplified as follows.

Definition 4 (Type of Barker's CPS transformation [1]). *$\{\!\langle\cdot\rangle\!\}$ is defined as follows, where capital letters are types of simply-typed lambda calculus. Further, A is the answer type uniquely determined in the global context. $\{\!\langle\cdot\rangle\!\}$ is corresponding to $\{\!|\cdot|\!\}$ in Definition 3.*

$$\{\!\langle X\rangle\!\}_A = (X \to A) \to A$$

The type of Barker's CPS transformation is exactly the same as the type-raising rule. Thus, the transformation is the generalized version of CCG.

2.2 Lambek Calculus

In this paper, we write α/β and $\beta\backslash\alpha$ as the implication from β to α.

Definition 5 (Lambek Calculus LC [5, 4]). *Lambek calculus* LC *is defined as a sequent calculus, which consists of atomic terms, right functional terms $\cdot\backslash\cdot$, and left functional terms $\cdot/\cdot$. The following are an initial sequent and deduction rules. A lower Greek letter is a term for a sequent and a capital Greek letter is a sequence of terms.*

$$\frac{}{\alpha \vdash \alpha}\,\mathrm{Id} \qquad \frac{\Sigma \vdash \alpha \qquad \Gamma, \alpha, \Delta \vdash \beta}{\Gamma, \Sigma, \Delta \vdash \beta}\,\mathrm{Cut} \qquad \frac{\Gamma, \alpha \vdash \beta}{\Gamma \vdash \beta/\alpha}\,I/$$

$$\frac{\alpha, \Gamma \vdash \beta}{\Gamma \vdash \alpha\backslash\beta}\,I\backslash \qquad \frac{\Sigma \vdash \alpha \qquad \Gamma, \beta, \Delta \vdash \gamma}{\Gamma, \Sigma, \alpha\backslash\beta, \Delta \vdash \gamma}\,\backslash I \qquad \frac{\Gamma, \beta, \Delta \vdash \gamma \qquad \Sigma \vdash \alpha}{\Gamma, \beta/\alpha, \Sigma, \Delta \vdash \gamma}\,/I$$

LC is the mathematical formalization of categorial grammar (CG). Hence, some of CCG rules are provable in this system. For instance, the application is provable.

$$\frac{\alpha \vdash \alpha \qquad \beta \vdash \beta}{\alpha, \alpha\backslash\beta \vdash \beta}\backslash I$$

In Definition 5, we have included *Cut* in the basic rules, however, it is known that we can exclude *Cut* from the basic rules.

Theorem 1 (Cut elimination in LC [5]). *If $\Sigma \vdash \alpha$ and $\Gamma, \alpha, \Delta \vdash \beta$ are both provable without Cut, then $\Gamma, \Sigma, \Delta \vdash \beta$ is also provable.*

Hereafter, we omit *Cut* from *LC* for the sake of brevity. Since the *Cut* rule produces a category which does not appear in the conclusion, the cut-free proof is important to show that a given sequent is unprovable in LC. For instance, we show two lemmas in both cases that is provable and unprovable in LC.

Lemma 1 (Provability of type-raising in LC). *$NP \Rightarrow S/(NP\backslash S)$ is a type-raising rule, which switches the biting relation from 'NP $NP\backslash S$' to 'S/(NP\S) NP\S'. Then, the sequent $\alpha \vdash \beta/(\alpha\backslash\beta)$ and $\alpha \vdash (\beta/\alpha)\backslash\beta$ are provable in* LC.

$$\frac{\dfrac{\alpha \vdash \alpha \qquad \beta \vdash \beta}{\alpha, \alpha\backslash\beta \vdash \beta}\backslash I}{\alpha \vdash \beta/(\alpha\backslash\beta)}I/ \qquad \frac{\dfrac{\alpha \vdash \alpha \qquad \beta \vdash \beta}{\alpha/\beta, \beta \vdash \beta}/I}{\alpha \vdash (\beta/\alpha)\backslash\beta}I\backslash$$

Lemma 2 (Unprovability of cross-composition in LC).
Let α, β be atomic. The sequent $\alpha/\beta, \alpha\backslash\gamma \vdash \gamma/\beta$ is unprovable in LC.

