Type Checking and Inference for Polymorphic and Existential Types in Multiple-Quantifier and Type-Free Systems

Koji Nakazawa* Makoto Tatsuta†

Abstract

A multiple quantifier is a quantifier having inference rules that introduce or eliminate arbitrary number of quantifiers by one inference. This paper introduces the lambda calculus with negation, conjunction, and multiple existential quantifiers, and the lambda calculus with implication and multiple universal quantifiers. Their type checking and type inference are proved to be undecidable. This paper also shows that the undecidability of type checking and type inference in the type-free-style lambda calculus with negation, conjunction, and existence is reduced to the undecidability of type checking and type inference in the type-free-style polymorphic lambda calculus.

Keywords. type checking, type inference, polymorphic type, existential type

1 Introduction

The second-order universal quantifiers and the second-order existential quantifiers are important from the point of view of both computer science and logic. Girard (1972) and Reynolds (1974) have independently established the typed lambda calculus with polymorphic types, which correspond to the second-order universal quantifiers in logical systems. Since their seminal papers, many papers have been devoted to investigation on the polymorphic types. The computational meaning of the second-order existential quantifiers has also been actively studied since the work of Mitchell & Plotkin (1988) on the abstract data types. More recently, Fujita (2005) and Hasegawa (2006) pointed out that calculi with negation, conjunction, and existence are suitable for target calculi of the *continuation-passing-style* (CPS) translations of polymorphic typed calculi.

Type checking and type inference are important for type assignment systems. *Type checking*, which we will write TC for, is the problem that asks whether a given typing judgment is derivable. *Type inference*, which we will write TI for, asks whether a given term has some type under some context. *Strong type inference*, which we will write STI for, asks whether, for a given term and a given context, the term has some type under some extension of the context. STI is a generalization of TI. We will write them as follows: TC asks $\Gamma \vdash M : A?$ for given Γ, M and A. TI asks $? \vdash M :?$ for given M. STI asks $\Gamma, ? \vdash M :?$ for given Γ and M. Since TI is an instance of STI, the undecidability of TI implies the undecidability of STI.

TC and TI in lambda calculi with polymorphic types have been actively studied. Wells (1999) proved that all these problems are undecidable in the Curry-style polymorphic lambda calculus. It was surprising that Schubert (1998) showed that TI is undecidable even in the Church style, where

*Graduate School of Informatics, Kyoto University, Kyoto 606-8501, Japan (E-mail:knak@kuis.kyoto-u.ac.jp)
†National Institute of Informatics, 2-1-2 Hitotsubashi, Tokyo 101-8430, Japan (E-mail:tatsuta@nii.ac.jp)

we consider typability of untyped pseudo terms. Barthe & Sørensen (2000) and Fujita & Schubert (2000) independently proved that TC and STI are undecidable in the domain-free polymorphic lambda calculus. In the *domain-free style*, a term contains information of applications of quantifier rules, for example, $\Lambda X.M$ for the $\forall$-introduction rule and MA for the $\forall$-elimination rule, but the domain type of a lambda abstraction is not indicated, for example, $\lambda x.M$.

On the other hand, properties of lambda calculi with existential types have not been enough studied yet. The inhabitation problem in the negation, conjunction, and existence fragment was recently proved to be decidable in Tatsuta et al. (2008). The *inhabitation*, which we will write INH for, is the problem that asks $\vdash ? : A$ for a given type A. It was also recently that TC and STI in its domain-free-style variant was proved to be undecidable in Nakazawa et al. (2008).

A multiple-quantifier introduction rule introduces several quantifiers at one step. For example, the system M-F has the rule that derives $\forall X_1 \forall X_2 \forall X_3 A$ from A at one step. The meaning of quantifiers is the same as that of usual quantifiers, and only inference rules and corresponding terms are different. In a similar way, a multiple-quantifier elimination rule eliminates several quantifiers at one step. We will call a quantifier a multiple quantifier when the system has its multiple-quantifier rules.

We will define the multiple-quantifier lambda calculus M-$\lambda^{\neg\wedge\exists}$ with negation, conjunction, and existence. The system has the following inference rules for the multiple existential quantifier:

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

where $\overline{X}$ denotes a finite sequence of type variables. We will also define the multiple-quantifier lambda calculus M-F with polymorphic types. The system has the following inference rules for the multiple universal quantifier:

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash \Lambda^*.M : \forall \overline{X}.A}\ (\forall \mathsf{I}) \qquad \frac{\Gamma \vdash M : \forall \overline{X}.A}{\Gamma \vdash M\bullet^* : A[\overline{X} := \overline{B}]}\ (\forall \mathsf{E})$$

We will discuss the type-free-style lambda calculus TF-$\lambda^{\neg\wedge\exists}$ with negation, conjunction, and existence. The system has the following inference rules for the existential types:

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

The type-free-style lambda calculus TF-F will be defined to have similar inference rules for polymorphic types. They are discussed in Tatsuta (2007), Tatsuta et al. (2008), and Fujita & Schubert (2009). The name "type-free" was used in Fujita & Schubert (2009). It means that terms contain no explicit type annotations. For the notation of terms for $(\exists \mathsf{E})$, we will follow Tatsuta (2007).

We will prove the following two claims: (1) TC and TI are undecidable in both M-F and M-$\lambda^{\neg\wedge\exists}$. (2) The undecidability of TC and TI in TF-$\lambda^{\neg\wedge\exists}$ is reduced to the undecidability of TC and TI in TF-F.

The systems M-F and M-$\lambda^{\neg\wedge\exists}$ are in an intermediate style between the Curry style and the Church style. The terms in M-F or M-$\lambda^{\neg\wedge\exists}$ have information of multiple quantifiers according to its multiple-quantifier rules. The sequence may be empty, and the annotations in terms just express possibility of use of quantifier rules.

The system M-$\lambda^{\neg\wedge\exists}$ is one of the most implicitly typed systems for existential types since terms of M-$\lambda^{\neg\wedge\exists}$ have less type information than other sound type systems for existential types. We may

F	TC	STI	TI	INH	$\lambda^{\neg\wedge\exists}$	TC	STI	TI	INH
Curry-	no$^{(1)}$	no	no$^{(1)}$	no					
M-	NO	NO	NO		M-	NO	NO	NO	yes$^{(5)}$
TF-	?	?	?		TF-	?	?	?	
DF-	no$^{(2)}$	no$^{(2)}$	no$^{(3)}$		DF-	no$^{(4)}$	no	no$^{(3)}$	

Figure 1: Decidability of TC, STI, TI and INH

consider a type assignment system for the pure lambda terms by the following rules for existential types:

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

where $N[x := M]$ is the ordinary substitution. This system has a more implicitly typed style than M-$\lambda^{\neg\wedge\exists}$, but it is not sound. Fujita & Schubert (2009) presented a counterexample for the subject reduction property in this system. Tatsuta (2007) also gave a counterexample for the subject reduction property in the second-order natural deduction with the above rules. On the other hand, we can define a sound reduction system for M-$\lambda^{\neg\wedge\exists}$. Another significance of M-$\lambda^{\neg\wedge\exists}$ is that the multiple existential rules can handle mutual abstract data types even without parameters.

In the polymorphic type system M-F, a term has more type information than a pure lambda term. We can define an embedding $\lfloor M \rfloor$ from the Curry-style F into M-F such that M has the type A in the Curry-style F if and only if $\lfloor M \rfloor$ has the type A in M-F. By this embedding, M-F can explicitly capture the behavior of the polymorphic lambda calculus discussed by Wells (1999).

This paper will discuss M-$\lambda^{\neg\wedge\exists}$ in two ways. Our first proof shows that TC is undecidable in M-$\lambda^{\neg\wedge\exists}$ by a direct reduction of the semi-unification problem to TC of M-$\lambda^{\neg\wedge\exists}$. The semi-unification problem has been proved to be undecidable in Kfoury et al. (1993). Our second proof shows the undecidability of TC and TI in M-$\lambda^{\neg\wedge\exists}$, by first proving that TC and TI are undecidable in M-F by the embedding from the Curry-style F to M-F, and then proving that TC and TI in M-F is reduced to TC and TI in M-$\lambda^{\neg\wedge\exists}$ by the method in Nakazawa et al. (2008) with some modification.

This paper will also show that the undecidability of TC and TI in TF-$\lambda^{\neg\wedge\exists}$ is reduced to the undecidability of TC and TI in TF-F. In order for this, we will adapt the method of Nakazawa et al. (2008) to TF-$\lambda^{\neg\wedge\exists}$. Nakazawa et al. (2008) showed that TC and TI in the domain-free F, denoted by DF-F, is reduced to TC and TI in the domain-free $\lambda^{\neg\wedge\exists}$, denoted by DF-$\lambda^{\neg\wedge\exists}$, by means of a CPS translation from DF-F to DF-$\lambda^{\neg\wedge\exists}$. By adapting this method to TF-$\lambda^{\neg\wedge\exists}$, we will give a CPS translation from TF-F to TF-$\lambda^{\neg\wedge\exists}$, which reduces TC and TI in TF-F to TC and TI in TF-$\lambda^{\neg\wedge\exists}$.

Figure 1 summarizes related results about the decidability of TC, STI, TI, and INH. In the figure, the problem marked by "no" means that the problem is undecidable, and the problems marked by "NO" denote the main results of this paper. The result (1) is proved by Wells (1999). The result (2) is proved by Barthe & Sørensen (2000) and Fujita & Schubert (2000) independently. The result (3) is proved by Nakazawa & Tatsuta (2009). The result (4) is proved by Nakazawa et al. (2008). The result (5) is proved by Tatsuta et al. (2008).

