Undecidability of Type-Checking in Domain-Free Typed Lambda-Calculi with Existence

Koji Nakazawa[1,⋆], Makoto Tatsuta[2],
Yukiyoshi Kameyama[3], and Hiroshi Nakano[4]

[1] Graduate School of Informatics, Kyoto University, Kyoto 606-8501, Japan
[2] National Institute of Informatics, Japan
[3] Department of Computer Science, University of Tsukuba, Japan
[4] Department of Applied Mathematics and Informatics, Ryukoku University, Japan

Abstract. This paper shows undecidability of type-checking and type-inference problems in domain-free typed lambda-calculi with existential types: a negation and conjunction fragment, and an implicational fragment. These are proved by reducing type-checking and type-inference problems of the domain-free polymorphic typed lambda-calculus to those of the lambda-calculi with existential types by continuation passing style translations.

Keywords: undecidability, existential type, CPS-translation, domain-free type system.

1 Introduction

Existential types correspond to second-order existence in logic by the Curry-Howard isomorphism, so they are a natural notion from the point of view of logic. They have been also actively studied from the point of view of computer science since Mitchell and Plotkin [7] showed that abstract data types are existential types.

Existential types are also important since, together with negation and conjunction, it gives a suitable target calculus for continuation-passing style (CPS) translations. Thielecke showed that the negation ($\neg$) and conjunction ($\wedge$) fragment of a λ-calculus suffices for a CPS calculus [14] as the target of various first-order calculi. Recent studies on CPS translations for polymorphic calculi have shown that the $\neg \wedge \exists$-fragment of λ-calculus is an essence of a target calculus of CPS translations for various systems, such as the polymorphic typed λ-calculus [4], the $\lambda\mu$-calculus [3,5], and delimited continuations. [6] showed that a $\neg \wedge \exists$-fragment is even more suitable as a target calculus of a CPS translation for delimited continuations such as `shift` and `reset` [2].

Domain-free type systems, which are in an intermediate style between Church- and Curry-style, are useful for having the subject reduction property. In domain-free style λ-calculus, the type of a bound variable is not explicit in $\lambda x.M$ as in

⋆ knak@kuis.kyoto-u.ac.jp

M. Kaminski and S. Martini (Eds.): CSL 2008, LNCS 5213, pp. 477–491, 2008.

Curry-style, while as in Church-style, terms may contain type information for second-order quantifiers, such as a type abstraction $\lambda X.M$ for $\forall$-introduction rule, and a term $\langle A, M\rangle$ with a witness A for $\exists$-introduction rule. Domain-free type systems are introduced for a study on the $\lambda\mu$-calculus. [9] showed the Curry-style call-by-value $\lambda\mu$-calculus does not enjoy the subject reduction property, and [3] introduced a domain-free $\lambda\mu$-calculus $\lambda_V\mu$ to have the subject reduction. In addition, the $\neg \wedge \exists$-fragment of the domain-free typed λ-calculus works as a target calculus of a CPS translation for $\lambda_V\mu$.

Type-inhabitation (INH) is a problem that asks whether there exists M such that $\vdash M : A$ is derivable for given A. INH corresponds to provability of the formula A. The other properties of typed λ-calculi are decidability of type-checking and type-inference. Type-checking (TC) is a problem that asks whether $\Gamma \vdash M : A$ is derivable for given Γ, M, and A. Type-inference (TI) is a problem that asks whether there exist Γ and A such that $\Gamma \vdash M : A$ is derivable for given M. These three questions are fundamentally important in computer science.

Although λ-calculi with existential types are important as computational systems, their properties have not been studied enough yet. It is only recent that INH in the $\neg \wedge \exists$-fragment was proved to be decidable in [13]. TC and TI in typed λ-calculi with existential types remained unknown until this paper.

This paper proves undecidability of the type-checking and the type-inference problems in domain-free typed λ-calculi with existential types: (1) a $\neg \wedge \exists$-fragment $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$, (2) another $\neg \wedge \exists$-fragment $\mathsf{DF}\text{-}\lambda_g^{\neg\wedge\exists}$ with a generalized $\wedge$-elimination rule, and (3) an $\rightarrow\exists$-fragment $\mathsf{DF}\text{-}\lambda^{\rightarrow\exists}$.

Our results show that the system $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$ is marginal and interesting, because Tatsuta et al [13] showed the decidability of its INH, while ours shows the undecidability of its TC and TI. So far we know few type systems that have this property.

In order to prove undecidability of TC and TI in $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$, $\mathsf{DF}\text{-}\lambda_g^{\neg\wedge\exists}$, and $\mathsf{DF}\text{-}\lambda^{\rightarrow\exists}$, we reduce it to undecidability of TC and TI in the domain-free polymorphic typed λ-calculus $\mathsf{DF}\text{-}\lambda 2$. For $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$, we define a negative translation $(\cdot)^\bullet$ from types of $\mathsf{DF}\text{-}\lambda 2$ to types of $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$, and a translation $[\![\cdot]\!]$ from terms of $\mathsf{DF}\text{-}\lambda 2$ to terms of $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$, which is a variant of call-by-name CPS translations inspired by [4]. We will show that $\Gamma \vdash M : A$ is derivable in $\mathsf{DF}\text{-}\lambda 2$ if and only if $\neg\Gamma^\bullet \vdash [\![M]\!] : \neg A^\bullet$ is derivable in $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$. By this fact, we can reduce TC of $\mathsf{DF}\text{-}\lambda 2$ to that of $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$, which concludes undecidability of TC of $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$.

The key of the proof is as follows. For a term M, a type derivation of $\neg\Gamma^\bullet \vdash [\![M]\!] : \neg A^\bullet$ in $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$ may contain a type B which is not any CPS type, where a CPS type is defined as a type of the form $\neg C^\bullet$ for some type C in $\mathsf{DF}\text{-}\lambda 2$. If a derivation contains such a type B, it does not correspond to any derivation in $\mathsf{DF}\text{-}\lambda 2$. However, in fact, we can define a contraction transformation that maps a type to a CPS type so that by the contraction transformation, from any type derivation of $\neg\Gamma^\bullet \vdash [\![M]\!] : \neg A^\bullet$, we can construct another type derivation of the same judgment in which every type is a CPS type. By this we can pull back it into a derivation in $\mathsf{DF}\text{-}\lambda 2$.

Systems	TC	TI	INH
Curry-$\lambda 2$	no[16]	no[16]	no
DF-$\lambda 2$	no[1]	no[1]	
Curry-$\lambda^{\neg\wedge\exists}$	?	?	yes[13]
DF-$\lambda^{\neg\wedge\exists}$	NO	NO	
Curry-$\lambda^{\rightarrow\exists}$	?	?	?
DF-$\lambda^{\rightarrow\exists}$	NO	NO	

Fig. 1. Decidability of TC, TI and INH

We summarize related results about decidability of TC, TI, and INH in several systems in Figure 1, where DF means domain-free, and NO denotes the main results of this paper.

Section 2 introduces the domain-free typed λ-calculus DF-$\lambda^{\neg\wedge\exists}$ with negation, conjunction and existence. Section 3 gives our main theorem which states undecidability of TC and TI in DF-$\lambda^{\neg\wedge\exists}$. Section 4 proves the main theorem, and applies the proof method to DF-$\lambda_g^{\neg\wedge\exists}$. Section 5 discusses CPS-translations for various systems to show that DF-$\lambda^{\neg\wedge\exists}$ is an essence of a target of CPS translations. Section 6 shows undecidability of TC and TI in a domain-free typed λ-calculus DF-$\lambda^{\rightarrow\exists}$ with implication and existence.

2 Typed λ-Calculus with Negation, Conjunction and Existence

In this section, we introduce the negation ($\neg$), conjunction ($\wedge$), and existence ($\exists$) fragment DF-$\lambda^{\neg\wedge\exists}$ of domain-free typed λ-calculus.