Proof. We must show that there is no way to derive the goal sequent from the initial sequent. Practically, there are three sequents that are unprovable in LC as follows, where the atomic term φ is different from the atomic term ψ.

$$\vdash \varphi \qquad\qquad \varphi \vdash \psi \qquad\qquad \varphi, \psi \vdash \varphi$$

We search all the possible rule applications in LC and show that each derivation path closes with an unprovable sequent. Assume that α, β, γ are atomic. In Fig. 1, we show a graph of the derivation path from the goal sequent $p2$. Note that the sequent is unprovable if at least one sequent in presumptions is unprovable. The red edge is $I\backslash$. The blue edge is $I/$. The green edge is $\backslash I$. The purple edge is $/I$. All leaves close with the unprovable sequent.

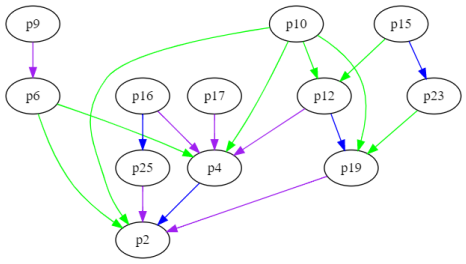

Fig. 1. Unprovability of cross-composition

$p9 : \vdash \beta$ $p6 : \alpha/\beta \vdash \alpha$ $p10 : \vdash \alpha$ $p15 : \gamma, \beta \vdash \gamma$
$p12 : \alpha, a\backslash\gamma, \beta \vdash \gamma$ $p16 : \alpha, \beta \vdash \gamma$ $p17 : \alpha \vdash \gamma$ $p4 : \alpha/\beta, \alpha\backslash\gamma, \beta \vdash \gamma$
$p23 : \gamma \vdash \gamma/\beta$ $p19 : \alpha, a\backslash\gamma \vdash \gamma/\beta$ $p25 : \alpha \vdash \gamma/\beta$ $p2 : \alpha/\beta, \alpha\backslash\gamma \vdash \gamma/b$

□

3 Provability of Continuation-Passing Style Transformation

The lambda calculus is not directed in implication, while LC is two-directional. In other words, if we translate the type properties of the lambda calculus into LC, the translation is multiple. Thus, the CPS transformation is not unique in LC. For instance, Barker's CPS transformation becomes as follows.

$$\alpha \vdash \beta/(\alpha\backslash\beta) \qquad\qquad \alpha \vdash (\beta/\alpha)\backslash\beta$$

By Lemma 1, the both sequents are provable. Thus, the Barker's CPS transformation is harmless as to provability in Lambek Calculus if we add them as a basic rule. On the other hand, Plotkin's CPS transformation is not.

Definition 6 (Plotkin's CPS transformation in LC). *Let γ be an answer type and A be a set of all atomic terms of* LC. *Two relations $\cdot \xrightarrow{\gamma}_{\rightarrow} \cdot$ and $\cdot \xrightarrow{\gamma}_{\searrow} \cdot$ are inductively defined as $\langle\!\langle\cdot\rangle\!\rangle$ and $\langle\cdot\rangle$, respectively.*