This paper is an extended version of the paper Nakazawa & Tatsuta (2009) by the same authors, particularly focusing on the decision problems in the calculi with multiple-quantifier rules and the type-free-style calculi.

This paper is organized as follows. Section 2 defines M-$\lambda^{\neg\wedge\exists}$ and M-F, and proves that their TC and TI are undecidable. Section 3 shows that the undecidability of TC and TI in TF-$\lambda^{\neg\wedge\exists}$ is reduced to the undecidability of TC and TI in TF-F.

2 Multiple-Quantifier Rules

In this section, we will introduce lambda calculi with multiple-quantifier rules, and prove that type checking and type inference in them are undecidable.

In Subsection 2.1, we define a negation, conjunction, and existence fragment $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$ with the *multiple-quantifier rules*. In Subsection 2.2, we will show the undecidability of TC in $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$ by reducing the semi-unification problem to TC in the system. In Subsection 2.3, we will define $\mathsf{M}\text{-}F$ and a translation from $\mathsf{Curry}\text{-}F$ to $\mathsf{M}\text{-}F$, and show TC and TI are undecidable in $\mathsf{M}\text{-}F$. In Subsection 2.4, we will define a CPS-translation from $\mathsf{M}\text{-}F$ to $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$, and show the undecidability of TC and TI in $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$ by the CPS translation with the undecidability of TC and TI in $\mathsf{M}\text{-}F$.

2.1 Multiple Existential Rules

In this subsection, we define the system $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$.

Definition 2.1 ($\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$) (1) The types are denoted by A, B,..., and called $\neg\wedge\exists$-types. The terms are denoted by M, N,.... They are defined by

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

$M ::= x \mid \lambda x.M \mid \langle M, M\rangle \mid \langle \exists^*, M\rangle \mid MM \mid M\pi_1 \mid M\pi_2 \mid M[x.M]$,

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 $N[x.M]$, the variable x is bound in M. We use $\equiv$ to denote the syntactic identity modulo renaming of bound variables.

(2) We use $\overline{X}$ and $\overline{\mathsf{A}}$ to denote a finite list of type variables and a finite list of types, respectively. When $\overline{X}$ denotes $(X_1, \cdots, X_n)$ and $\overline{\mathsf{B}}$ denotes $(\mathsf{B}_1, \cdots, \mathsf{B}_n)$, $\exists\overline{X}.\mathsf{A}$ denotes the type $\exists X_1 \cdots \exists X_n.\mathsf{A}$, and $\mathsf{A}[\overline{X} := \overline{\mathsf{B}}]$ denotes the simultaneous substitution $\mathsf{A}[X_1 := \mathsf{B}_1, \cdots, X_n := \mathsf{B}_n]$. $\overline{X}$ and $\overline{\mathsf{B}}$ may denote the empty list. When they are empty, both $\exists\overline{X}.\mathsf{A}$ and $\mathsf{A}[\overline{X} := \overline{\mathsf{B}}]$ denote the same type A. When $\overline{X}$ denotes $(X_1, \cdots, X_n)$, $\overline{X}Y$ denotes $(X_1, \cdots, X_n, Y)$.

(3) Γ denotes a context, which is a finite set of type assignments in the form of $(x : \mathsf{A})$. We require the variable condition for every context Γ by stipulating that each term variable occurs at most once in Γ. We write $\Gamma, x : \mathsf{A}$ for $\Gamma \cup \{(x : \mathsf{A})\}$. We also write Γ_1, Γ_2 for $\Gamma_1 \cup \Gamma_2$. $\neg\Gamma$ is defined as $\{(x : \neg\mathsf{A}) | (x : \mathsf{A}) \in \Gamma\}$. $\mathrm{dom}(\Gamma)$ is defined as $\{x | (x : \mathsf{A}) \in \Gamma\}$. The typing rules of $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$ are the following:

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

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

$$\frac{\Gamma_1 \vdash M : \mathsf{A} \quad \Gamma_2 \vdash N : \mathsf{B}}{\Gamma_1, \Gamma_2 \vdash \langle M, N\rangle : \mathsf{A}\wedge\mathsf{B}}\ (\wedge\mathsf{I})$$

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

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

In the rule $(\exists\mathsf{E})$, Γ_2 and C do not contain any free occurrence of any type variable from $\overline{X}$. We write $\Gamma \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} M : \mathsf{A}$ when $\Gamma \vdash M : \mathsf{A}$ is derivable by the typing rules above.

The terms of $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$ have enough type annotations to enjoy the generation lemma in the following proposition. We will implicitly use this proposition in the rest of this paper.

Proposition 2.2 (Generation Lemma) Suppose $\Gamma \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} M : \mathsf{A}$.

(1) If M is x, then Γ contains $(x : \mathsf{A})$.

(2) If M is $\lambda x.N$, then A is of the form $\neg\mathsf{B}$ and we have $\Gamma, x : \mathsf{B} \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} N : \bot$.

(3) If M is $M_1 M_2$, then A is $\bot$ and there exists B such that both $\Gamma \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} M_1 : \neg\mathsf{B}$ and $\Gamma \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} M_2 : \mathsf{B}$ hold.

(4) If M is $\langle M_1, M_2\rangle$, then A is of the form $\mathsf{A}_1 \wedge \mathsf{A}_2$ and we have $\Gamma \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} M_1 : \mathsf{A}_1$ and $\Gamma \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} M_2 : \mathsf{A}_2$.

(5) If M is $N\pi_1$, then there exists B such that $\Gamma \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} N : \mathsf{A} \wedge \mathsf{B}$ holds.

(6) If M is $N\pi_2$, then there exists B such that $\Gamma \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} N : \mathsf{B} \wedge \mathsf{A}$ holds.

(7) If M is $\langle \exists^*, N\rangle$, then A is of the form of $\exists\overline{X}.\mathsf{B}$ and we have $\Gamma \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} N : \mathsf{B}[\overline{X} := \overline{\mathsf{C}}]$ for some sequence $\overline{\mathsf{C}}$ of types.

(8) If M is $M_1[x.M_2]$, then there exists $\exists\overline{X}.\mathsf{B}$ such that no type variable from $\overline{X}$ occurs freely in Γ nor A, and we have both $\Gamma \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} M_1 : \exists\overline{X}.\mathsf{B}$ and $\Gamma, x : \mathsf{B} \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} M_2 : \mathsf{A}$.

2.2 Undecidability of TC in $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$

In this subsection, we will show the undecidability of type checking by reducing the semi-unification problem to TC in $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$.

Definition 2.3 (Semi-Unification Problem) (1) A $\wedge$-*type* is defined as a $\neg\wedge\exists$-type which contains neither $\neg$ nor $\exists$. A $\wedge$-*substitution* is defined as a simultaneous type substitution $\Theta = [\overline{X} := \overline{\mathsf{A}}]$ where each element in $\overline{\mathsf{A}}$ is a $\wedge$-type.

(2) A pair $(\mathsf{A}_1 \leq \mathsf{B}_1, \mathsf{A}_2 \leq \mathsf{B}_2)$ of inequalities between two $\wedge$-types is called an instance of the *semi-unification problem*. An instance $I = (\mathsf{A}_1 \leq \mathsf{B}_1, \mathsf{A}_2 \leq \mathsf{B}_2)$ has a solution if there exist $\wedge$-substitutions Θ, Θ_1, and Θ_2 such that $\mathsf{A}_1\Theta\Theta_1 \equiv \mathsf{B}_1\Theta$ and $\mathsf{A}_2\Theta\Theta_2 \equiv \mathsf{B}_2\Theta$.

Kfoury et al. (1993) considered the semi-unification problem for first-order expressions with the binary function symbol $\rightarrow$. Since their discussion did not depend on the semantics of the implication, the choice of the binary function symbol is arbitrary. By replacing $\rightarrow$ by $\wedge$ in their proof, the undecidability of the semi-unification problem on $\wedge$-types is proved.

Theorem 2.4 (Kfoury et al. (1993)) It is not possible to effectively decide whether a given instance of the semi-unification problem has a solution.

The next lemma shows that the semi-unification problem is reduced to TC in $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$. The proof is done by adapting the technique by Wells (1999) to $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$.

Lemma 2.5 For any instance I of the semi-unification problem, there exist Γ, M, and A such that I has a solution if and only if $\Gamma \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} M : \mathsf{A}$ holds.

Proof: In this proof, we use the abbreviation $\mathsf{A}\wedge\mathsf{B}\wedge\mathsf{C}$ for $\mathsf{A}\wedge(\mathsf{B}\wedge\mathsf{C})$ and $\langle M, N, L\rangle$ for $\langle M, \langle N, L\rangle\rangle$ respectively.

Let $I = (\mathsf{A}_1 \leq \mathsf{B}_1, \mathsf{A}_2 \leq \mathsf{B}_2)$ be an instance of the semi-unification problem, and $\overline{X}$ be the free type variables in I. Suppose that c is a fresh term variable, and Z_1 and Z_2 are fresh type variables. Take C to be $\neg\exists\overline{X}Z_1Z_2.\mathsf{C}'$ and M to be $\langle\exists^*, \lambda k.(k\pi_2)[k'.c\langle\exists^*, \langle P, P, k'\rangle\rangle]\rangle$, where C' and P are defined by

$\mathsf{C}' \equiv \neg(\neg\mathsf{B}_1 \wedge Z_2) \wedge \neg(\neg Z_1 \wedge \mathsf{B}_2) \wedge (\neg\mathsf{A}_1 \wedge \mathsf{A}_2)$,

$P \equiv \lambda k''.(k\pi_1)\langle\exists^*, k''\rangle$.

We can show that I has a solution if and only if $c : \mathsf{C} \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} M : \exists Y.\neg(\neg Y \wedge Y)$ holds.