Definition 1 (DF-$\lambda^{\neg\wedge\exists}$). *(1) The types (denoted by* A, B,*... , and called* $\neg\wedge\exists$*-types) and the terms (denoted by* M, N,*...) of* DF-$\lambda^{\neg\wedge\exists}$ *are defined by*

$$\mathsf{A} ::= X \mid \bot \mid \neg\mathsf{A} \mid \mathsf{A}\wedge\mathsf{A} \mid \exists X.\mathsf{A},$$
$$M ::= x \mid \lambda x.M \mid \langle M, M\rangle \mid \langle \mathsf{A}, M\rangle \mid MM \mid M\pi_1 \mid M\pi_2 \mid M[Xx.M],$$

where X *and* x *denote a type variable and a term variable, respectively. In the type* $\exists X.\mathsf{A}$*, the variable* X *is bound in* A*. In the term* $\lambda x.M$*, the variable* x *is bound in* M*. In the term* $N[Xx.M]$*, the variables* X *and* x *are bound in* M*. We use* $\equiv$ *to denote syntactic identity modulo renaming of bound variables.*

(2) Γ *denotes a context, which is a finite set of type assignments in the form of* $(x : \mathsf{A})$*. We suppose that if both* $(x : \mathsf{A})$ *and* $(x : \mathsf{B})$ *are in* Γ*,* $\mathsf{A} \equiv \mathsf{B}$ *holds. We write* $\Gamma, x : \mathsf{A}$ *for* $\Gamma \cup \{x : \mathsf{A}\}$*, and* Γ_1, Γ_2 *for* $\Gamma_1 \cup \Gamma_2$*.* $\neg\Gamma$ *is defined as* $\{(x : \neg\mathsf{A}) | (x : \mathsf{A}) \in \Gamma\}$*. The typing rules of* DF-$\lambda^{\neg\wedge\exists}$ *are the following.*

$$\frac{}{\Gamma, x : \mathsf{A} \vdash x : \mathsf{A}}\ (\mathsf{Ax})$$

$$\frac{\Gamma, x : \mathsf{A} \vdash M : \bot}{\Gamma \vdash \lambda x.M : \neg\mathsf{A}}\ (\neg\mathsf{I}) \qquad \frac{\Gamma_1 \vdash M : \neg\mathsf{A} \quad \Gamma_2 \vdash N : \mathsf{A}}{\Gamma_1, \Gamma_2 \vdash MN : \bot}\ (\neg\mathsf{E})$$

$$\frac{\Gamma_1 \vdash M : \mathsf{A} \quad \Gamma_2 \vdash N : \mathsf{B}}{\Gamma_1, \Gamma_2 \vdash \langle M, N\rangle : \mathsf{A} \wedge \mathsf{B}}\ (\wedge\mathsf{I}) \qquad \frac{\Gamma \vdash N : \mathsf{A}[X := \mathsf{B}]}{\Gamma \vdash \langle \mathsf{B}, N\rangle : \exists X.\mathsf{A}}\ (\exists\mathsf{I})$$

$$\frac{\Gamma \vdash M : \mathsf{A}_1 \wedge \mathsf{A}_2}{\Gamma \vdash M\pi_1 : \mathsf{A}_1}\ (\wedge\mathsf{E1}) \qquad \frac{\Gamma \vdash M : \mathsf{A}_1 \wedge \mathsf{A}_2}{\Gamma \vdash M\pi_2 : \mathsf{A}_2}\ (\wedge\mathsf{E2})$$

$$\frac{\Gamma_1 \vdash M : \exists X.\mathsf{A} \quad \Gamma_2, x : \mathsf{A} \vdash N : \mathsf{C}}{\Gamma_1, \Gamma_2 \vdash M[Xx.N] : \mathsf{C}}\ (\exists\mathsf{E})$$

$\mathsf{A}[X := \mathsf{B}]$ *is the ordinary capture-avoiding substitution for types. In the rule* $(\exists\mathsf{E})$*,* Γ_2 *and* C *must not contain* X *freely. We write* $\Gamma \vdash_{\lambda^{\neg\wedge\exists}} M : \mathsf{A}$ *to denote that* $\Gamma \vdash M : \mathsf{A}$ *is derivable by the typing rules above.*

In Section 5, we will show this calculus is useful for a target of CPS translations. In addition, $\lambda^{\neg\wedge\exists}$ represents every function representable in the polymorphic typed λ-calculus, because of a CPS-translation from the polymorphic typed λ-calculus to this calculus.

3 Type-Checking and Type-Inference

Type-inhabitation (INH) is a problem that asks whether there exists M such that $\vdash M : A$ is derivable for given A, which corresponds to provability of the formula A. In [13], INH of $\lambda^{\neg\wedge\exists}$ was proved to be decidable. Moreover, it immediately implies decidability of INH in $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$.

Type-checking (TC) is a problem that asks whether $\Gamma \vdash M : A$ is derivable for given Γ, M, and A. Type-inference (TI) is a problem that asks whether there exist Γ and A such that $\Gamma \vdash M : A$ is derivable for given M.

Theorem 1. *Type-checking and type-inference of* $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$ *are undecidable.*

This theorem is proved in the Section 4.

4 Proof of Undecidability of TC and TI in $\mathsf{DF}\text{-}\boldsymbol{\lambda}^{\neg\wedge\exists}$

This section will prove Theorem 1. The subsection 4.1 will give a definition of a domain-free polymorphic typed λ-calculus $\mathsf{DF}\text{-}\lambda 2$. The subsection 4.2 will define a CPS translation from that calculus to $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$. We will also define an inverse CPS translation from the image $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}_{\mathsf{cps}}$ of the CPS translation to $\mathsf{DF}\text{-}\lambda 2$. The subsection 4.3 will show our main lemma, which states that $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$ is conservative over $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}_{\mathsf{cps}}$. The subsection 4.4 will finish our undecidability proof. Our proof method will be applied to a variant $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}_{g}$ with general elimination rules in the subsection 4.5.

4.1 Domain-Free Polymorphic Typed $\boldsymbol{\lambda}$-Calculus

In this subsection, we introduce the domain-free variant $\mathsf{DF}\text{-}\lambda 2$ of the polymorphic typed λ-calculus, for which TC and TI have been already known to be undecidable [1].

Definition 2 (DF-$\lambda 2$). *(1) The types (denoted by A, B,..., and called $\to\forall$-types), and the terms (denoted by M, N,...) of* DF-$\lambda 2$ *are defined by*

$A ::= X \mid A \to A \mid \forall X.A$,

$M ::= x \mid \lambda x.M \mid \lambda X.M \mid MM \mid MA$.

(2) The typing rules of DF-$\lambda 2$ *are the following.*

$$\frac{}{\Gamma, x : A \vdash x : A}\ (\mathsf{Ax})$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \to B}\ (\to\mathsf{I}) \qquad \frac{\Gamma_1 \vdash M : A \to B \quad \Gamma_2 \vdash N : A}{\Gamma_1, \Gamma_2 \vdash MN : B}\ (\to\mathsf{E})$$

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash \lambda X.M : \forall X.A}\ (\forall\mathsf{I}) \qquad \frac{\Gamma \vdash M : \forall X.A}{\Gamma \vdash MB : A[X := B]}\ (\forall\mathsf{E})$$

In the rule ($\forall$I), *the lower sequent must not contain X freely.*

Theorem 2 ([1]). *Type-checking and type-inference of* DF-$\lambda 2$ *are undecidable.*

4.2 CPS Translation

We give a CPS translation for DF-$\lambda 2$ in this subsection. Our translation is inspired by Fujita's translation in [4], but since it is in Church-style, we cannot use it directly for domain-free calculi, and we modified it appropriately.

Definition 3 (CPS Translation). *(1) The negative translation from $\to\forall$-types to $\neg \wedge \exists$-types is defined by*

$X^\bullet \equiv X$, $(A \to B)^\bullet \equiv \neg A^\bullet \wedge B^\bullet$, $(\forall X.A)^\bullet \equiv \exists X.A^\bullet$.

$\Gamma^\bullet$ is defined as $\{(x : \mathsf{A}^\bullet) \mid (x : \mathsf{A}) \in \Gamma\}$.

(2) The CPS translation from terms in DF-$\lambda 2$ *to terms in* DF-$\lambda^{\neg\wedge\exists}$ *is defined by*

$[\![x]\!] \equiv \lambda k.xk$,

$[\![\lambda x.M]\!] \equiv \lambda k.(\lambda x.[\![M]\!](k\pi_2))(k\pi_1)$,

$[\![\lambda X.M]\!] \equiv \lambda k.k[Xk'.[\![M]\!]k']$,

$[\![MN]\!] \equiv \lambda k.[\![M]\!]\langle [\![N]\!], k\rangle$,

$[\![MA]\!] \equiv \lambda k.[\![M]\!]\langle A^\bullet, k\rangle$,

where variables k and k' are supposed to be fresh.