$$\frac{\alpha \xrightarrow{\gamma}_{\rightarrow} \tau}{\alpha \xrightarrow{\gamma}_{\rightarrow} \gamma/(\tau\backslash\gamma)} \qquad \frac{\alpha \xrightarrow{\gamma}_{\rightarrow} \tau}{\alpha \xrightarrow{\gamma}_{\rightarrow} (\gamma/\tau)\backslash\gamma} \qquad \frac{\alpha \in A}{\alpha \xrightarrow{\gamma}_{\searrow} \alpha} \qquad \frac{\alpha \xrightarrow{\gamma}_{\searrow} \tau \qquad \beta \xrightarrow{\gamma}_{\rightarrow} \upsilon}{\alpha\backslash\beta \xrightarrow{\gamma}_{\searrow} \tau\backslash\upsilon} \qquad \frac{\alpha \xrightarrow{\gamma}_{\searrow} \tau \qquad \beta \xrightarrow{\gamma}_{\rightarrow} \upsilon}{\beta/\alpha \xrightarrow{\gamma}_{\searrow} \upsilon/\tau}$$

Compared to the Barker's work, we execute the transformation recursively. Thus, the size of resulting term increase exponentially. Moreover, we must try

the all possible rules to find a derivation path from the goal sequent to leaves. Since the generated proof is lengthy, we only show the graph of the unprovable derivation paths in Fig. 2.

Theorem 2 (Unprovability of Plotkin's CPS transformation in LC). *There exists an unprovable sequent $\varphi \vdash \psi$ even if $\varphi \xrightarrow{\delta} \psi$.*

Proof. We consider the sequent $\gamma/(\beta/\alpha) \vdash \psi$ where α, β, γ are atomic and $\gamma/(\beta/\alpha) \xrightarrow{\delta} \psi$. Here, $\psi \equiv \delta/(((\delta/(\gamma\backslash\delta))/((\delta/(\beta\backslash\delta))/\alpha))\backslash\delta)$, which is a CPS-transformed term. We search all the possible rule applications in LC and show that each derivation path closes with an unprovable sequent.

$p17 : \gamma, \delta \vdash \gamma$
$p14 : \gamma, (\delta/(\beta\backslash\delta))/\alpha \vdash \gamma$
$p27 : \delta \vdash \beta$
$p29 : \delta/(\beta\backslash\delta) \vdash \beta$
$p33 : \delta \vdash \beta/\alpha$
$p21 : (\delta/(\beta\backslash\delta))/\alpha \vdash \beta/\alpha$
$p35 : \gamma/(\beta/\alpha), \delta/(\beta\backslash\delta) \vdash \gamma$
$p46 : \delta \vdash \gamma$
$p43 : (\delta/(\beta\backslash\delta))/\alpha \vdash \gamma$
$p53 : \gamma, \delta, \gamma\backslash\delta \vdash \delta$
$p51 : \gamma, \delta/(\beta\backslash\delta), \gamma\backslash\delta \vdash \delta$
$p49 : \gamma, (\delta/(\beta\backslash\delta))/\alpha, \gamma\backslash\delta \vdash \delta$
$p73 : \gamma/(\beta/\alpha), \delta, \gamma\backslash\delta \vdash \delta$
$p65 : \gamma/(\beta/\alpha), \delta/(\beta\backslash\delta), \gamma\backslash\delta \vdash \delta$
$p10 : \gamma/(\beta/\alpha), (\delta/(\beta\backslash\delta))/\alpha, \gamma\backslash\delta \vdash \delta$
$p87 : \gamma, \delta/(\beta\backslash\delta) \vdash \delta/(\gamma\backslash\delta)$
$p109 : \gamma/(\beta/\alpha), \delta \vdash \delta/(\gamma\backslash\delta)$
$p8 : \gamma/(\beta/\alpha), (\delta/(\beta\backslash\delta))/\alpha \vdash \delta/(\gamma\backslash\delta)$
$p6 : \gamma/(\beta/\alpha) \vdash (\delta/(\gamma\backslash\delta))/((\delta/(\beta\backslash\delta))/\alpha)$
$p136 : \beta \vdash \gamma$
$p133 : \gamma\backslash\delta \vdash \beta\backslash\delta$
$p143 : \beta \vdash \delta$
$p138 : \delta/(\beta\backslash\delta) \vdash \delta$
$p147 : \delta \vdash \delta/(\gamma\backslash\delta)$
$p123 : (\delta/(\beta\backslash\delta))/\alpha \vdash \delta/(\gamma\backslash\delta)$
$p149 : \gamma, ((\delta/(\gamma\backslash\delta))/((\delta/(\beta\backslash\delta))/\alpha))\backslash\delta \vdash \delta$
$p151 : \gamma \vdash \delta/(((\delta/(\gamma\backslash\delta))/((\delta/(\beta\backslash\delta))/\alpha))\backslash\delta)$