We suppose that I has a solution $(\Theta, \Theta_1, \Theta_2)$. Let $\overline{W}$ include the union of domains of Θ_1 and Θ_2, D' be $\neg \mathsf{A}_1\Theta \wedge \mathsf{A}_2\Theta$, D be $\exists \overline{W}.\mathsf{D}'$, and Γ denote the context $\{c : \mathsf{C}, k : \neg \mathsf{D} \wedge \mathsf{D}\}$. For any substitution Ξ whose domain is included in $\overline{W}$, we have the following:

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash k\pi_1 : \neg \mathsf{D}} \quad \dfrac{\overline{k'' : \mathsf{D}'\Xi \vdash k'' : \mathsf{D}'\Xi}}{k'' : \mathsf{D}'\Xi \vdash \langle \exists^*, k'' \rangle : \mathsf{D}}}{\dfrac{\Gamma, k'' : \mathsf{D}'\Xi \vdash (k\pi_1)\langle \exists^*, k'' \rangle : \bot}{\Gamma \vdash P : \neg \mathsf{D}'\Xi}} \quad .$$

Since $\mathsf{A}_i\Theta\Theta_i \equiv \mathsf{B}_i\Theta$ holds for $i = 1$ and 2, we have $\mathsf{D}'\Theta_1 \equiv \neg \mathsf{B}_1\Theta \wedge \mathsf{A}_2\Theta\Theta_1$ and $\mathsf{D}'\Theta_2 \equiv \neg \mathsf{A}_1\Theta\Theta_2 \wedge \mathsf{B}_2\Theta$. So we have $\Gamma \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} P : \neg(\neg \mathsf{B}_1\Theta \wedge \mathsf{A}_2\Theta\Theta_1)$ and $\Gamma \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} P : \neg(\neg \mathsf{A}_1\Theta\Theta_2 \wedge \mathsf{B}_2\Theta)$. Therefore, under the context $\Gamma \cup \{k' : \mathsf{D}'\}$, the term $\langle P, P, k' \rangle$ has the type $\neg(\neg \mathsf{B}_1\Theta \wedge \mathsf{A}_2\Theta\Theta_1) \wedge \neg(\neg \mathsf{A}_1\Theta\Theta_2 \wedge \mathsf{B}_2\Theta) \wedge (\neg \mathsf{A}_1\Theta \wedge \mathsf{A}_2\Theta)$, which is identical to $\mathsf{C}'\Theta[Z_1 := \mathsf{A}_1\Theta\Theta_2][Z_2 := \mathsf{A}_2\Theta\Theta_1]$. Therefore the term $\langle \exists^*, \langle P, P, k' \rangle \rangle$ has the type $\exists \overline{X} Z_1 Z_2.\mathsf{C}'$, and $c\langle \exists^*, \langle P, P, k' \rangle \rangle$ has the type $\bot$. Hence we have the following:

$$\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\Gamma \vdash k\pi_2 : \mathsf{D}} \quad \dfrac{\vdots}{\Gamma, k' : \mathsf{D}' \vdash c\langle \exists^*, \langle P, P, k' \rangle \rangle : \bot}}{\Gamma \vdash (k\pi_2)[k'.c\langle \exists^*, \langle P, P, k' \rangle \rangle] : \bot}}{c : \mathsf{C} \vdash \lambda k.(k\pi_2)[k'.c\langle \exists^*, \langle P, P, k' \rangle \rangle] : \neg(\neg \mathsf{D} \wedge \mathsf{D})}}{c : \mathsf{C} \vdash M : \exists Y.\neg(\neg Y \wedge Y)} \quad .$$

Conversely, we suppose that $c : \mathsf{C} \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} M : \exists Y.\neg(\neg Y \wedge Y)$ holds. Then $c : \mathsf{C}, k : \neg \mathsf{E} \wedge \mathsf{E} \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} (k\pi_2)[k'.c\langle \exists^*, \langle P, P, k' \rangle \rangle] : \bot$ holds for some E. Since this judgment is a conclusion of $(\exists \mathsf{E})$, the type E of the term $k\pi_2$ must be of the form $\exists \overline{W}.\mathsf{E}'$ for some $\overline{W}$ and E' such that E' is the type of k'. Moreover, $\langle \exists^*, \langle P, P, k' \rangle \rangle$ has the type $\exists \overline{X} Z_1 Z_2.\mathsf{C}'$, so we have

(1) $k : \neg \mathsf{E} \wedge \mathsf{E} \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} P : \neg(\neg \mathsf{B}_1\Theta \wedge Z_2\Theta)$,

(2) $k : \neg \mathsf{E} \wedge \mathsf{E} \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} P : \neg(\neg Z_1\Theta \wedge \mathsf{B}_2\Theta)$,

(3) $k' : \mathsf{E}' \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} k' : \neg \mathsf{A}_1\Theta \wedge \mathsf{A}_2\Theta$,

for some substitution Θ whose domain is included in $\overline{X} Z_1 Z_2$. From (3), E' is identical to $\neg \mathsf{A}_1\Theta \wedge \mathsf{A}_2\Theta$. From (1), we have

$$\dfrac{\dfrac{\vdots}{k : \neg \mathsf{E} \wedge \mathsf{E} \vdash k\pi_1 : \neg \mathsf{E}} \quad \dfrac{\overline{k'' : \neg \mathsf{B}_1\Theta \wedge Z_2\Theta \vdash k'' : \neg \mathsf{B}_1\Theta \wedge Z_2\Theta}}{k'' : \neg \mathsf{B}_1\Theta \wedge Z_2\Theta \vdash \langle \exists^*, k'' \rangle : \mathsf{E}}\ (*)}{\dfrac{k : \neg \mathsf{E} \wedge \mathsf{E}, k'' : \neg \mathsf{B}_1\Theta \wedge Z_2\Theta \vdash (k\pi_1)\langle \exists^*, k'' \rangle : \bot}{k : \neg \mathsf{E} \wedge \mathsf{E} \vdash P : \neg(\neg \mathsf{B}_1\Theta \wedge Z_2\Theta)}} \quad .$$

Since the rule (*) is $(\exists \mathsf{I})$ and E is identical to $\exists \overline{W}.\mathsf{E}' \equiv \exists \overline{W}.(\neg \mathsf{A}_1\Theta \wedge \mathsf{A}_2\Theta)$, we have $\neg \mathsf{B}_1\Theta \wedge Z_2\Theta \equiv \mathsf{E}'\Theta_1 \equiv \neg \mathsf{A}_1\Theta\Theta_1 \wedge \mathsf{A}_2\Theta\Theta_1$ for some substitution Θ_1 whose domain is included in $\overline{W}$. Hence we have $\mathsf{B}_1\Theta \equiv \mathsf{A}_1\Theta\Theta_1$. Similarly, we have $\mathsf{B}_2\Theta \equiv \mathsf{A}_2\Theta\Theta_2$ from (2).

Since Θ, Θ_1 and Θ_2 may contain $\neg$ and $\exists$, we have to construct $\wedge$-substitutions from them. Following the idea of Wells (1999), we define the erase function with a fresh type variable Z as follows:

$e(X) = X$,

$e(\mathsf{A} \wedge \mathsf{B}) = e(\mathsf{A}) \wedge e(\mathsf{B})$,

$e(\neg \mathsf{A}) = e(\mathsf{A})$,

$e(\exists X.\mathsf{A}) = e(\mathsf{A}[X := Z])$.

We prove $e(\mathsf{A}\Theta) \equiv e(e(\mathsf{A})\Theta)$ for any $\neg \wedge \exists$-type A and any substitution Θ by induction on the size of A as follows.

(Case X) Both the sides are equal to $e(X\Theta)$.

(Case $\mathsf{B} \wedge \mathsf{C}$) The left-hand side is $e(\mathsf{B}\Theta) \wedge e(\mathsf{C}\Theta)$, which is identical to $e(e(\mathsf{B})\Theta) \wedge e(e(\mathsf{C})\Theta)$ by the induction hypothesis. The right-hand side is $e((e(\mathsf{B}) \wedge e(\mathsf{C}))\Theta) \equiv e(e(\mathsf{B})\Theta) \wedge e(e(\mathsf{C})\Theta)$.

(Case $\neg\mathsf{B}$) The left-hand side is $e(\mathsf{B}\Theta)$, which is identical to $e(e(\mathsf{B})\Theta)$ by the induction hypothesis. The right-hand side is $e(e(\neg\mathsf{B})\Theta) \equiv e(e(\mathsf{B})\Theta)$.

(Case $\exists X.\mathsf{B}$) By renaming bound variables, we suppose that X does not occur in Θ. The left-hand side is $e(\mathsf{B}\Theta[X := Z])$, which is identical to $e(\mathsf{B}[X := Z]\Theta)$ since Z is fresh. By the induction hypothesis, this type is identical to $e(e(\mathsf{B}[X := Z])\Theta)$, which is equal to the right-hand side.

Define the $\wedge$-substitutions (Ξ, Ξ_1, Ξ_2) by $X\Xi = e(X\Theta)$ and $X\Xi_i = e(X\Theta_i)$. Since $\mathsf{A}_i\Theta\Theta_i \equiv \mathsf{B}_i\Theta$ holds, we have $e(\mathsf{A}_i\Theta\Theta_i) \equiv e(\mathsf{B}_i\Theta) \equiv \Xi(\mathsf{B}_i)$. By the claim proved above, we have $e(\mathsf{A}_i\Theta\Theta_i) \equiv e(e(\mathsf{A}_i\Theta)\Theta_i) \equiv \mathsf{A}_i\Xi\Xi_i$, so the triple (Ξ, Ξ_1, Ξ_2) is a solution of I. ■

Theorem 2.6 Type checking is undecidable in M-$\lambda^{\neg\wedge\exists}$.