Proposition 1. *$\Gamma \vdash_{\lambda 2} M : A$ implies $\neg\Gamma^\bullet \vdash_{\lambda^{\neg\wedge\exists}} [\![M]\!] : \neg A^\bullet$.*

Definition 4 (DF-$\lambda^{\neg\wedge\exists}_{\mathsf{cps}}$). *(1) The continuation types (denoted by $\mathcal{A}$, $\mathcal{B}$,...) and the CPS terms (denoted by P, Q,...) are defined as the image of the negative translation and that of the CPS translation, respectively. These are inductively defined by*

$\mathcal{A} ::= X \mid \neg\mathcal{A} \wedge \mathcal{A} \mid \exists X.\mathcal{A}$,

$P ::= \lambda k.xk \mid \lambda k.(\lambda x.P(k\pi_2))(k\pi_1) \mid \lambda k.k[Xk'.Pk'] \mid \lambda k.P\langle P, k\rangle \mid \lambda k.P\langle \mathcal{A}, k\rangle$,

where occurrences of k and k' denote those of the same variable, for example, $\lambda k.xk$ denotes $\lambda k_1.xk_1$ but does not denote $\lambda k_1.xk_2$ for $k_1 \not\equiv k_2$. The CPS types are defined as types of the form $\neg\mathcal{A}$.

(2) We define the subsystem $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}_{\mathsf{cps}}$ *of* $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$ *by restricting terms and types to CPS terms and CPS types, respectively. We write* $\neg\Gamma \vdash_{\mathsf{cps}} P : \neg\mathcal{A}$ *to denote that the judgment is derivable in* $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}_{\mathsf{cps}}$.

Definition 5 (Inverse CPS Translation). *The inverse translation* $(\cdot)^\circ$ *from continuation types to* $\to\forall$*-types is defined by*

$$X^\circ \equiv X,\ \ (\neg\mathcal{A}\wedge\mathcal{B})^\circ \equiv \mathcal{A}^\circ\to\mathcal{B}^\circ,\ \ (\exists X.\mathcal{A})^\circ \equiv \forall X.\mathcal{A}^\circ.$$

The inverse translation $(\cdot)^\#$ *from CPS terms to terms of* $\mathsf{DF}\text{-}\lambda 2$ *is defined by*

$$(\lambda k.xk)^\# \equiv x,$$
$$(\lambda k.(\lambda x.P(k\pi_2))(k\pi_1))^\# \equiv \lambda x.P^\#,$$
$$(\lambda k.k[Xk'.Pk'])^\# \equiv \lambda X.P^\#,$$
$$(\lambda k.P\langle Q,k\rangle)^\# \equiv P^\# Q^\#,$$
$$(\lambda k.P\langle \mathcal{A},k\rangle)^\# \equiv P^\# \mathcal{A}^\circ.$$

Lemma 1. *(1) For any* $\to\forall$*-type* A, $A^\bullet$ *is a continuation type, and* $A^{\bullet\circ} \equiv A$.
(2) For any $\mathsf{DF}\text{-}\lambda 2$*-term* M, $[\![M]\!]$ *is a CPS term, and* $[\![M]\!]^\# \equiv M$ *holds.*

Proposition 2. *(1) If* $\neg\Gamma \vdash_{\mathsf{cps}} P : \neg\mathcal{A}$ *holds, then* $\Gamma^\circ \vdash_{\lambda 2} P^\# : \mathcal{A}^\circ$ *holds.*
(2) If $\neg\Gamma^\bullet \vdash_{\mathsf{cps}} [\![M]\!] : \neg A^\bullet$, *then* $\Gamma \vdash_{\lambda 2} M : A$ *holds.*

Proof. (1) By induction on the derivation.
(2) By (1), we have $\Gamma^{\bullet\circ} \vdash_{\lambda 2} [\![M]\!]^\# : A^{\bullet\circ}$. By Lemma 1, we have the claim. □

4.3 Typing for CPS Terms in $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$

Proposition 1 shows that, for any typable term M in $\mathsf{DF}\text{-}\lambda 2$, $[\![M]\!]$ has a CPS type. In fact, its converse can be also proved. In order to prove that, in this subsection, we will show that $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$ is conservative over $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}_{\mathsf{cps}}$.

A type derivation of a CPS term in $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$ may contain a non CPS type. For example, a CPS term $Q \equiv \lambda k'.xk'$ has an arbitrary negation type $\neg\mathsf{A}$ under a context $\{x : \neg\mathsf{A}\}$, and then $P \equiv \lambda k.(\lambda x.Q(k\pi_2))(k\pi_1)$ has a type $\neg(\neg\mathsf{A}\wedge\mathsf{A})$ as

$$\dfrac{\dfrac{\dfrac{x:\neg\mathsf{A}\vdash Q:\neg\mathsf{A} \quad \dfrac{}{k:\neg\mathsf{A}\wedge\mathsf{A}\vdash k:\neg\mathsf{A}\wedge\mathsf{A}}}{k:\neg\mathsf{A}\wedge\mathsf{A}\vdash k\pi_2:\mathsf{A}}}{k:\neg\mathsf{A}\wedge\mathsf{A},x:\neg\mathsf{A}\vdash Q(k\pi_2):\bot}}{k:\neg\mathsf{A}\wedge\mathsf{A}\vdash\lambda x.Q(k\pi_2):\neg\neg\mathsf{A}} \qquad \dfrac{\dfrac{}{k:\neg\mathsf{A}\wedge\mathsf{A}\vdash k:\neg\mathsf{A}\wedge\mathsf{A}}}{k:\neg\mathsf{A}\wedge\mathsf{A}\vdash k\pi_1:\neg\mathsf{A}}$$
$$\dfrac{k:\neg\mathsf{A}\wedge\mathsf{A}\vdash(\lambda x.Q(k\pi_2))(k\pi_1):\bot}{\vdash\lambda k.(\lambda x.Q(k\pi_2))(k\pi_1):\neg(\neg\mathsf{A}\wedge\mathsf{A})},$$

where the type A may not be a continuation type, for example, A may be $X\wedge Y$. However, as we can see in the example, such a type A cannot be consumed in the type derivation of a CPS term, so we can replace A by any type without changing the form of the derivation. In general, we can define a translation $(\cdot)^{\mathsf{c}}$ from $\neg\wedge\exists$-types to CPS types such that, for any CPS term P and any type derivation of $\Gamma \vdash_{\lambda^{\neg\wedge\exists}} P : \mathsf{A}$, we have $\Gamma^{\mathsf{c}} \vdash_{\mathsf{cps}} P : \mathsf{A}^{\mathsf{c}}$. We call the translation $(\cdot)^{\mathsf{c}}$ the contraction translation. Moreover, we have $(\neg A^\bullet)^{\mathsf{c}} \equiv \neg A^\bullet$.

Definition 6 (Contraction Translation). *Let $\mathcal{S}$ be a fixed closed continuation type, such as $\exists X.X$. The contraction translation $(\cdot)^{\mathsf{c}}$ from $\neg \wedge \exists$-types to CPS types is defined by*

$$(\neg \mathsf{A})^{\mathsf{c}} \equiv \neg \mathsf{A}^{\mathsf{d}}, \qquad \mathsf{A}^{\mathsf{c}} \equiv \neg \mathcal{S} \quad (\mathsf{A} \textit{ is not a negation}),$$

$$X^{\mathsf{d}} \equiv X, \quad \bot^{\mathsf{d}} \equiv \mathcal{S}, \quad (\neg \mathsf{A})^{\mathsf{d}} \equiv \mathcal{S}, \quad (\mathsf{A} \wedge \mathsf{B})^{\mathsf{d}} \equiv \mathsf{A}^{\mathsf{c}} \wedge \mathsf{B}^{\mathsf{d}}, \quad (\exists X.\mathsf{A})^{\mathsf{d}} \equiv \exists X.\mathsf{A}^{\mathsf{d}}.$$

Γ^{c} is defined as $\{(x : \mathsf{A}^{\mathsf{c}}) | (x : \mathsf{A}) \in \Gamma\}$.

Lemma 2. *(1) For any continuation type $\mathcal{A}$, $(\neg \mathcal{A})^{\mathsf{c}} \equiv \neg \mathcal{A}$ and $\mathcal{A}^{\mathsf{d}} \equiv \mathcal{A}$ hold.*

(2) For any continuation type $\mathcal{A}$ and any $\neg \wedge \exists$-type B, $(\mathsf{B}[X := \mathcal{A}])^{\mathsf{c}} \equiv \mathsf{B}^{\mathsf{c}}[X := \mathcal{A}]$ and $(\mathsf{B}[X := \mathcal{A}])^{\mathsf{d}} \equiv \mathsf{B}^{\mathsf{d}}[X := \mathcal{A}]$ hold.

Proof. (1) By induction on $\mathcal{A}$.