$p16 : \gamma, \delta/(\beta\backslash\delta) \vdash \gamma$
$p26 : \delta, \alpha \vdash \beta$
$p25 : \delta/(\beta\backslash\delta), \alpha \vdash \beta$
$p23 : (\delta/(\beta\backslash\delta))/\alpha, \alpha \vdash \beta$
$p31 : \delta/(\beta\backslash\delta) \vdash \beta/\alpha$
$p39 : \gamma/(\beta/\alpha), \delta \vdash \gamma$
$p12 : \gamma/(\beta/\alpha), (\delta/(\beta\backslash\delta))/\alpha \vdash \gamma$
$p45 : \delta/(\beta\backslash\delta) \vdash \gamma$
$p47 : \vdash \gamma$
$p54 : \gamma, \delta \vdash \delta$
$p56 : \gamma, \delta/(\beta\backslash\delta) \vdash \delta$
$p63 : \gamma \vdash \delta$
$p81 : \gamma/(\beta/\alpha), \delta \vdash \delta$
$p83 : \gamma/(\beta/\alpha), \delta/(\beta\backslash\delta) \vdash \delta$
$p89 : \gamma, \delta \vdash \delta/(\gamma\backslash\delta)$
$p85 : \gamma, (\delta/(\beta\backslash\delta))/\alpha \vdash \delta/(\gamma\backslash\delta)$
$p99 : \gamma/(\beta/\alpha), \delta/(\beta\backslash\delta) \vdash \delta/(\gamma\backslash\delta)$
$p119 : \gamma \vdash (\delta/(\gamma\backslash\delta))/((\delta/(\beta\backslash\delta))/\alpha)$
$p129 : \delta, \gamma\backslash\delta \vdash \delta$
$p135 : \beta, \gamma\backslash\delta \vdash \delta$
$p127 : \delta/(\beta\backslash\delta), \gamma\backslash\delta \vdash \delta$
$p142 : \vdash \beta\backslash\delta$
$p125 : (\delta/(\beta\backslash\delta))/\alpha, \gamma\backslash\delta \vdash \delta$
$p145 : \delta/(\beta\backslash\delta) \vdash \delta/(\gamma\backslash\delta)$
$p121 : \vdash (\delta/(\gamma\backslash\delta))/((\delta/(\beta\backslash\delta))/\alpha)$
$p4 : \gamma/(\beta/\alpha), ((\delta/(\gamma\backslash\delta))/((\delta/(\beta\backslash\delta))/\alpha))\backslash\delta \vdash \delta$
$p2 : \gamma/(\beta/\alpha) \vdash \delta/(((\delta/(\gamma\backslash\delta))/((\delta/(\beta\backslash\delta))/\alpha))\backslash\delta)$