Proof: By Theorem 2.4 and Lemma 2.5. ■

2.3 Polymorphic Lambda Calculus with Multiple-Quantifier Rules

In this subsection, we prove that TC and TI are undecidable in M-F by showing that it is reduced to the undecidability of TC and TI in the Curry-style polymorphic lambda calculus.

Definition 2.7 (M-F) (1) The types are denoted by A, B,..., and called $\rightarrow\forall$-*types*. The terms are denoted by M, N,.... They are defined by

$A ::= X \mid A{\rightarrow}A \mid \forall X.A$,

$M ::= x \mid \lambda x.M \mid \Lambda^*.M \mid MM \mid M\bullet^*$.

In the type $\forall X.A$, the variable X is bound in A. In the term $\lambda x.M$, the variable x is bound in M. In M-F, we use $\bot$ to denote the type $\forall X.X$.

(2) The typing rules of M-F are the following:

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

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

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash \Lambda^*.M : \forall\overline{X}.A}\ (\forall\mathsf{I}) \qquad \frac{\Gamma \vdash M : \forall\overline{X}.A}{\Gamma \vdash M\bullet^* : A[\overline{X} := \overline{B}]}\ (\forall\mathsf{E})$$

In the rule $(\forall\mathsf{I})$, Γ does not contain any free occurrence of any type variable from $\overline{X}$.

The Curry-style F is defined as follows.

Definition 2.8 (Curry-F) The types of Curry-F are $\rightarrow\forall$-types. The terms of Curry-F are defined by

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

The typing rules of Curry-F are the same as those of M-F except for the rules of the universal quantifiers given as follows.

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

Wells (1999) showed the following INST-before-GEN generation lemma for Curry-F. INST and GEN mean $(\forall\mathsf{E})$ and $(\forall\mathsf{I})$, respectively.

Definition 2.9 (INST-before-GEN Property) A derivation in Curry-F is defined to satisfy the INST-before-GEN property if any premise of ($\forall$E) is not the conclusion of ($\forall$I) in the derivation.

Proposition 2.10 (Wells (1999)) (1) If $\Gamma \vdash M : A$ is derivable in Curry-F, then there is a derivation of $\Gamma \vdash M : A$ which satisfies the INST-before-GEN property.

(2) (INST-before-GEN Generation Lemma) Assume a derivation of $\Gamma \vdash M : A$ satisfies the INST-before-GEN property. We have the following.

(i) If M is a variable x, then Γ contains $x : B$ for some B such that the derivation has the following form.

$$\begin{array}{l} \overline{\Gamma \vdash x : B} \\ \quad \vdots\ (\forall\mathsf{E}) \text{ (zero or more times)} \\ \quad \vdots\ (\forall\mathsf{I}) \text{ (zero or more times)} \\ \Gamma \vdash x : A \end{array}$$

(ii) If M is a λ-abstraction $\lambda x.N$, then the derivation has the following form for some B and C.

$$\begin{array}{l} \dfrac{\Gamma, x : B \vdash N : C}{\Gamma \vdash \lambda x.N : B \to C} \\ \quad \vdots\ (\forall\mathsf{I}) \text{ (zero or more times)} \\ \Gamma \vdash \lambda x.N : A \end{array}$$

(iii) If M is an application $N_1 N_2$, then the derivation has the following form for some B and C.

$$\begin{array}{l} \dfrac{\Gamma \vdash N_1 : C \to B \quad \Gamma \vdash N_2 : C}{\Gamma \vdash N_1 N_2 : B} \\ \quad \vdots\ (\forall\mathsf{E}) \text{ (zero or more times)} \\ \quad \vdots\ (\forall\mathsf{I}) \text{ (zero or more times)} \\ \Gamma \vdash N_1 N_2 : A \end{array}$$

A term of M-F contains more type annotations than that of Curry-F, so M-F seems more restricted when we type terms of M-F. However, we can define an embedding from Curry-F into M-F. The embedding shows that the system M-F explicitly captures the behavior of the INST-before-GEN generation lemma.

Definition 2.11 The embedding map $\lfloor \cdot \rfloor$ from the Curry-F-terms to the M-F-terms is defined by
$\lfloor x \rfloor \equiv \Lambda^*.x\bullet^*$,
$\lfloor \lambda x.M \rfloor \equiv \Lambda^*.\lambda x.\lfloor M \rfloor$,
$\lfloor MN \rfloor \equiv \Lambda^*.\lfloor M \rfloor \lfloor N \rfloor \bullet^*$.

Proposition 2.12 $\Gamma \vdash_{\text{Curry-}F} M : A$ holds if and only if $\Gamma \vdash_{\text{M-}F} \lfloor M \rfloor : A$.

Proof: By noting that $\overline{X}$ in ($\forall$I) and ($\forall$E) of M-F may be an empty sequence, it is easily proved that $\Gamma \vdash_{\text{M-}F} \lfloor M \rfloor : A$ implies $\Gamma \vdash_{\text{Curry-}F} M : A$. We will prove the converse direction by induction on M. Suppose that we have a derivation of $\Gamma \vdash_{\text{Curry-}F} M : A$. By Proposition 2.10 (1), we can assume that the derivation enjoys the INST-before-GEN property.

(Case $M \equiv x$) By Proposition 2.10 (2) (i), the derivation of $\Gamma \vdash_{\text{Curry-}F} x : A$ is of the following form, where $\Gamma', x : B$ is identical to Γ.

$$\begin{array}{l} \overline{\Gamma', x : B \vdash x : B} \\ \quad \vdots\ (\forall\mathsf{E}) \text{ (zero or more times)} \\ \Gamma', x : B \vdash x : B' \\ \quad \vdots\ (\forall\mathsf{I}) \text{ (zero or more times)} \\ \Gamma', x : B \vdash x : A \end{array}$$

Hence we have $\Gamma', x : B \vdash_{\mathsf{M}\text{-}F} x\bullet^* : B'$, so $\Gamma', x : B \vdash_{\mathsf{M}\text{-}F} \Lambda^*.x\bullet^* : A$.

(Case $M \equiv \lambda x.N$) By Proposition 2.10 (2) (ii), the derivation of $\Gamma \vdash_{\mathsf{Curry}\text{-}F} \lambda x.N : A$ is of the following form.

$$\dfrac{\dfrac{\Gamma, x : B \vdash N : C}{\Gamma \vdash \lambda x.N : B{\to}C}}{\vdots\ (\forall\mathsf{I}) \text{ (zero or more times)}}$$
$$\Gamma \vdash \lambda x.N : A$$

By the induction hypothesis, we have $\Gamma, x : B \vdash_{\mathsf{M}\text{-}F} \lfloor N \rfloor : C$, so we have $\Gamma \vdash_{\mathsf{M}\text{-}F} \lambda x.\lfloor N \rfloor : B{\to}C$. Hence we have $\Gamma \vdash_{\mathsf{M}\text{-}F} \Lambda^*.\lambda x.\lfloor N \rfloor : A$.

(Case $M \equiv N_1N_2$) By Proposition 2.10 (2) (iii), the derivation of $\Gamma \vdash_{\mathsf{Curry}\text{-}F} N_1N_2 : A$ is of the following form.

$$\dfrac{\dfrac{\dfrac{\Gamma \vdash N_1 : C{\to}B \quad \Gamma \vdash N_2 : C}{\Gamma \vdash N_1N_2 : B}}{\vdots\ (\forall\mathsf{E}) \text{ (zero or more times)}}}{\dfrac{\Gamma \vdash N_1N_2 : B'}{\vdots\ (\forall\mathsf{I}) \text{ (zero or more times)}}}$$
$$\Gamma \vdash N_1N_2 : A$$

By the induction hypotheses, we have $\Gamma \vdash_{\mathsf{M}\text{-}F} \lfloor N_1 \rfloor : C{\to}B$ and $\Gamma \vdash_{\mathsf{M}\text{-}F} \lfloor N_2 \rfloor : C$, so we have $\Gamma \vdash_{\mathsf{M}\text{-}F} \lfloor N_1 \rfloor \lfloor N_2 \rfloor : B$. We have $\Gamma \vdash_{\mathsf{M}\text{-}F} \lfloor N_1 \rfloor \lfloor N_2 \rfloor \bullet^* : B'$, so $\Gamma \vdash_{\mathsf{M}\text{-}F} \Lambda^*.\lfloor N_1 \rfloor \lfloor N_2 \rfloor \bullet^* : A$. ■

Proposition 2.13 Type checking and type inference are undecidable in $\mathsf{M}\text{-}F$.

Proof: By Proposition 2.12, the undecidability of TC and TI in $\mathsf{M}\text{-}F$ is reduced to the undecidability of TC and TI in $\mathsf{Curry}\text{-}F$, which has been proved in Wells (1999). ■

2.4 Undecidability of TI in $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$

We will prove that TC and TI in $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$ is reduced to those in $\mathsf{M}\text{-}F$.