(2) By induction on B. Note that any continuation type $\mathcal{A}$ is not a negation, so we have $\mathcal{A}^{\mathsf{c}} \equiv \neg \mathcal{S}$. □

Lemma 3 (Main Lemma). *For a CPS term P, $\Gamma \vdash_{\lambda^{\neg\wedge\exists}} P : A$ implies $\Gamma^{\mathsf{c}} \vdash_{\mathsf{cps}} P : A^{\mathsf{c}}$.*

Proof. By induction on P. Note that any type of P is a negation, since any CPS term is a λ-abstraction. So we will show that $\Gamma \vdash_{\lambda^{\neg\wedge\exists}} P : \neg \mathsf{A}$ implies $\Gamma^{\mathsf{c}} \vdash_{\mathsf{cps}} P : \neg \mathsf{A}^{\mathsf{d}}$.

Case $P \equiv \lambda k.Q\langle R, k\rangle$. Any derivation of $\Gamma \vdash_{\lambda^{\neg\wedge\exists}} P : \neg \mathsf{A}$ has the following form.

$$\dfrac{\dfrac{\Gamma \vdash Q : \neg(\mathsf{B} \wedge \mathsf{A}) \quad \dfrac{\Gamma \vdash R : \mathsf{B} \quad \overline{k : \mathsf{A} \vdash k : \mathsf{A}}}{\Gamma, k : \mathsf{A} \vdash \langle R, k\rangle : \mathsf{B} \wedge \mathsf{A}}}{\Gamma, k : \mathsf{A} \vdash Q\langle R, k\rangle : \bot}}{\Gamma \vdash \lambda k.Q\langle R, k\rangle : \neg \mathsf{A}}$$

By the induction hypotheses, we have $\Gamma^{\mathsf{c}} \vdash_{\mathsf{cps}} Q : \neg(\mathsf{B}^{\mathsf{c}} \wedge \mathsf{A}^{\mathsf{d}})$ and $\Gamma^{\mathsf{c}} \vdash_{\mathsf{cps}} R : \mathsf{B}^{\mathsf{c}}$, so we have $\Gamma^{\mathsf{c}} \vdash_{\mathsf{cps}} P : \neg \mathsf{A}^{\mathsf{d}}$.

Case $P \equiv \lambda k.Q\langle \mathcal{B}, k\rangle$. Any derivation of $\Gamma \vdash_{\lambda^{\neg\wedge\exists}} P : \neg \mathsf{A}$ has the following form, where A must be $\mathsf{C}[X := \mathcal{B}]$.

$$\dfrac{\dfrac{\Gamma \vdash Q : \neg\exists X.\mathsf{C} \quad \dfrac{\overline{k : \mathsf{A} \vdash k : \mathsf{A}}}{k : \mathsf{A} \vdash \langle \mathcal{B}, k\rangle : \exists X.\mathsf{C}}}{\Gamma, k : \mathsf{A} \vdash Q\langle \mathcal{B}, k\rangle : \bot}}{\Gamma \vdash \lambda k.Q\langle \mathcal{B}, k\rangle : \neg \mathsf{A}}$$

By the induction hypothesis, $\Gamma^{\mathsf{c}} \vdash_{\mathsf{cps}} Q : \neg\exists X.\mathsf{C}^{\mathsf{d}}$ holds, so we have $\Gamma^{\mathsf{c}} \vdash_{\mathsf{cps}} P : \neg \mathsf{C}^{\mathsf{d}}[X := \mathcal{B}]$ by letting $k : \mathsf{C}^{\mathsf{d}}[X := \mathcal{B}]$, where $\mathsf{C}^{\mathsf{d}}[X := \mathcal{B}]$ is identical to $(\mathsf{C}[X := \mathcal{B}])^{\mathsf{d}}$ by Lemma 2 (2).

Other cases are similarly proved. □

4.4 Proof of Undecidability

By the main lemma, we can reduce TC and TI of DF-$\lambda 2$ to those of DF-$\lambda^{\neg\wedge\exists}$, and then conclude undecidability of TC and TI in DF-$\lambda^{\neg\wedge\exists}$.

Proposition 3. *(1) $\Gamma \vdash_{\lambda 2} M : A$ holds if and only if $\neg\Gamma^\bullet \vdash_{\lambda^{\neg\wedge\exists}} [\![M]\!] : \neg A^\bullet$ holds.*

(2) For any DF-$\lambda 2$*-term M, $\Gamma \vdash_{\lambda 2} M : A$ holds for some Γ and A if and only if $\Gamma' \vdash_{\lambda^{\neg\wedge\exists}} [\![M]\!] : \mathsf{A}'$ holds for some Γ' and A'.*

Proof. (1) The only-if part is Proposition 1, so we will show the if part. If $\neg\Gamma^\bullet \vdash_{\lambda^{\neg\wedge\exists}} [\![M]\!] : \neg A^\bullet$ holds, by Lemma 3, we have $(\neg\Gamma^\bullet)^{\mathsf{c}} \vdash_{\mathsf{cps}} [\![M]\!] : (\neg A^\bullet)^{\mathsf{c}}$, from which $\neg\Gamma^\bullet \vdash_{\mathsf{cps}} [\![M]\!] : \neg A^\bullet$ follows by Lemma 2 (1). By Proposition 2 (2), $\Gamma \vdash_{\lambda 2} M : A$ holds.

(2) The only-if part follows from the only-if part of (1). The if part follows from Lemma 3 and Proposition 2 (2). □

Proof of Theorem 1. Undecidability of TC and TI in DF-$\lambda^{\neg\wedge\exists}$ are proved by Proposition 3 and Theorem 2. □

4.5 TC and TI of DF-$\lambda_g^{\neg\wedge\exists}$ Are Undecidable

The discussion for DF-$\lambda^{\neg\wedge\exists}$ in the previous subsections can be applied to a variant DF-$\lambda_g^{\neg\wedge\exists}$ with general elimination rules by defining a suitable CPS translation from DF-$\lambda 2$ to DF-$\lambda_g^{\neg\wedge\exists}$.

Definition 7 (DF-$\lambda_g^{\neg\wedge\exists}$)**.** *The terms of* DF-$\lambda_g^{\neg\wedge\exists}$ *are defined by*

$M ::= \ x \mid \lambda x.M \mid \langle M, M\rangle \mid \langle \mathsf{A}, M\rangle \mid MM \mid M[xx.M] \mid M[Xx.M]$. *The typing rules of* DF-$\lambda_g^{\neg\wedge\exists}$ *are the same as* DF-$\lambda^{\neg\wedge\exists}$ *except for replacing* ($\wedge$E1) *and* ($\wedge$E2) *by the following rule.*

$$\frac{\Gamma_1 \vdash M : \mathsf{A}\wedge\mathsf{B} \quad \Gamma_2, x:\mathsf{A}, y:\mathsf{B} \vdash N : \mathsf{C}}{\Gamma_1, \Gamma_2 \vdash M[xy.N] : \mathsf{C}}\ (\wedge\mathsf{E})$$

$\Gamma \vdash_{\lambda_g^{\neg\wedge\exists}} M : \mathsf{A}$ *is defined similarly to that in* DF-$\lambda^{\neg\wedge\exists}$.

Definition 8. *The CPS translation $[\![\cdot]\!]$ of* DF-$\lambda_g^{\neg\wedge\exists}$ *and its inverse $(\cdot)^\#$ are the same as those of* DF-$\lambda^{\neg\wedge\exists}$ *except for the cases of λ-abstractions, which are defined by $[\![\lambda x.M]\!] \equiv \lambda k.k[xk'.[\![M]\!]k']$, and $(\lambda k.k[xk'.Pk'])^\# \equiv \lambda x.P^\#$, where the definition of CPS terms is also changed by*

$P ::= \lambda k.xk \mid \lambda k.k[xk'.Pk'] \mid \lambda k.k[Xk'.Pk'] \mid \lambda k.P\langle P, k\rangle \mid \lambda k.P\langle \mathcal{A}, k\rangle$.

$\neg\Gamma \vdash_{\mathsf{g\text{-}cps}} P : \neg\mathcal{A}$ *is defined similarly to that in* DF-$\lambda^{\neg\wedge\exists}$.