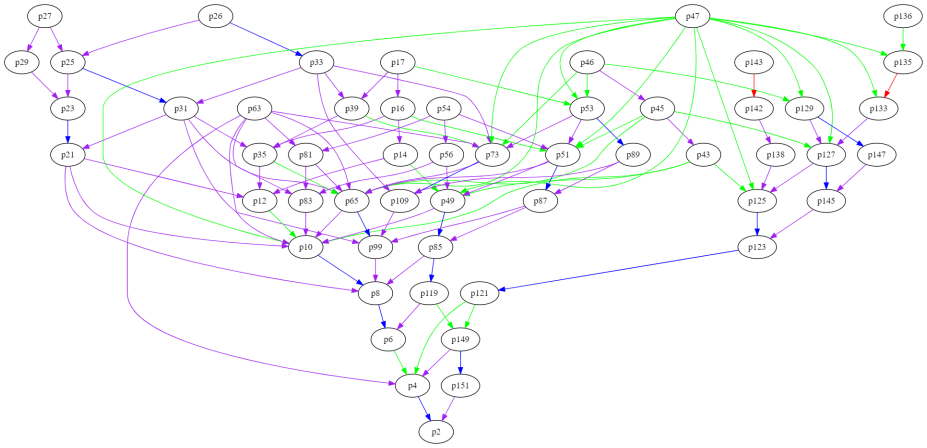

Fig. 2. Proof of unprovability

□

We showed Baker's work is provable in LC and Plotkin's work is not. Finally, we propose a moderate CPS translation which is the same translation of Plotkin's work but we restrict the types of lambda term. Our CPS translation is closer to Plotkin's work than Barker's, but is still provable in LC.

Definition 7 (Type-restricted CPS transformation in LC). *Let γ be an answer term and A be a set of all atomic terms of* LC*. We inductively define two relations $\cdot \overset{\gamma}{\twoheadrightarrow} \cdot$ and $\cdot \overset{\gamma}{\rightarrow\!\!\!\!\!\rightarrow} \cdot$ corresponding to $\langle\!\langle\cdot\rangle\!\rangle$ and $\langle\cdot\rangle$, respectively.*

$$\frac{\alpha \overset{\gamma}{\rightarrow\!\!\!\!\!\rightarrow} \tau}{\alpha \overset{\gamma}{\twoheadrightarrow} \gamma/(\tau\backslash\gamma)} \qquad \frac{\alpha \overset{\gamma}{\rightarrow\!\!\!\!\!\rightarrow} \tau}{\alpha \overset{\gamma}{\twoheadrightarrow} (\gamma/\tau)\backslash\gamma} \qquad \frac{\alpha \in A}{\alpha \overset{\gamma}{\rightarrow\!\!\!\!\!\rightarrow} \alpha} \qquad \frac{\alpha \in A \qquad \beta \overset{\gamma}{\twoheadrightarrow} \upsilon}{\alpha\backslash\beta \overset{\gamma}{\rightarrow\!\!\!\!\!\rightarrow} \alpha\backslash\upsilon} \qquad \frac{\alpha \in A \qquad \beta \overset{\gamma}{\twoheadrightarrow} \upsilon}{\beta/\alpha \overset{\gamma}{\rightarrow\!\!\!\!\!\rightarrow} \upsilon/\alpha}$$

Note that there is no transformation from a higher-order functional term, which recursively takes other functional terms, *e.g.*, $\gamma/(\beta/\alpha)$, that caused failure of proof in Plotkin's work. Theorem 3 shows that the transformation in Definition 7 is provable in LC.

Theorem 3 (Provability of type-restricted CPS transformation in LC). *Let γ, φ and ψ be terms of* LC*. Then, (i) $\alpha \vdash \beta$ if $\alpha \overset{\gamma}{\twoheadrightarrow} \beta$, and moreover, (ii) $\alpha \vdash \beta$ if $\alpha \overset{\gamma}{\rightarrow\!\!\!\!\!\rightarrow} \beta$.*

Proof. We mutually prove both statements (i) and (ii) by mathematical induction with respect to the length of α.

1. Assume that α is atomic.
   (a) Assume $\alpha \overset{\gamma}{\rightarrow\!\!\!\!\!\rightarrow} \alpha$. $\alpha \vdash \alpha$ holds.
   (b) Assume $\alpha \overset{\gamma}{\twoheadrightarrow} (\gamma/\alpha)\backslash\gamma$. $\alpha \vdash (\gamma/\alpha)\backslash\gamma$ holds by Lemma 1.