In order for the reduction, we will borrow the idea of the contraction translation on types from Nakazawa et al. (2008). However, we cannot directly apply the original definition of the translation, and some modification is needed. Along the lines of the proof in Nakazawa et al. (2008), we will define a negative translation $(\cdot)^\bullet$ from the $\to\forall$-types to the $\neg\wedge\exists$-types, a CPS translation $[\![\cdot]\!]$ from $\mathsf{M}\text{-}F$ to $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$, and a subsystem $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}_{\mathsf{cps}}$ of $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$, which is the image of the CPS translation. Then we will prove that M has a type A in $\mathsf{M}\text{-}F$ if and only if $[\![M]\!]$ has the type $\neg A^\bullet$ in $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}_{\mathsf{cps}}$. Furthermore, we will show that $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$ is a conservative extension of $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}_{\mathsf{cps}}$, that is, for any term of the form $[\![M]\!]$, if $[\![M]\!]$ has a type $\neg A^\bullet$ in $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$, then $[\![M]\!]$ has the type $\neg A^\bullet$ in $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}_{\mathsf{cps}}$. By these facts, we will conclude that M has a type A in $\mathsf{M}\text{-}F$ if and only if $[\![M]\!]$ has the type $\neg A^\bullet$ in $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$. Hence TC and TI in $\mathsf{M}\text{-}F$ will be reduced to those in $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$.

Definition 2.14 (CPS Translation) (1) The *negative translation* from the $\to\forall$-types to the $\neg\wedge\exists$-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 $\{(x : \mathsf{A}^\bullet) | (x : \mathsf{A}) \in \Gamma\}$.

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

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

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

$[\![MN]\!] \equiv \lambda k.[\![M]\!]\langle[\![N]\!], k\rangle$,
$[\![\Lambda^*.M]\!] \equiv \lambda k.k[k'.[\![M]\!]k']$,
$[\![M\bullet^*]\!] \equiv \lambda k.[\![M]\!]\langle\exists^*, k\rangle$,
where k and k' are supposed to be fresh variables.

Proposition 2.15 $\Gamma \vdash_{\mathsf{M}\text{-}F} M : A$ implies $\neg\Gamma^\bullet \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} [\![M]\!] : \neg A^\bullet$.

Proof: The proposition is proved by induction on the derivation of $\Gamma \vdash_{\mathsf{M}\text{-}F} M : A$.

Case (Ax). When the derivation is

$$\frac{}{\Gamma, x : A \vdash x : A}$$

we have the following in $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$.

$$\dfrac{\dfrac{\dfrac{}{\neg\Gamma^\bullet, x : \neg A^\bullet, k : A^\bullet \vdash x : \neg A^\bullet} \quad \dfrac{}{\neg\Gamma^\bullet, x : \neg A^\bullet, k : A^\bullet \vdash k : A^\bullet}}{\neg\Gamma^\bullet, x : \neg A^\bullet, k : A^\bullet \vdash xk : \bot}}{\neg\Gamma^\bullet, x : \neg A^\bullet \vdash \lambda k.xk : \neg A^\bullet}$$

Case ($\to$I). Suppose that the last rule of the derivation is the following.

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \to B}$$

By the induction hypothesis, we have $\neg\Gamma^\bullet, x : \neg A^\bullet \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} [\![M]\!] : \neg B^\bullet$, so we have $\neg\Gamma^\bullet, k : \neg A^\bullet \wedge B^\bullet, x : \neg A^\bullet \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} [\![M]\!](k\pi_2) : \bot$. We have the following in $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$.

$$\dfrac{\dfrac{\dfrac{\neg\Gamma^\bullet, k : \neg A^\bullet \wedge B^\bullet, x : \neg A^\bullet \vdash [\![M]\!](k\pi_2) : \bot}{\neg\Gamma^\bullet, k : \neg A^\bullet \wedge B^\bullet \vdash \lambda x.[\![M]\!](k\pi_2) : \neg\neg A^\bullet} \quad k : \neg A^\bullet \wedge B^\bullet \vdash k\pi_1 : \neg A^\bullet}{\neg\Gamma^\bullet, k : \neg A^\bullet \wedge B^\bullet \vdash (\lambda x.[\![M]\!](k\pi_2))(k\pi_1) : \bot}}{\neg\Gamma^\bullet \vdash \lambda k.(\lambda x.[\![M]\!](k\pi_2))(k\pi_1) : \neg(\neg A^\bullet \wedge B^\bullet)}$$

Case ($\to$E). Suppose that the last rule of the derivation is the following.

$$\frac{\Gamma_1 \vdash M : A \to B \quad \Gamma_2 \vdash N : A}{\Gamma_1, \Gamma_2 \vdash MN : B}$$

By the induction hypotheses, we have $\neg\Gamma_1^\bullet \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} [\![M]\!] : \neg(\neg A^\bullet \wedge B^\bullet)$ and $\neg\Gamma_2^\bullet \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} [\![N]\!] : \neg A^\bullet$. We have the following.

$$\dfrac{\dfrac{\neg\Gamma_1^\bullet, k : B^\bullet \vdash [\![M]\!] : \neg(\neg A^\bullet \wedge B^\bullet) \quad \dfrac{\neg\Gamma_2^\bullet \vdash [\![N]\!] : \neg A^\bullet \quad \dfrac{}{k : B^\bullet \vdash k : B^\bullet}}{\neg\Gamma_2^\bullet, k : B^\bullet \vdash \langle[\![N]\!], k\rangle : \neg A^\bullet \wedge B^\bullet}}{\neg\Gamma_1^\bullet, \neg\Gamma_2^\bullet, k : B^\bullet \vdash [\![M]\!]\langle[\![N]\!], k\rangle : \bot}}{\neg\Gamma_1^\bullet, \neg\Gamma_2^\bullet \vdash \lambda k.[\![M]\!]\langle[\![N]\!], k\rangle : \neg B^\bullet}$$

Case ($\forall$I). Suppose that the last rule of the derivation is the following.

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash \Lambda^*.M : \forall \overline{X}.A}$$

By the induction hypothesis, we have $\neg\Gamma^\bullet \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} [\![M]\!] : \neg A^\bullet$, so we have $\neg\Gamma^\bullet, k : A^\bullet \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} [\![M]\!]k : \bot$. Since Γ does not contain any free type variable from $\overline{X}$ by the variable condition of $(\forall\mathsf{I})$, $\neg\Gamma^\bullet$ does not contain any free type variable from $\overline{X}$ either. We have the following.

$$\dfrac{\dfrac{}{k : \exists\overline{X}.A^\bullet \vdash k : \exists\overline{X}.A^\bullet} \quad \begin{array}{c}\vdots\\ \neg\Gamma^\bullet, k' : A^\bullet \vdash [\![M]\!]k' : \bot\end{array}}{\dfrac{\neg\Gamma^\bullet, k : \exists\overline{X}.A^\bullet \vdash k[k'.[\![M]\!]k'] : \bot}{\neg\Gamma^\bullet \vdash \lambda k.k[k'.[\![M]\!]k'] : \neg\exists\overline{X}.A^\bullet}}$$

Case $(\forall\mathsf{E})$. Suppose that the last rule of the derivation is the following.

$$\dfrac{\Gamma \vdash M : \forall\overline{X}.A}{\Gamma \vdash M\bullet^* : A[\overline{X} := \overline{B}]}$$

By the induction hypothesis, we have $\neg\Gamma^\bullet \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} [\![M]\!] : \neg\exists\overline{X}.A^\bullet$. We have the following.

$$\dfrac{\begin{array}{c}\vdots\\ \neg\Gamma^\bullet \vdash [\![M]\!] : \neg\exists\overline{X}.A^\bullet\end{array} \quad \dfrac{\dfrac{}{k : A^\bullet[\overline{X} := \overline{B^\bullet}] \vdash k : A^\bullet[\overline{X} := \overline{B^\bullet}]}}{k : A^\bullet[\overline{X} := \overline{B^\bullet}] \vdash \langle\exists^*, k\rangle : \exists\overline{X}.A^\bullet}}{\dfrac{\neg\Gamma^\bullet, k : A^\bullet[\overline{X} := \overline{B^\bullet}] \vdash [\![M]\!]\langle\exists^*, k\rangle : \bot}{\neg\Gamma^\bullet \vdash \lambda k.[\![M]\!]\langle\exists^*, k\rangle : \neg A^\bullet[\overline{X} := \overline{B^\bullet}]}}$$

where $\overline{B^\bullet}$ denotes the sequence $(B_1^\bullet, \cdots, B_n^\bullet)$ for $\overline{B} = (B_1, \cdots, B_n)$. Note that $(A[\overline{X} := \overline{B}])^\bullet \equiv A^\bullet[\overline{X} := \overline{B^\bullet}]$ is easily proved by induction on A. ■

Definition 2.16 ($\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}_{\mathsf{cps}}$) (1) The *continuation types*, denoted by $\mathcal{A}$, $\mathcal{B}$,..., are defined by
$\mathcal{A} ::= X \mid \neg\mathcal{A} \wedge \mathcal{A} \mid \exists X.\mathcal{A}$.
A *CPS type* is define as a type of the form $\neg\mathcal{A}$ for some continuation type $\mathcal{A}$. The *CPS terms*, denoted by P, Q,..., are defined by
$P ::= \lambda k.xk \mid \lambda k.(\lambda x.P(k\pi_2)(k\pi_1)) \mid \lambda k.P\langle Q, k\rangle \mid \lambda k.P\langle\exists^*, k\rangle \mid \lambda k.k[k'.Pk']$,
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 $k_1 \not\equiv k_2$. We define the subsystem $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}_{\mathsf{cps}}$ of $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$ by restricting their terms and their types to the CPS terms and the CPS types, respectively. The judgments of $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}_{\mathsf{cps}}$ are restricted to those of the form $\neg\Gamma \vdash P : \neg\mathcal{A}$, where each type assignment in $\neg\Gamma$ is of the form $(x : \neg\mathcal{B})$ for some continuation type $\mathcal{B}$. The typing rules of $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}_{\mathsf{cps}}$ are the following.

$$\dfrac{}{\neg\Gamma, x : \neg\mathcal{A} \vdash \lambda k.xk : \neg\mathcal{A}}$$

$$\dfrac{\neg\Gamma, x : \neg\mathcal{A} \vdash P : \neg\mathcal{B}}{\neg\Gamma \vdash \lambda k.(\lambda x.P(k\pi_2))(k\pi_1) : \neg(\neg\mathcal{A} \wedge \mathcal{B})}$$

$$\dfrac{\neg\Gamma_1 \vdash P : \neg(\neg\mathcal{A} \wedge \mathcal{B}) \quad \neg\Gamma_2 \vdash Q : \neg\mathcal{A}}{\neg\Gamma_1, \neg\Gamma_2 \vdash \lambda k.P\langle Q, k\rangle : \neg\mathcal{B}}$$

$$\dfrac{\neg\Gamma \vdash P : \neg(\exists\overline{X}.\mathcal{B})}{\neg\Gamma \vdash \lambda k.P\langle\exists^*, k\rangle : \neg\mathcal{B}[\overline{X} := \overline{\mathcal{A}}]} \qquad \dfrac{\neg\Gamma \vdash P : \neg\mathcal{B}}{\neg\Gamma \vdash \lambda k.k[k'.Pk'] : \neg\exists\overline{X}.\mathcal{B}}$$

In the last rule, Γ does not contain any type variable in $\overline{X}$ freely. We write $\neg\Gamma \vdash_{\mathsf{cps}} P : \neg\mathcal{A}$ when the judgment is derivable by the rules above.