Lemma 4 (Main Lemma). *If P is a CPS term, $\Gamma \vdash_{\lambda_g^{\neg\wedge\exists}} P : \mathsf{A}$ implies $\Gamma^{\mathsf{c}} \vdash_{\mathsf{g\text{-}cps}} P : \mathsf{A}^{\mathsf{c}}$.*

Proposition 4. *(1) $\Gamma \vdash_{\lambda 2} M : A$ holds if and only if $\neg\Gamma^\bullet \vdash_{\lambda_g^{\neg\wedge\exists}} [\![M]\!] : \neg A^\bullet$ holds.*

(2) For any DF-λ2*-term M, $\Gamma \vdash_{\lambda 2} M : A$ holds for some Γ and A if and only if $\Gamma' \vdash_{\lambda_g^{\neg\wedge\exists}} [\![M]\!] : \mathsf{A}'$ holds for some Γ' and A'.*

Theorem 3. *Type-checking and type-inference of* DF-$\lambda_g^{\neg\wedge\exists}$ *are undecidable.*

Proof. By Proposition 4 and Theorem 2. □

5 A Target of CPS Translations

In this section, we discuss that DF-$\lambda^{\neg\wedge\exists}$ is an essence of a target of CPS translations by showing it works well as a CPS target for the call-by-value computational λ-calculus, the call-by-value $\lambda\mu$-calculus, and delimited continuations. At first sight, $\lambda^{\neg\wedge\exists}$ may look weak as a computational system, but it suffices as a target calculus of several CPS translations [4,5]. Moreover, the domain-free style calculus with existence works also as a CPS target of the domain-free call-by-value $\lambda\mu$-calculus $\lambda_V\mu$ [3].

First, we define the reduction relation in DF-$\lambda^{\neg\wedge\exists}$. We omit η-rules, but the results in this section can be extended straightforwardly to η-rules.

Definition 9. *The reduction rules of* DF-$\lambda^{\neg\wedge\exists}$ *are the following.*

$(\beta_\to)$ $\quad (\lambda x.M)N \to M[x := N]$
$(\beta_\wedge)$ $\quad \langle M_1, M_2\rangle\pi_i \to M_i \qquad (i = 1 \text{ or } 2)$
$(\beta_\exists)$ $\quad \langle \mathsf{A}, M\rangle[Xx.N] \to N[X := \mathsf{A}, x := M]$

The relation $\to_{\lambda^{\neg\wedge\exists}}$ is the compatible closure of the above rules, and the relation $\to^*_{\lambda^{\neg\wedge\exists}}$ is its reflexive transitive closure.

5.1 Call-by-Value Second-Order Computational λ-Calculus

In [11], Sabry and Wadler gave a call-by-value CPS translation from the computational λ-calculus λ_c [8] to a CPS calculus λ_{cps}, which is a subsystem of the ordinary λ-calculus. Furthermore, they gave an inverse translation from λ_{cps} to λ_c, and showed that those translations form a reflection of λ_{cps} in λ_c.

DF-$\lambda^{\neg\wedge\exists}$ can be a target of a CPS translation for λ_c with polymorphic types. In this subsection, we define DF-$\lambda^{\neg\wedge\exists}_{\mathsf{cps/v}}$ as a subsystem of DF-$\lambda^{\neg\wedge\exists}$, and show that we have a reflection of DF-$\lambda^{\neg\wedge\exists}_{\mathsf{cps/v}}$ in λ_c with polymorphic types.

Definition 10 (DF-$\lambda_c^\forall$). *The system* DF-$\lambda_c^\forall$ *is an extension of* DF-λ2 *by adding* `let`*-expressions with the typing rule for them as follows.*

$$\frac{\Gamma_1 \vdash M : A \quad \Gamma_2, x : A \vdash N : B}{\Gamma_1, \Gamma_2 \vdash \texttt{let } x = M \texttt{ in } N : B}\ (\mathsf{let})$$

The values are defined by $V ::= x \mid \lambda x.M \mid \lambda X.M$. We use P, Q,... to denote terms that are not values. The call-by-value reduction is defined by the following rules.

$(\beta.v)$ $(\lambda x.M)V \to M[x := V]$
$(\beta.t)$ $(\lambda X.M)A \to M[X := A]$
$(\beta.\mathtt{let})$ $\mathtt{let}\ x = V\ \mathtt{in}\ M \to M[x := V]$
(ass) $\mathtt{let}\ y = (\mathtt{let}\ x = L\ \mathtt{in}\ M)\ \mathtt{in}\ N \to \mathtt{let}\ x = L\ \mathtt{in}\ (\mathtt{let}\ y = M\ \mathtt{in}\ N)$
$(\mathtt{let}.1)$ $PM \to \mathtt{let}\ x = P\ \mathtt{in}\ xM$
$(\mathtt{let}.2)$ $VP \to \mathtt{let}\ x = P\ \mathtt{in}\ Vx$
$(\mathtt{let}.3)$ $PA \to \mathtt{let}\ x = P\ \mathtt{in}\ xA$

In (ass), N *must not contain* x *freely.*

Definition 11 (DF-$\lambda^{\neg\wedge\exists}_{\mathsf{cps/v}}$)**.** *(1) Let* k *be a fixed term variable. The value types (denoted by* $\mathcal{A}$, $\mathcal{B}$,*...), the terms (denoted by* M, N,*...), the values (denoted by* V, W,*...), and the continuations (denoted by* K,*...) of* DF-$\lambda^{\neg\wedge\exists}_{\mathsf{cps/v}}$ *are defined by*

$$\begin{aligned}
\mathcal{A} &::= X \mid \neg(\mathcal{A} \wedge \neg\mathcal{A}) \mid \neg\exists X.\neg\mathcal{A},\\
M &::= KV \mid V\langle V, K\rangle \mid V\langle \mathcal{A}, K\rangle,\\
V &::= x \mid \lambda c.(\lambda x.(\lambda k.M)(c\pi_2))(c\pi_1) \mid \lambda c.c[Xk.M],\\
K &::= k \mid \lambda x.M,
\end{aligned}$$

where c *is a fresh variable, and occurrences of* c *denote those of the same variable. We write* $\lambda\langle x, k\rangle.M$ *for* $\lambda c.(\lambda x.(\lambda k.M)(c\pi_2))(c\pi_1)$, *and* $\lambda\langle X, k\rangle.M$ *for* $\lambda c.c[Xk.M]$.

(2) The reduction rules of DF-$\lambda^{\neg\wedge\exists}_{\mathsf{cps/v}}$ *are the following.*

$(\beta.v)$ $(\lambda\langle x, k\rangle.M)\langle V, K\rangle \to M[x := V, k := K]$
$(\beta.t)$ $(\lambda\langle X, k\rangle.M)\langle \mathcal{A}, K\rangle \to M[x := \mathcal{A}, k := K]$
$(\beta.\mathtt{let})$ $(\lambda x.M)V \to M[x := V]$

DF-$\lambda^{\neg\wedge\exists}_{\mathsf{cps/v}}$ is a subsystem of DF-$\lambda^{\neg\wedge\exists}$, and closed under the reduction. The first-order fragment of DF-$\lambda^{\neg\wedge\exists}_{\mathsf{cps/v}}$ is isomorphic to λ_{cps} in [11].

Definition 12. *(1) The negative translation* $(\cdot)^{\triangle}$ *from* $\to\forall$*-types to value types and its inverse* $(\cdot)^{\triangledown}$ *are defined by*

$$\begin{aligned}
&X^{\triangle} \equiv X, && X^{\triangledown} \equiv X,\\
&(A{\to}B)^{\triangle} \equiv \neg(A^{\triangle} \wedge \neg B^{\triangle}), && (\neg(\mathcal{A} \wedge \neg\mathcal{B}))^{\triangledown} \equiv \mathcal{A}^{\triangledown}{\to}\mathcal{B}^{\triangledown},\\
&(\forall X.A)^{\triangle} \equiv \neg\exists X.\neg A^{\triangle}, && (\neg\exists X.\neg\mathcal{A})^{\triangledown} \equiv \forall X.\mathcal{A}^{\triangledown}.
\end{aligned}$$