   (c) Assume $\alpha \xrightarrow{\gamma} \gamma/(\alpha\backslash\gamma)$. $\alpha \vdash \gamma/(\alpha\backslash\gamma)$ holds by Lemma 1.
2. Assume $\alpha = \beta\backslash\delta$ and $\delta \xrightarrow{\gamma} \eta$.
   (a) Assume $\alpha \xrightarrow{\gamma} \beta\backslash\eta$. As β is atomic by Definition 7, $\beta \vdash \beta$ holds. Moreover, $\delta \vdash \eta$ holds by the induction hypothesis. Thus, $\beta\backslash\delta \vdash \beta\backslash\eta$.
   (b) Assume $\alpha \xrightarrow{\gamma} \gamma/((\beta\backslash\eta)\backslash\gamma)$. As the same way as Proof 2a, $\beta\backslash\delta \vdash \beta\backslash\eta$. Thus, $\alpha \vdash \gamma/((\beta\backslash\eta)\backslash\gamma)$ holds by Lemma 1.
   (c) Assume $\alpha \xrightarrow{\gamma} (\gamma/(\beta\backslash\eta))\backslash\gamma$. As the same way as Proof 2a, $\beta\backslash\delta \vdash \beta\backslash\eta$. Thus, $\alpha \vdash (\gamma/(\beta\backslash\eta))\backslash\gamma$ holds by Lemma 1.
3. Assume $\alpha = \delta/\beta$ and $\delta \xrightarrow{\gamma} \eta$
   (a) Assume $\alpha \xrightarrow{\gamma} \eta/\beta$. As β is atomic by Definition 7, $\beta \vdash \beta$ holds. Moreover, $\delta \vdash \eta$ holds by the induction hypothesis. Thus $\delta/\beta \vdash \eta/\beta$.
   (b) Assume $\alpha \xrightarrow{\gamma} \gamma/((\eta/\beta)\backslash\gamma)$. As the same way as Proof 3a, $\delta/\beta \vdash \eta/\beta$. Thus, $\alpha \vdash \gamma/((\eta/\beta)\backslash\gamma)$ holds by Lemma 1.
   (c) Assume $\alpha \xrightarrow{\gamma} (\gamma/(\eta/\beta))\backslash\gamma$. As the same way as Proof 3a, $\delta/\beta \vdash \eta/\beta$. Thus, $\alpha \vdash (\gamma/(\eta/\beta))\backslash\gamma$ holds by Lemma 1.

Therefore, (i) and (ii) hold. □

4 Formalization in Isabelle/HOL

5 Conclusion

In this paper, we investigated Barker's and Plotkin's transformations using Lambek Calculus. Since the negative proofs by Lambek calculus were hard to verify manually due to the huge search space, we showed graphs of the paths of the proofs using Isabelle/HOL. The results are attached in the Appendix. We showed that Barker's work is provable in Lambek calculus, whereas Plotkin's work is unprovable. Finally, among the CPS transformations, we proposed a moderated boundary of CPS transformations with the restriction to be provable in Lambek calculus. Future work includes the analysis of Lambek calculus with substructural logic [6] in relation to Plotkin's CPS transformation.

References

1. Barker, C., Shan, C.c.: Continuations and natural language, vol. 53. Oxford Studies in Theoretical (2014)
2. Bekki, D., Asai, K.: Representing covert movements by delimited continuations. In: JSAI International Symposium on Artificial Intelligence. pp. 161–180. Springer (2009)
3. Hindley, J.R., Seldin, J.P.: Lambda-calculus and Combinators, an Introduction, vol. 2. Cambridge University Press Cambridge (2008)
4. Kanazawa, M.: The lambek calculus enriched with additional connectives. Journal of Logic, Language and Information **1**(2), 141–171 (1992)
5. Lambek, J.: The mathematics of sentence structure. The American Mathematical Monthly **65**(3), 154–170 (1958)