(2)The inverse translation $(\cdot)^\circ$ from the continuation types to the $\rightarrow\forall$-types is defined by

$X^\circ \equiv X$,
$(\neg\mathcal{A} \wedge \mathcal{B})^\circ \equiv \mathcal{A}^\circ \to \mathcal{B}^\circ$,
$(\exists X.\mathcal{A})^\circ \equiv \forall X.\mathcal{A}^\circ$.
(3) The inverse translation $(\cdot)^\#$ from the CPS terms to the terms of M-F is defined by
$(\lambda k.xk)^\# \equiv x$,
$(\lambda k.(\lambda x.P(k\pi_2))(k\pi_1))^\# \equiv \lambda x.P^\#$,
$(\lambda k.k[k'.Pk'])^\# \equiv \Lambda^*.P^\#$,
$(\lambda k.P\langle Q, k\rangle)^\# \equiv P^\# Q^\#$,
$(\lambda k.P\langle \exists^*, k\rangle)^\# \equiv P^\# \bullet^*$.

Lemma 2.17 (1) For any $\to\forall$-type A, $A^\bullet$ is a continuation type, and $A^{\bullet\circ} \equiv A$ holds.
(2) For any M-F-term M, $[\![M]\!]$ is a CPS term, and $[\![M]\!]^\# \equiv M$ holds.

Proof: (1) By induction on A.
(2) By induction on M. ■

Proposition 2.18 (1) If $\neg\Gamma \vdash_{\mathsf{cps}} P : \neg\mathcal{A}$ holds, then $\Gamma^\circ \vdash_{\text{M-}F} P^\# : \mathcal{A}^\circ$ holds.
(2) If $\neg\Gamma^\bullet \vdash_{\mathsf{cps}} [\![M]\!] : \neg A^\bullet$, then $\Gamma \vdash_{\text{M-}F} M : A$ holds.

Proof: (1) By induction on the derivation of $\neg\Gamma \vdash_{\mathsf{cps}} P : \neg\mathcal{A}$.
(2) By (1), we have $\Gamma^{\bullet\circ} \vdash_{\text{M-}F} [\![M]\!]^\# : A^{\bullet\circ}$. By Lemma 2.17, we have the claim. ■

In order to prove the conservativeness of M-$\lambda^{\neg\wedge\exists}$ over M-$\lambda^{\neg\wedge\exists}_{\mathsf{cps}}$, that is, $\neg\Gamma^\bullet \vdash_{\text{M-}\lambda^{\neg\wedge\exists}} [\![M]\!] : \neg A^\bullet$ implies $\neg\Gamma^\bullet \vdash_{\mathsf{cps}} [\![M]\!] : \neg A^\bullet$, we define the contraction translation on types, which has been introduced in Nakazawa et al. (2008). A type derivation of a CPS term in M-$\lambda^{\neg\wedge\exists}$ may contain a $\neg\wedge\exists$-type that is not a CPS type. For example, a CPS term $Q \equiv \lambda k'.xk'$ may have an arbitrary negation type $\neg\mathsf{A}$ under the context $\{x : \neg\mathsf{A}\}$, and then $P \equiv \lambda k.(\lambda x.Q(k\pi_2))(k\pi_1)$ may have the type $\neg(\neg\mathsf{A}\wedge\mathsf{A})$ under the empty context. If A is not a continuation type, the type derivation of $P : \neg(\neg\mathsf{A}\wedge\mathsf{A})$ is not in M-$\lambda^{\neg\wedge\exists}_{\mathsf{cps}}$. However, such a type A cannot be consumed in type derivations of CPS terms, so we can replace A by some continuation type without changing the form of the derivation. The contraction translation formally realizes this replacement, and derivations in M-$\lambda^{\neg\wedge\exists}$ are translated to those in M-$\lambda^{\neg\wedge\exists}_{\mathsf{cps}}$. The contraction translation $(\cdot)^{\mathsf{c}}$ is defined as follows so that $\Gamma^{\mathsf{c}} \vdash_{\mathsf{cps}} P : \mathsf{A}^{\mathsf{c}}$ holds for any type derivation of $\Gamma \vdash_{\text{M-}\lambda^{\neg\wedge\exists}} P : \mathsf{A}$ for a CPS term P.

Definition 2.19 (Contraction Translation) Let $\mathcal{S}$ be a fixed closed continuation type, for example, $\exists X.X$. The contraction translation $(\cdot)^{\mathsf{c}}$ from the $\neg\wedge\exists$-types to the CPS types, and the auxiliary translation $(\cdot)^{\mathsf{d}}$ from the $\neg\wedge\exists$-types to the continuation types are defined by
$(\neg\mathsf{A})^{\mathsf{c}} \equiv \neg\mathsf{A}^{\mathsf{d}}$,
$\mathsf{A}^{\mathsf{c}} \equiv \neg\mathsf{A}^{\mathsf{d}}$ (A is not a negation),
$X^{\mathsf{d}} \equiv X$,
$\bot^{\mathsf{d}} \equiv \mathcal{S}$,
$(\neg\mathsf{A})^{\mathsf{d}} \equiv \mathsf{A}^{\mathsf{d}}$,
$(\mathsf{A}\wedge\mathsf{B})^{\mathsf{d}} \equiv \mathsf{A}^{\mathsf{c}} \wedge \mathsf{B}^{\mathsf{d}}$,
$(\exists X.\mathsf{A})^{\mathsf{d}} \equiv \exists X.\mathsf{A}^{\mathsf{d}}$.
Γ^{c} is defined as $\{(x : \mathsf{A}^{\mathsf{c}}) | (x : \mathsf{A}) \in \Gamma\}$.

In Nakazawa et al. (2008), A^{c} for a non-negation type A was defined as $\mathcal{S}$, and it worked well for the domain-free calculi. For the calculi with multiple-quantifier rules, we have to change the definition for the following reason. The contraction translation is expected to have the following

property: $\Gamma \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} P : \mathsf{A}$ implies $\Gamma^{\mathsf{c}} \vdash_{\mathsf{cps}} P : \mathsf{A}^{\mathsf{c}}$ for any CPS term P. In the case of $P \equiv \lambda k.Q\langle \exists^*, k\rangle$, Q must have a type of the form $\neg\exists\overline{X}.\mathsf{C}$, and P has a type $\mathsf{A} \equiv \neg\mathsf{C}[\overline{X} := \overline{\mathsf{B}}]$ for some $\overline{\mathsf{B}}$, so we must show that P has the type $(\neg\mathsf{C}[\overline{X} := \overline{\mathsf{B}}])^{\mathsf{c}} \equiv \neg(\mathsf{C}[\overline{X} := \overline{\mathsf{B}}])^{\mathsf{d}}$ from the fact that Q has the type $(\neg\exists\overline{X}.\mathsf{C})^{\mathsf{c}} \equiv \neg\exists\overline{X}.\mathsf{C}^{\mathsf{d}}$. In order to prove it, it is sufficient to show the commutativity of the contraction and the substitution, that is, $(\mathsf{C}[\overline{X} := \overline{\mathsf{B}}])^{\mathsf{d}} \equiv \mathsf{C}^{\mathsf{d}}[\overline{X} := \overline{\mathsf{B}}^{\mathsf{d}}]$. However, the original contraction does not have this property.

We note that, in the case of the domain-free $\lambda^{\neg\wedge\exists}$, we needed only the restricted form of commutativity, that is, $(\mathsf{C}[X := \mathcal{B}])^{\mathsf{d}} \equiv \mathsf{C}^{\mathsf{d}}[X := \mathcal{B}]$, because the CPS term corresponding to P above is $\lambda k.Q\langle \mathcal{B}, k\rangle$, where the type abstracted by the $\exists$-introduction is restricted to the continuation type $\mathcal{B}$. The restricted commutativity is proved more easily, since any continuation type is not a negation.

Lemma 2.20 (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 $\neg\wedge\exists$-types A and B, we have (i) $(\mathsf{B}[X := \mathsf{A}])^{\mathsf{c}} \equiv \mathsf{B}^{\mathsf{c}}[X := \mathsf{A}^{\mathsf{d}}]$ and (ii) $(\mathsf{B}[X := \mathsf{A}])^{\mathsf{d}} \equiv \mathsf{B}^{\mathsf{d}}[X := \mathsf{A}^{\mathsf{d}}]$.