(2) The CPS translation $[\![\cdot]\!]$ *from* DF-$\lambda^{\forall}_{c}$ *to* DF-$\lambda^{\neg\wedge\exists}_{\mathsf{cps/v}}$ *is defined by*

$$\begin{aligned}
&[\![M]\!] \equiv M : k, && V : K \equiv K\Phi(V),\\
& && VW : K \equiv \Phi(V)\langle\Phi(W), K\rangle,\\
&\Phi(x) \equiv x, && PW : K \equiv P : \lambda m.m\langle\Phi(W), K\rangle,\\
&\Phi(\lambda x.M) \equiv \lambda\langle x, k\rangle.[\![M]\!], && VQ : K \equiv Q : \lambda n.\Phi(V)\langle n, K\rangle,\\
&\Phi(\lambda X.M) \equiv \lambda\langle X, k\rangle.[\![M]\!], && PQ : K \equiv P : \lambda m.(Q : \lambda n.m\langle n, K\rangle),\\
& && VA : K \equiv \Phi(V)\langle A^{\triangle}, K\rangle,\\
& && PA : K \equiv P : \lambda m.m\langle A^{\triangle}, K\rangle,\\
& && \mathtt{let}\ x = M\ \mathtt{in}\ N : K \equiv M : \lambda x.(N : K),
\end{aligned}$$

where m *and* n *are fresh variables.*

(3) The inverse translation $(\cdot)^{\#}$ *from* $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}_{\mathsf{cps/v}}$ *to* $\mathsf{DF}\text{-}\lambda^{\forall}_{c}$ *is defined by*

$$
\begin{array}{ll}
(KV)^{\#} \equiv K^{\flat}[V^{\natural}], & x^{\natural} \equiv x, \\
(V\langle W, K\rangle)^{\#} \equiv K^{\flat}[V^{\natural}W^{\natural}], & (\lambda\langle x, k\rangle.M)^{\natural} \equiv \lambda x.M^{\#}, \\
(V\langle \mathcal{A}, K\rangle)^{\#} \equiv K^{\flat}[\mathcal{A}^{\triangledown}W^{\natural}], & (\lambda\langle X, k\rangle.M)^{\natural} \equiv \lambda X.M^{\#}, \\
k^{\flat} \equiv [], & \\
(\lambda x.M)^{\flat} \equiv \mathtt{let}\ x = [] \ \mathtt{in}\ M^{\#}. &
\end{array}
$$

Proposition 5. *(1)* $\Gamma \vdash_{\lambda^{\forall}_{c}} M : A$ *implies* $\Gamma^{\triangle}, k : \neg A^{\triangle} \vdash_{\lambda^{\neg\wedge\exists}} [\![M]\!] : \bot$.

(2) $[\![\cdot]\!]$ *and* $(\cdot)^{\#}$ *form a reflection of* $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}_{\mathsf{cps/v}}$ *in* $\mathsf{DF}\text{-}\lambda^{\forall}_{c}$, *that is, (a)* $[\![\cdot]\!]$ *and* $(\cdot)^{\#}$ *preserve reduction relation* $\rightarrow^{*}$, *(b)* $[\![M^{\#}]\!] \equiv M$ *holds for any term* M *of* $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}_{\mathsf{cps/v}}$, *and (c)* $M \rightarrow^{*} [\![M]\!]^{\#}$ *holds for any term* M *of* $\mathsf{DF}\text{-}\lambda^{\forall}_{c}$.

5.2 Call-by-Value $\lambda\mu$-Calculus

The $\lambda\mu$-calculus was introduced by Parigot in [9] as an extension of λ-calculus, and it corresponds to the classical natural deduction for second-order propositional logic by the Curry-Howard isomorphism. In [3], Fujita pointed out that the Curry-style call-by-value $\lambda\mu$-calculus does not enjoy the subject reduction property, so he introduced a domain-free call-by-value $\lambda\mu$-calculus $\lambda_{V}\mu$ to avoid the problem. In this subsection, we show that $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$ works as a target calculus of a CPS translation for $\lambda_{V}\mu$.

Definition 13 $(\lambda_{V}\mu)$**.** *(1) The system* $\lambda_{V}\mu$ *has a set of another sort of variables called* μ*-variables (denoted by* α, β,*...). The types of* $\lambda_{V}\mu$ *are the* $\rightarrow\forall$*-types. The terms (denoted by* M, N,*...), and the values (denoted by* V, W,*...) of* $\lambda_{V}\mu$ *are defined by*

$M ::= V \mid MM \mid MA \mid \mu\alpha.[\alpha]M$,

$V ::= x \mid \lambda x.M \mid \lambda X.M$.

(2) The typing rules of $\lambda_{V}\mu$ *are the following.*

$$
\frac{}{\Gamma, x : A \vdash x : A; \Delta}\ (\mathsf{Ax}) \qquad \frac{\Gamma \vdash M : B; \Delta}{\Gamma \vdash \mu\alpha.[\beta]M : A; (\Delta, \beta : B) - \{\alpha : A\}}\ (\mu)
$$

$$
\frac{\Gamma, x : A \vdash M : B; \Delta}{\Gamma \vdash \lambda x.M : A{\rightarrow}B; \Delta}\ (\rightarrow\mathsf{I}) \qquad \frac{\Gamma_1 \vdash M : A{\rightarrow}B; \Delta_1 \quad \Gamma_2 \vdash N : A; \Delta_2}{\Gamma_1, \Gamma_2 \vdash MN : B; \Delta_1, \Delta_2}\ (\rightarrow\mathsf{E})
$$

$$
\frac{\Gamma \vdash M : A; \Delta}{\Gamma \vdash \lambda X.M : \forall X.A; \Delta}\ (\forall\mathsf{I}) \qquad \frac{\Gamma \vdash M : \forall X.A; \Delta}{\Gamma \vdash MB : A[X := B]; \Delta}\ (\forall\mathsf{E})
$$

Γ denotes a context similarly to $\mathsf{DF}\text{-}\lambda^{\neg\wedge\exists}$. *$\Delta$ denotes a μ-context, which is a finite set of type assignments for μ-variables in the form of* $(\alpha : A)$. *In the rule* $(\forall\mathsf{I})$, *the lower sequent must not contain X freely.*

(3) The singular contexts are defined by $\mathcal{C} ::= []M \mid V[] \mid []A$. *The term* $\mathcal{C}[M]$ *is obtained from $\mathcal{C}$ by replacing $[]$ by M. The structural substitution* $M[\alpha{\Leftarrow}\mathcal{C}]$ *is obtained from M by replacing each subterm* $[\alpha]L$ *by* $[\alpha]\mathcal{C}[L[\alpha{\Leftarrow}\mathcal{C}]]$. *The reduction rules of* $\lambda_{V}\mu$ *are the following.*

(β_{tm}) $(\lambda x.M)N \to M[x := N]$ $\qquad$ (μ) $\mathcal{C}[\mu\alpha.M] \to \mu\alpha.M[\alpha \Leftarrow \mathcal{C}]$
(β_{tp}) $(\lambda X.M)A \to M[X := A]$

Definition 14. *The negative translation* $(\cdot)^{\triangle}$ *and the CPS translation* $[\![\cdot]\!]$ *from* $\lambda_V\mu$ *to* DF-$\lambda^{\neg\wedge\exists}$ *are the same as Definition 12, except for replacing the definition for* `let` *by* $\mu\alpha.[\beta]M : K \equiv (M : x_\beta)[x_\alpha := K]$*, where we suppose that* DF-$\lambda^{\neg\wedge\exists}$ *contains a term variable* x_α *for each* μ*-variable* α*.*

Proposition 6. *(1)* $\Gamma \vdash_{\lambda_V\mu} M : A; \Delta$ *implies* $\Gamma^{\triangle}, \neg\Delta^{\triangle}, k\colon \neg A^{\triangle} \vdash_{\lambda^{\neg\wedge\exists}} [\![M]\!]\colon \bot$*.*
(2) $M \to^*_{\lambda_V\mu} N$ *implies* $[\![M]\!] \to^*_{\lambda^{\neg\wedge\exists}} [\![N]\!]$*.*

5.3 Delimited Continuations

The $\neg \wedge \exists$-fragments are also useful as a target of a CPS translation for delimited continuations such as `shift` and `reset` [2]. For calculi with delimited continuations, we consider multi-staged CPS translations, and we need call-by-value calculi as intermediate CPS calculi. However, in order to have a sound CPS translation to an $\to\exists$-fragment, the calculus has to have not only the call-by-value η-reduction, but also the full η-reduction. On the other hand, as it was shown in [6], we can define a sound CPS translation from a calculus with `shift` and `reset` to a call-by-value $\neg \wedge \exists$-fragment without full η-reduction.

6 Undecidability in Implicational Fragment

Our method by means of CPS translations can be used for the domain-free typed λ-calculus DF-$\lambda^{\to\exists}$ with implication and existence. In this section, we define DF-$\lambda^{\to\exists}$ and a CPS translation from DF-$\lambda 2$ to DF-$\lambda^{\to\exists}$, by which TC and TI of DF-$\lambda 2$ are reduced to those of DF-$\lambda^{\to\exists}$.