6. Ono, H.: Substructural logics and residuated lattices—an introduction. Trends in logic pp. 193–228 (2003)
7. Plotkin, G.D.: Call-by-name, call-by-value and the λ-calculus. Theoretical computer science **1**(2), 125–159 (1975)
8. Steedman, M.: The syntactic process. MIT press (2001)

Appendix

We attach the formal proof that the cross composition is unprovable in LC in Isabelle/HOL. We could also show the proof of Theorem 2 in the same way.

```
datatype 'a category =
  Atomic 'a ("^")
  | RightFunctional "'a category" "'a category"
    (infix "→" 60)
  | LeftFunctional "'a category" "'a category"
    (infix "←" 60)
inductive
  LC:: "'a category list ⇒ 'a category ⇒ bool"
  (infix "⇒" 55)
  where
    r1: "([x] ⇒ x)"
  | r2: "(X@[x] ⇒ y) ⟹ (X ⇒ y ← x)"
  | r3: "([x]@X ⇒ y) ⟹ (X ⇒ x → y)"
  | r4: "⟦(Y ⇒ y); (X@[x]@Z ⇒ z)⟧ ⟹ (X@Y@[y →
x]@Z ⇒ z)"
  | r5: "⟦(X@[x]@Z ⇒ z); (Y ⇒ y)⟧ ⟹ (X@[x ←
y]@Y@Z ⇒ z)"
theorem
  assumes "distinct [a,b,c,d]"
  shows "¬([^a ←^b,^a →^c] ⇒^c ←^b)"
proof -
have p9: "¬([] ⇒^b)"
  apply auto apply(subst(asm)LC.simps) by auto
have p6: "¬([^a ← ^b] ⇒^a)"
  apply auto apply(subst(asm)LC.simps) apply auto
  by  (simp add: p9 Cons_eq_append_conv)+
have p10: "¬([] ⇒^a)"
  apply auto apply(subst(asm)LC.simps) by auto
have p15: "¬([^c,^b] ⇒^c)"
  apply auto apply(subst(asm)LC.simps) apply auto
  by (simp add: p10 Cons_eq_append_conv)+
have p12: "¬([^a,^a → ^c,^b] ⇒^c)"
  apply auto apply(subst(asm)LC.simps) apply auto
  apply (simp_all add: Cons_eq_append_conv)+
  using p10 p15 by fastforce
have p16: "¬([^a,^b] ⇒^c)"
  apply auto apply(subst(asm)LC.simps) apply auto
  by (simp_all add: Cons_eq_append_conv)+
have p17: "¬([^a] ⇒^c)"
  apply auto apply(subst(asm)LC.simps) apply auto
  apply (simp_all add: Cons_eq_append_conv)+
  using assms by auto
have p4: "¬([^a ← ^b,^a → ^c,^b] ⇒^c)"
  apply auto apply(subst(asm)LC.simps) apply auto
  apply (simp_all add: Cons_eq_append_conv)+
  using p10 p15 p12 p16 p17 by fastforce+
have p23: "¬([^c] ⇒^c ← ^b)"
  apply auto apply(subst(asm)LC.simps) apply auto
  apply (simp_all add: Cons_eq_append_conv)+
  by (simp add: p15)
have p19: "¬([^a,^a → ^c] ⇒^c ← ^b)"
  apply auto apply(subst(asm)LC.simps) apply auto
  apply (simp_all add: Cons_eq_append_conv)+
  using p10 p23 p12 by auto+
have p25: "¬([^a] ⇒^c ← ^b)"
  apply auto apply(subst(asm)LC.simps) apply auto
  apply (simp_all add: Cons_eq_append_conv)+
  by (simp add: p16)
show p2: "¬([^a ← ^b,^a → ^c] ⇒^c ← ^b)"
  apply auto apply(subst(asm)LC.simps) apply auto
  apply (simp_all add: Cons_eq_append_conv)+
using p10 p23 p4 p25 p9 by fastforce+
qed
```