Proof: (1) It is easily proved by induction on $\mathcal{A}$.

(2) By induction on B. In the following, we write $\mathsf{B}[\mathsf{A}]$ for $\mathsf{B}[X := \mathsf{A}]$. We will prove (i) $(\mathsf{B}[\mathsf{A}])^{\mathsf{c}} \equiv \mathsf{B}^{\mathsf{c}}[\mathsf{A}^{\mathsf{d}}]$ and (ii) $(\mathsf{B}[\mathsf{A}])^{\mathsf{d}} \equiv \mathsf{B}^{\mathsf{d}}[\mathsf{A}^{\mathsf{d}}]$ simultaneously by induction on B.

(i) Case $\mathsf{B} \equiv X$. We have $(X[\mathsf{A}])^{\mathsf{c}} \equiv \mathsf{A}^{\mathsf{c}}$ and $X^{\mathsf{c}}[\mathsf{A}^{\mathsf{d}}] \equiv \neg\mathsf{A}^{\mathsf{d}}$. If A is not a negation, these are the same type by the definition. Otherwise, by letting $\mathsf{A} \equiv \neg\mathsf{C}$, we have $(\neg\mathsf{C})^{\mathsf{c}} \equiv \neg\mathsf{C}^{\mathsf{d}}$ and $\neg(\neg\mathsf{C})^{\mathsf{d}} \equiv \neg\mathsf{C}^{\mathsf{d}}$.

The other cases are proved by (ii).

(ii) Case $\mathsf{B} \equiv \bot$. In this case, both the sides are $\mathcal{S}$.

Case $\mathsf{B} \equiv X$. In this case, both the sides are A^{d}.

Case $\mathsf{B} \equiv Y$ ($Y \not\equiv X$). Both the sides are Y.

Case $\mathsf{B} \equiv \neg\mathsf{C}$. We have $(\neg\mathsf{C}[\mathsf{A}])^{\mathsf{d}} \equiv (\mathsf{C}[\mathsf{A}])^{\mathsf{d}}$, which is identical to $\mathsf{C}^{\mathsf{d}}[\mathsf{A}^{\mathsf{d}}]$ by the induction hypothesis. On the other hand, we have $(\neg\mathsf{C})^{\mathsf{d}}[\mathsf{A}^{\mathsf{d}}] \equiv \mathsf{C}^{\mathsf{d}}[\mathsf{A}^{\mathsf{d}}]$ by the definition.

Case $\mathsf{B} \equiv \mathsf{C} \wedge \mathsf{D}$. If C is a negation, $\mathsf{C}[\mathsf{A}]$ is a negation, so this case is easily proved by the induction hypothesis. If $\mathsf{C}[\mathsf{A}]$ is not a negation, C is not a negation either, so this case is also easily proved by the induction hypothesis. The remaining case is the case where C is not a negation and $\mathsf{C}[\mathsf{A}]$ is a negation, that is, $\mathsf{C} \equiv X$ and $\mathsf{A} \equiv \neg\mathsf{E}$. Then we have $(X[\neg\mathsf{E}] \wedge \mathsf{B}[\neg\mathsf{E}])^{\mathsf{d}} \equiv \neg\mathsf{E}^{\mathsf{d}} \wedge (\mathsf{B}[\neg\mathsf{E}])^{\mathsf{d}}$, which is identical to $\neg\mathsf{E}^{\mathsf{d}} \wedge \mathsf{B}^{\mathsf{d}}[\mathsf{E}^{\mathsf{d}}]$ by the induction hypothesis. On the other hand, we have $(X \wedge \mathsf{B})^{\mathsf{d}}[(\neg\mathsf{E})^{\mathsf{d}}] \equiv (\neg X \wedge \mathsf{B}^{\mathsf{d}})[\mathsf{E}^{\mathsf{d}}]$, so both the sides are the same type.

Case $\mathsf{B} \equiv \exists Y.\mathsf{C}$. This case is proved by the induction hypothesis. ■

Lemma 2.21 For any CPS term P, $\Gamma \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} P : A$ implies $\Gamma^{\mathsf{c}} \vdash_{\mathsf{cps}} P : A^{\mathsf{c}}$.

Proof: By induction on P. In this proof, we implicitly use the generation lemma (Proposition 2.2). Any type of P is a negation, since any CPS term is a λ-abstraction. We will show that $\Gamma \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} P : \neg\mathsf{A}$ implies $\Gamma^{\mathsf{c}} \vdash_{\mathsf{cps}} P : \neg\mathsf{A}^{\mathsf{d}}$.

Case $P \equiv \lambda k.xk$. Any derivation of $\Gamma \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} P : \neg\mathsf{A}$ has the following form.

$$\dfrac{\dfrac{\overline{\Gamma \vdash x : \neg\mathsf{A}} \quad \overline{k : \mathsf{A} \vdash k : \mathsf{A}}}{\Gamma, k : \mathsf{A} \vdash xk : \bot}}{\Gamma \vdash \lambda k.xk : \neg\mathsf{A}}$$

We have $(x : \neg\mathsf{A}) \in \Gamma$, so $(x : \neg\mathsf{A}^{\mathsf{d}}) \in \Gamma^{\mathsf{c}}$ holds.

Case $P \equiv \lambda k.(\lambda x.Q(k\pi_2))(k\pi_1)$. Any derivation of $\Gamma \vdash_{\lambda^{\neg\wedge\exists}} P : \neg\mathsf{A}$ has the following form:

$$\dfrac{\dfrac{\dfrac{\Gamma, x : \mathsf{B} \vdash Q : \neg\mathsf{C} \quad \dfrac{\overline{\Gamma' \vdash k : \mathsf{A}}}{\Gamma' \vdash k\pi_2 : \mathsf{C}}}{\Gamma', x : \mathsf{B} \vdash Q(k\pi_2) : \bot}}{\Gamma' \vdash \lambda x.Q(k\pi_2) : \neg\mathsf{B}} \quad \dfrac{\overline{\Gamma' \vdash k : \mathsf{A}}}{\Gamma' \vdash k\pi_1 : \mathsf{B}}}{\dfrac{\Gamma' \vdash (\lambda x.Q(k\pi_2))(k\pi_1) : \bot}{\Gamma \vdash \lambda k.(\lambda x.Q(k\pi_2))(k\pi_1) : \neg\mathsf{A}}}\ ,$$

where A is $\mathsf{B} \wedge \mathsf{C}$ and $\Gamma' = \Gamma, k : \mathsf{A}$. By the induction hypothesis, we have $\Gamma^{\mathsf{c}}, x : \mathsf{B}^{\mathsf{c}} \vdash_{\mathsf{cps}} Q : \neg\mathsf{C}^{\mathsf{d}}$, so we have $\Gamma^{\mathsf{c}} \vdash_{\mathsf{cps}} P : \neg(\mathsf{B}^{\mathsf{c}} \wedge \mathsf{C}^{\mathsf{d}})$, where $\mathsf{A}^{\mathsf{d}} \equiv (\mathsf{B} \wedge \mathsf{C})^{\mathsf{d}} \equiv \mathsf{B}^{\mathsf{c}} \wedge \mathsf{C}^{\mathsf{d}}$.

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

$$\dfrac{\dfrac{\overline{k : \mathsf{A} \vdash k : \mathsf{A}} \quad \dfrac{\Gamma \vdash Q : \neg\mathsf{B} \quad \overline{k' : \mathsf{B} \vdash k' : \mathsf{B}}}{\Gamma, k' : \mathsf{B} \vdash Qk' : \bot}}{\Gamma, k : \mathsf{A} \vdash k[k.Qk'] : \bot}}{\Gamma \vdash \lambda k.k[k'.Qk'] : \neg\mathsf{A}}$$

where A is $\exists\overline{X}.\mathsf{B}$, and Γ does not contain any type variable in $\overline{X}$ freely. By the induction hypothesis, we have $\Gamma^{\mathsf{c}} \vdash_{\mathsf{cps}} Q : \neg\mathsf{B}^{\mathsf{d}}$. Since Γ^{c} does not contain any variable of $\overline{X}$ freely, we have $\Gamma^{\mathsf{c}} \vdash_{\mathsf{cps}} P : \neg\exists\overline{X}.\mathsf{B}^{\mathsf{d}}$ where $\exists\overline{X}.\mathsf{B}^{\mathsf{d}} \equiv (\exists\overline{X}.\mathsf{B})^{\mathsf{d}}$.

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

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

for some list of $\neg\wedge\exists$-types $\overline{\mathsf{B}}$, where A is $\mathsf{C}[\overline{X} := \overline{\mathsf{B}}]$. By the induction hypothesis, $\Gamma^{\mathsf{c}} \vdash_{\mathsf{cps}} Q : \neg\exists\overline{X}.\mathsf{C}^{\mathsf{d}}$ holds, so we have $\Gamma^{\mathsf{c}} \vdash_{\mathsf{cps}} P : \neg\mathsf{C}^{\mathsf{d}}[\overline{X} := \overline{\mathsf{B}}^{\mathsf{d}}]$ by letting $k : \mathsf{C}^{\mathsf{d}}[X := \overline{\mathsf{B}}^{\mathsf{d}}]$, which is identical to $(\mathsf{C}[\overline{X} := \overline{\mathsf{B}}])^{\mathsf{d}}$ by Lemma 2.20 (2). ■

By Lemma 2.21, we reduce TC and TI of M-F to those of M-$\lambda^{\neg\wedge\exists}$, and conclude the undecidability of TC and TI in M-$\lambda^{\neg\wedge\exists}$.

Proposition 2.22 (1) $\Gamma \vdash_{\mathsf{M}\text{-}F} M : A$ holds if and only if $\neg\Gamma^{\bullet} \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} [\![M]\!] : \neg A^{\bullet}$ holds.