Definition 15 (DF-$\lambda^{\to\exists}$). *The types (called* $\to\exists$*-types) and the terms of* DF-$\lambda^{\to\exists}$ *are defined by*
$\mathsf{A} ::= X \mid \bot \mid \mathsf{A}\to\mathsf{A} \mid \exists X.\mathsf{A}$,
$M ::= x \mid \lambda x.M \mid \langle \mathsf{A}, M\rangle \mid MM \mid M[Xx.M]$,
We write $\neg\mathsf{A}$ *for* $\mathsf{A}\to\bot$*. The typing rules of* DF-$\lambda^{\to\exists}$ *are* (Ax), $(\exists\mathsf{I})$, $(\exists\mathsf{E})$ *of* DF-$\lambda^{\neg\wedge\exists}$ *and*

$$\frac{\Gamma, x : \mathsf{A} \vdash M : \mathsf{B}}{\Gamma \vdash \lambda x.M : \mathsf{A}\to\mathsf{B}}\ (\to\mathsf{I}), \qquad \frac{\Gamma_1 \vdash M : \mathsf{A}\to\mathsf{B} \quad \Gamma_2 \vdash N : \mathsf{A}}{\Gamma_1, \Gamma_2 \vdash MN : \mathsf{B}}\ (\to\mathsf{E}).$$

Definition 16 (CPS translation). *(1) The negative translation* $(\cdot)^\bullet$ *from* $\to\forall$*-types to* $\to\exists$*-types and its inverse* $(\cdot)^\circ$ *from continuation types to* $\to\forall$*-types are defined by*

$X^\bullet \equiv X$, $\qquad$ $X^\circ \equiv X$,
$(A\to B)^\bullet \equiv \neg(\neg A^\bullet \to \neg B^\bullet)$, $\qquad$ $(\neg(\neg\mathcal{A}\to\neg\mathcal{B}))^\circ \equiv \mathcal{A}^\circ\to\mathcal{B}^\circ$,
$(\forall X.A)^\bullet \equiv \exists X.A^\bullet$, $\qquad$ $(\exists X.\mathcal{A})^\circ \equiv \forall X.\mathcal{A}^\circ$,

where the continuation types are defined by $\mathcal{A} ::= X \mid \neg(\neg\mathcal{A}\to\neg\mathcal{A}) \mid \exists X.\mathcal{A}$. *The CPS types are defined as types of the form* $\neg\mathcal{A}$.

(2) The CPS translation from terms in DF-$\lambda 2$ *to terms in* DF-$\lambda^{\to\exists}$ *and its inverse from CPS terms to terms of* DF-$\lambda 2$ *are defined by*

$$\begin{array}{ll}
[\![x]\!] \equiv \lambda k.xk, & (\lambda k.xk)^{\#} \equiv x,\\
[\![\lambda x.M]\!] \equiv \lambda k.k(\lambda x.[\![M]\!]), & (\lambda k.k(\lambda x.P))^{\#} \equiv \lambda x.P^{\#},\\
[\![\lambda X.M]\!] \equiv \lambda k.k[Xk'.[\![M]\!]k'], & (\lambda k.k[Xk'.Pk'])^{\#} \equiv \lambda X.P^{\#},\\
[\![MN]\!] \equiv \lambda k.[\![M]\!](\lambda m.m[\![N]\!]k), & (\lambda k.P(\lambda m.mQk))^{\#} \equiv P^{\#}Q^{\#},\\
[\![MA]\!] \equiv \lambda k.[\![M]\!]\langle A^{\bullet}, k\rangle, & (\lambda k.P\langle\mathcal{A}, k\rangle)^{\#} \equiv P^{\#}A^{\circ},
\end{array}$$

where the CPS terms are defined by

$P ::= \lambda k.xk \mid \lambda k.k(\lambda x.P) \mid \lambda k.k[Xk'.Pk'] \mid \lambda k.P(\lambda m.mPk) \mid \lambda k.P\langle\mathcal{A}, k\rangle$, *where occurrences of* k *and* k' *denote those of the same variable.*

(3) The system $\mathsf{DF}\text{-}\lambda^{\to\exists}_{\mathsf{cps}}$ *is defined as a subsystem of* DF-$\lambda^{\to\exists}$ *by restricting terms and types to CPS terms and CPS types, respectively. We write* $\neg\Gamma \vdash_{\to\exists\mathsf{cps}} P : \neg\mathcal{A}$ *to denote that the judgment is derivable in* $\mathsf{DF}\text{-}\lambda^{\to\exists}_{\mathsf{cps}}$.

Lemma 5. *(1) For any* $\to\forall$*-type* A, $A^{\bullet}$ *is a continuation type, and* $A^{\bullet\circ} \equiv A$ *holds.*

(2) For any DF-$\lambda 2$*-term* M, $[\![M]\!]$ *is a CPS term, and* $[\![M]\!]^{\#} \equiv M$ *holds.*

Proposition 7. *(1)* $\Gamma \vdash_{\lambda 2} M : A$ *implies* $\neg\Gamma^{\bullet} \vdash_{\lambda^{\to\exists}} [\![M]\!] : \neg A^{\bullet}$.

(2) $\neg\Gamma \vdash_{\to\exists\mathsf{cps}} P : \neg\mathcal{A}$ *implies* $\Gamma^{\circ} \vdash_{\lambda 2} P^{\#} : \mathcal{A}^{\circ}$.

(3) $\neg\Gamma^{\bullet} \vdash_{\to\exists\mathsf{cps}} [\![M]\!] : \neg A^{\bullet}$ *implies* $\Gamma \vdash_{\lambda 2} M : A$.

Definition 17 (Contraction Translation). *Let* $\mathcal{S}$ *be a fixed closed continuation type. The contraction translation* $(\cdot)^{\mathsf{c}}$ *from* $\to\exists$*-types to CPS types is defined by*

$(\mathsf{A}\to\mathsf{B})^{\mathsf{c}} \equiv \neg\mathsf{A}^{\mathsf{d}}$,
$\mathsf{A}^{\mathsf{c}} \equiv \neg\mathcal{S}$ *(*A *is not an implication),*
$X^{\mathsf{d}} \equiv X$,
$\bot^{\mathsf{d}} \equiv \mathcal{S}$,
$((\mathsf{A}\to\mathsf{B}\to\mathsf{C})\to\mathsf{D})^{\mathsf{d}} \equiv \neg(\mathsf{A}^{\mathsf{c}}\to\neg\mathsf{B}^{\mathsf{d}})$,
$((\mathsf{A}\to\mathsf{B})\to\mathsf{D})^{\mathsf{d}} \equiv \neg(\mathsf{A}^{\mathsf{c}}\to\neg\mathcal{S})$, *(*$\mathsf{B}$ *is neither an implication nor* $\bot$*),*
$(\mathsf{A}\to\mathsf{D})^{\mathsf{d}} \equiv \mathcal{S}$ *(otherwise),*
$(\exists X.\mathsf{A})^{\mathsf{d}} \equiv \exists X.\mathsf{A}^{\mathsf{d}}$.

Lemma 6. *(1) For any continuation type* $\mathcal{A}$, $(\neg\mathcal{A})^{\mathsf{c}} \equiv \neg\mathcal{A}$ *and* $\mathcal{A}^{\mathsf{d}} \equiv \mathcal{A}$ *hold.*

(2) For any continuation type $\mathcal{A}$ *and any* $\to\exists$*-type* B, $(\mathsf{B}[X := \mathcal{A}])^{\mathsf{c}} \equiv \mathsf{B}^{\mathsf{c}}[X := \mathcal{A}]$ *and* $(\mathsf{B}[X := \mathcal{A}])^{\mathsf{d}} \equiv \mathsf{B}^{\mathsf{d}}[X := \mathcal{A}]$ *hold.*

Proof. (1) is straightforwardly proved by induction. For (2), we use the fact $\mathcal{A}^{\mathsf{c}} \equiv \mathcal{S}$ and $(\mathcal{A} \to \mathsf{B})^{\mathsf{d}} \equiv \mathcal{S}$. □

Lemma 7 (Main Lemma). *If* P *is a CPS term,* $\Gamma \vdash_{\lambda^{\to\exists}} P : \mathsf{A}$ *implies* $\Gamma^{\mathsf{c}} \vdash_{\to\exists\mathsf{cps}} P : \mathsf{A}^{\mathsf{c}}$.

Proof. By induction on P. Note that any type of a CPS term is an implication, so we will show that $\Gamma \vdash_{\lambda^{\to\exists}} P : \mathsf{A}_1 \to \mathsf{A}_2$ implies $\Gamma^{\mathsf{c}} \vdash_{\to\exists\mathsf{cps}} P : \neg\mathsf{A}_1^{\mathsf{d}}$. We will show only non-trivial cases, and other cases are proved similarly to DF-$\lambda^{\neg\wedge\exists}$.