(2) $\Gamma \vdash_{\mathsf{M}\text{-}F} M : A$ holds for some Γ and A if and only if $\Gamma' \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} [\![M]\!] : \mathsf{A}'$ holds for some Γ' and A'.

Proof: (1) The only-if part is Proposition 2.15. We will show the if part. If $\neg\Gamma^\bullet \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} [\![M]\!] : \neg A^\bullet$ holds, by Lemma 2.21, 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 2.20 (1). By Proposition 2.18 (2), $\Gamma \vdash_{\mathsf{M}\text{-}F} M : A$ holds.

(2) The only-if part follows from the only-if part of (1). We will show the if part. Suppose $\Gamma' \vdash_{\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}} [\![M]\!] : \mathsf{A}'$ holds for some Γ' and A'. By Lemma 2.21, we have $\Gamma'^{\mathsf{c}} \vdash_{\mathsf{cps}} [\![M]\!] : \mathsf{A}'^{\mathsf{c}}$, so $[\![M]\!]^\#$ is typable by Proposition 2.18 (2). $[\![M]\!]^\#$ is identical to M by Lemma 2.17 (2). ■

Theorem 2.23 Type checking and type inference are undecidable in $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$.

Proof: By Propositions 2.13 and 2.22. ■

3 TC and TI in Type-Free-Style $\lambda^{\neg\wedge\exists}$

This section defines the negation, conjunction, and existence fragment $\mathsf{TF}\text{-}\lambda^{\neg\wedge\exists}$ in the type-free style, and proves that the undecidability of TC and TI in $\mathsf{TF}\text{-}\lambda^{\neg\wedge\exists}$ is reduced to the undecidability of TC and TI in $\mathsf{TF}\text{-}F$.

The type-free-style calculi are discussed in Tatsuta (2007), Tatsuta et al. (2008), and Fujita & Schubert (2009). This system was called the Curry style in Tatsuta (2007), since the system has the least type annotations among systems that have the subject reduction property.

Definition 3.1 ($\mathsf{TF}\text{-}\lambda^{\neg\wedge\exists}$) (1) The types of $\mathsf{TF}\text{-}\lambda^{\neg\wedge\exists}$ are $\neg\wedge\exists$-types. The terms are denoted by M, N,... and defined by

$M ::= x \mid \lambda x.M \mid \langle M, M\rangle \mid \langle \exists, M\rangle \mid MM \mid M\pi_1 \mid M\pi_2 \mid M[x.M]$.

In the term $\lambda x.M$, the variable x is bound in M. In the term $N[x.M]$, the variable x is bound in M.

(2) The typing rules of $\mathsf{TF}\text{-}\lambda^{\neg\wedge\exists}$ are the same as those of $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$ except for ($\exists$ I) and ($\exists$ E). The rules ($\exists$ I) and ($\exists$ E) in this system are the following.

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

In the rule ($\exists$E), Γ_2 and C do not contain X freely.

The method for $\mathsf{M}\text{-}\lambda^{\neg\wedge\exists}$ by the CPS translation in the previous section can be adapted to $\mathsf{TF}\text{-}\lambda^{\neg\wedge\exists}$. By the method we can prove that the undecidability of TC and TI in $\mathsf{TF}\text{-}\lambda^{\neg\wedge\exists}$ is reduced to the undecidability in the type-free F. The proof is almost the same as that in the previous section, so here we give the definition of $\mathsf{TF}\text{-}F$ and the CPS translation only.

Definition 3.2 ($\mathsf{TF}\text{-}F$) (1) The types of $\mathsf{TF}\text{-}F$ are the $\to\forall$-types. The terms of $\mathsf{TF}\text{-}F$ are defined by

$M ::= x \mid \lambda x.M \mid \Lambda.M \mid MM \mid M\bullet$.

(2) The typing rules of $\mathsf{TF}\text{-}F$ are the same as those of $\mathsf{M}\text{-}F$ except for the following rules of universal quantifiers.

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

Definition 3.3 (CPS Translation) The *CPS translation* from the $\mathsf{TF}\text{-}F$-terms to the $\mathsf{TF}\text{-}\lambda^{\neg\wedge\exists}$-terms is defined by

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

$[\![\lambda x.M]\!] \equiv \lambda k.(\lambda x.[\![M]\!](k\pi_2))(k\pi_1)$,
$[\![MN]\!] \equiv \lambda k.[\![M]\!]\langle[\![N]\!], k\rangle$,
$[\![\Lambda.M]\!] \equiv \lambda k.k[k'.[\![M]\!]k']$,
$[\![M\bullet]\!] \equiv \lambda k.[\![M]\!]\langle\exists, k\rangle$.

Theorem 3.4 (1) Type checking in TF-F is reduced to type checking in TF-$\lambda^{\neg\wedge\exists}$. If type checking is undecidable in TF-F, then type checking is undecidable in TF-$\lambda^{\neg\wedge\exists}$.

(2) Type inference in TF-F is reduced to type inference in TF-$\lambda^{\neg\wedge\exists}$. If type inference is undecidable in TF-F, then type inference is undecidable in TF-$\lambda^{\neg\wedge\exists}$.

4 Concluding Remarks

In this paper, we proved the undecidability of type checking and type inference in both M-F and M-$\lambda^{\neg\wedge\exists}$. Moreover, we proved that the undecidability of the type checking and type inference in TF-$\lambda^{\neg\wedge\exists}$ is reduced to the undecidability of those in TF-F.

The decidability of type checking and type inference in TF-$\lambda^{\neg\wedge\exists}$ is an interesting question. As proved in this paper, their undecidability directly follows from the undecidability in TF-F. We cannot adapt the existing proofs to TF-F, although TF-F is similar to M-F and DF-F. For example, the proofs in Wells (1999) and this paper used the undecidability of the semi-unification problem. It is essential for this approach that the terms in the systems do not contain any information of the number of the quantifier rules. Fujita & Schubert (2009) showed the undecidability of TC and TI in the type-free lambda calculus with implication and existence by reducing the second-order unification problem to it. We expect the same idea can be adapted to show the undecidability of TC and TI for TF-F, which will show the undecidability of TC and TI for TF-$\lambda^{\neg\wedge\exists}$ with the result of our paper.

We have shown that the method in Nakazawa et al. (2008) can be used for the domain-free style calculi, the type-free style calculi, and the calculi with multiple-quantifier rules. It would be future work to generalize this method.

Acknowledgments The authors would like to thank Professor James Noble and the anonymous referees for their helpful comments to the preceding version (Nakazawa & Tatsuta (2009)). They would also like to thank Professor Ken-etsu Fujita and Professor Ryu Hasegawa for fruitful discussions.

References

Barthe, G., & Sørensen, M.H. (2000), Domain-free pure type systems, *J. Functional Programming* 10:412–452.

Fujita, K. & Schubert, A. (2000), Partially Typed Terms between Church-Style and Curry-Style, In *International Conference IFIP TCS 2000*, LNCS 1872, pp. 505–520.

Fujita, K. (2005), Galois embedding from polymorphic types in to existential types, In *Proceedings of 7th International Conference on Typed Lambda Calculi and Applications (TLCA 2005)*, LNCS 3461, pp. 194–208.

Fujita, K. & Schubert, A. (2009), Existential Type Systems with No Types in Terms, In *Proceedings of 9th International Conference on Typed Lambda Calculi and Applications (TLCA 2009)*, LNCS 5608, pp. 112–126.

Girard, J.Y. (1972), Interprétation Fonctionnelle et Élimination des Coupures de l'Arithmétique d'Ordre Supérieur, Thèse de doctorat d'Etat, Université de Paris VII.

Hasegawa, M. (2006), Relational parametricity and control, *Logical Methods in Computer Science*, 2(3:3):1–22.

Kfoury, A.J., Tiuryn, J. & Urzyczyn, P. (1993), The Undecidability of the Semi-unification Problem, *Information and Computation* 102:83–101.

Mitchell, J.C. & Plotkin, G.D. (1988), Abstract types have existential type, *ACM Transactions on Programming Languages and Systems* 10(3):470–502.

Nakazawa, K., Tatsuta, M., Kameyama, Y. & Nakano, H. (2008), Undecidability of Type-Checking in Domain-Free Typed Lambda-Calculi with Existence, In *the 17th EACSL Annual Conference on Computer Science Logic (CSL 2008)*, LNCS 5213, pp. 477–491.

Nakazawa, K. & Tatsuta, M. (2009), Type Checking and Inference for Polymorphic and Existential Types, In *the 15th Computing: The Australasian Theory Symposium (CATS 2009)*, CRPIT 94, pp. 61–69.

Reynolds, J.C. (1974), Towards a theory of type structure, In *Symposium on Programming*, LNCS 19, pp.408–425.

Schubert, A. (1998), Second-order unification and type inference for Church-style polymorphism, In *the 25th Annual ACM Symposium on Principles of Programming Languages (POPL '98)*, pp.279–288.

Tatsuta, M. (2007), Simple saturated sets for disjunction and second-order existential quantification, In *Proceedings of 8th International Conference on Typed Lambda Calculi and Applications (TLCA 2007)*, LNCS 4583, pp. 366–380.

Tatsuta, M., Fujita, K., Hasegawa, R. & Nakano, H. (2008), Inhabitation of Existential Types is Decidable in Negation-Product Fragment, In *Proceedings of 2nd International Workshop on Classical Logic and Computation (CLC2008)*.

Wells, J.B. (1999), Typability and Type Checking in System F Are Equivalent and Undecidable, In *Annals of Pure and Applied Logic* 98:111–156.