Case $P \equiv \lambda k.k(\lambda x.Q)$. Any derivation of $\Gamma \vdash_{\lambda^{\to\exists}} P : \mathsf{A}_1 \to \mathsf{A}_2$ has the following form, where A_1 must be $(\mathsf{B}_1 \to \mathsf{B}_2) \to \mathsf{A}_2$.

$$\dfrac{\dfrac{}{k : \mathsf{A}_1 \vdash k : \mathsf{A}_1} \quad \dfrac{\Gamma, x : \mathsf{B}_1 \vdash Q : \mathsf{B}_2}{\Gamma \vdash \lambda x.Q : \mathsf{B}_1 \to \mathsf{B}_2}}{\dfrac{\Gamma, k : \mathsf{A}_1 \vdash k(\lambda x.Q) : \mathsf{A}_2}{\Gamma \vdash \lambda k.k(\lambda x.Q) : \mathsf{A}_1 \to \mathsf{A}_2}}$$

Note that B_2 is an implication since it is a type of a CPS term Q, so we have $\mathsf{A}_1^{\mathsf{d}} \equiv ((\mathsf{B}_1 \to \mathsf{B}_2) \to \mathsf{A}_2)^{\mathsf{d}} \equiv \neg(\mathsf{B}_1^{\mathsf{c}} \to \mathsf{B}_2^{\mathsf{c}})$ by Definition 17. By the induction hypothesis, we have $\Gamma^{\mathsf{c}}, x : \mathsf{B}_1^{\mathsf{c}} \vdash_{\to\exists\mathsf{cps}} Q : \mathsf{B}_2^{\mathsf{c}}$, so $\Gamma^{\mathsf{c}} \vdash_{\to\exists\mathsf{cps}} P : \neg\neg(\mathsf{B}_1^{\mathsf{c}} \to \mathsf{B}_2^{\mathsf{c}})$.

Case $P \equiv \lambda k.Q(\lambda m.mRk)$. Any derivation of $\Gamma \vdash_{\lambda^{\to\exists}} P : \mathsf{A}_1 \to \mathsf{A}_2$ has the following form, where B must be $(\mathsf{C} \to \mathsf{A}_1 \to \mathsf{D}) \to \mathsf{D}$.

$$\dfrac{\Gamma \vdash Q : \mathsf{B} \to \mathsf{A}_2 \qquad \dfrac{\dfrac{\dfrac{\dfrac{m : \mathsf{C} \to \mathsf{A}_1 \to \mathsf{D} \vdash m : \mathsf{C} \to \mathsf{A}_1 \to \mathsf{D}} \quad \Gamma \vdash R : C}{\Gamma, m : \mathsf{C} \to \mathsf{A}_1 \to \mathsf{D} \vdash mR : \mathsf{A}_1 \to \mathsf{D}} \quad \dfrac{}{k : \mathsf{A}_1 \vdash k : \mathsf{A}_1}}{\Gamma, k : \mathsf{A}_1, m : \mathsf{C} \to \mathsf{A}_1 \to \mathsf{D} \vdash mRk : \mathsf{D}}}{\Gamma, k : \mathsf{A}_1 \vdash \lambda m.mRk : (\mathsf{C} \to \mathsf{A}_1 \to \mathsf{D}) \to \mathsf{D}}}{\dfrac{\Gamma, k : \mathsf{A}_1 \vdash Q(\lambda m.mRk) : \mathsf{A}_2}{\Gamma \vdash \lambda k.Q(\lambda m.mRk) : \mathsf{A}_1 \to \mathsf{A}_2}}$$

By the induction hypotheses, we have $\Gamma^{\mathsf{c}} \vdash_{\to\exists\mathsf{cps}} Q : \neg\mathsf{B}^{\mathsf{d}}$ and $\Gamma^{\mathsf{c}} \vdash_{\to\exists\mathsf{cps}} R : \mathsf{C}^{\mathsf{c}}$, where B^{d} is identical to $\neg(\mathsf{C}^{\mathsf{c}} \to \neg\mathsf{A}_1^{\mathsf{d}})$. So we have $\Gamma^{\mathsf{c}} \vdash_{\to\exists\mathsf{cps}} P : \neg\mathsf{A}_1^{\mathsf{d}}$ by letting $k : \mathsf{A}_1^{\mathsf{d}}$ and $m : \mathsf{C}^{\mathsf{c}} \to \neg\mathsf{A}_1^{\mathsf{d}}$. □

Proposition 8. *(1) $\Gamma \vdash_{\lambda 2} M : A$ holds if and only if $\neg\Gamma^{\bullet} \vdash_{\lambda^{\to\exists}} [\![M]\!] : \neg A^{\bullet}$ holds.*

(2) For any DF-$\lambda 2$*-term M, $\Gamma \vdash_{\lambda 2} M : A$ holds for some Γ and A if and only if $\Gamma' \vdash_{\lambda^{\to\exists}} [\![M]\!] : \mathsf{A}'$ holds for some Γ' and A'.*

Theorem 4. *Type-checking and type-inference of* DF-$\lambda^{\to\exists}$ *are undecidable.*

7 Concluding Remarks

We can consider the Curry-style system with negation, conjunction, and existence, where the inference rules for $\exists$ are

$$\dfrac{\Gamma \vdash N : \mathsf{A}[X := \mathsf{B}]}{\Gamma \vdash \langle \exists, N \rangle : \exists X.\mathsf{A}}\ (\exists\mathsf{I}), \qquad \dfrac{\Gamma_1 \vdash M : \exists X.\mathsf{A} \quad \Gamma_2, x : \mathsf{A} \vdash N : \mathsf{C}}{\Gamma_1, \Gamma_2 \vdash M[x.N] : \mathsf{C}}\ (\exists\mathsf{E}),$$

where terms do not contain any type information [12]. We could not directly apply our approach to this system. Proving undecidability of TC and TI in this system would be future work.

Acknowledgments. The authors would like to thank Professor Ken-etsu Fujita for helpful comments, and Professor Masahito Hasegawa for a copy of his draft [6]. The first author was partially supported by the Japanese Ministry of Education, Culture, Sports, Science and Technology, Grant-in-Aid for Young Scientists (B) 18700008.

References

1. Barthe, G., Sørensen, M.H.: Domain-free pure type systems. J. Functional Programming 10, 412–452 (2000)
2. Danvy, O., Fillinski, A.: Representing Control: a Study of the CPS Translation. Mathematical Structures in Computer Science 2(4), 361–391 (1992)
3. Fujita, K.: Explicitly typed $\lambda\mu$-calculus for polymorphism and call-by-value. In: Girard, J.-Y. (ed.) TLCA 1999. LNCS, vol. 1581, pp. 162–177. Springer, Heidelberg (1999)
4. Fujita, K.: Galois embedding from polymorphic types in to existential types. In: Urzyczyn, P. (ed.) TLCA 2005. LNCS, vol. 3461, pp. 194–208. Springer, Heidelberg (2005)
5. Hasegawa, M.: Relational parametricity and control. Logical Methods in Computer Science 2(3:3), 1–22 (2006)
6. Hasegawa, M.: (unpublished manuscript, 2007)
7. Mitchell, J.C., Plotkin, G.D.: Abstract types have existential type. ACM Transactions on Programming Languages and Systems 10(3), 470–502 (1988)
8. Moggi, E.: Computational lambda-calculus and monads. In: Proceedings of 4th Annual Symposium on Logic in Computer Science (LICS 1989), pp. 14–23 (1989)
9. Parigot, M.: $\lambda\mu$-calculus: an algorithmic interpretation of classical natural deduction. In: Voronkov, A. (ed.) LPAR 1992. LNCS, vol. 624, pp. 190–201. Springer, Heidelberg (1992)
10. Plotkin, G.: Call-by-name, call-by-value, and the λ-calculus. Theoretical Computer Science 1, 125–159 (1975)
11. Sabry, A., Wadler, P.: A reflection on call-by-value. ACM Transactions on Programming Languages and Systems 19(6), 916–941 (1997)
12. Tatsuta, M.: Simple saturated sets for disjunction and second-order existential quantification. In: Della Rocca, S.R. (ed.) TLCA 2007. LNCS, vol. 4583, pp. 366–380. Springer, Heidelberg (2007)
13. Tatsuta, M., Fujita, K., Hasegawa, R., Nakano, H.: Inhabitance of Existential Types is Decidable in Negation-Product Fragment. In: Proceedings of 2nd International Workshop on Classical Logic and Computation (CLC 2008) (2008)
14. Thielecke, H.: Categorical Structure of Continuation Passing Style. Ph.D. Thesis, University of Edinburgh (1997)
15. van Benthem Jutting, L.S.: Typing in pure type systems. Information and Computation 105, 30–41 (1993)
16. Wells, J.B.: Typability and type checking in the second-order λ-calculus are equivalent and undecidable. In: Proceedings of 9th Symposium on Logic in Computer Science (LICS 1994), pp. 176–185 (1994)