International Journal of Foundations of Computer Science
© World Scientific Publishing Company

TWO REALIZABILITY INTERPRETATIONS OF MONOTONE INDUCTIVE DEFINITIONS*

MAKOTO TATSUTA
Research Institute of Electrical Communication
Tohoku University
Sendai 980, JAPAN
e-mail: tatsuta@riec.tohoku.ac.jp

Received (received date)
Revised (revised date)
Communicated by Editor's name

ABSTRACT

Realizability of monotone inductive definitions is studied to construct a logical system in which recursive data structures and recursive control structures of programs are naturally formalized by using inductive definitions and programs can be produced from constructive proofs by using realizability. Two **q**-realizability interpretations of monotone inductive definitions are presented. One interpretation is defined so that a realizer of an inductively defined predicate is the same as a realizer of its expansion. It is proved that the interpretation of restricted monotone inductive definitions is sound and that the interpretation of full monotone inductive definitions is not sound, though the interpretation seems very natural. Another interpretation is presented by using the higher order coding and **q**-realizability for the second order logic. The soundness of this interpretation of full monotone inductive definitions is also proved. To prove this result, greatest-lower-bound inductive definitions are introduced.

1. Introduction

Recently inductive definitions in logical systems to formalize properties of programs have been studied widely.[1,2,3,4,5,6,7] Realizability has been used to synthesize programs in untyped theories.[5,6,7] We study **q**-realizability of monotone inductive definitions in this paper.

Inductive definitions play an important role in formalizing properties of programs in logical systems. Many important data structures such as lists and trees are naturally represented in logical systems by using inductive definitions, since they are recursively defined by nature. Loop structures, which are one of the most important control structures of programs, can be also naturally formalized by using inductive definitions, since they are realized by recursive calls. Specifications and

*This is an expanded version of "Monotone Recursive Definition of Predicates and Its Realizability Interpretation", *Proceedings of International Conference on Theoretical Aspects of Computer Software,* LNCS **526** (1991) pp. 38–52.

properties of programs can be formally represented in a natural way in a logical system by inductive definitions if the logical system has the facility of inductive definitions.

Realizability is a powerful mathematical tool to synthesize programs from constructive proofs. In other words, realizability can be used as a basis for constructive programming.[8,9] If a logical system is based on constructive logic and has a sound **q**-realizability interpretation, we can produce a program and its verification proof effectively from a proof of a specification formula of the program by using the interpretation. For example, PX[5] is a logical system actually implemented on computers in which properties of LISP programs are formalized and LISP programs can be produced by using realizability.

For untyped theories which have sound **q**-realizability, conditional inductive generation,[5] almost strictly positive inductive definitions[6] and positive inductive definitions[7] have been studied. Conditional inductive generation is included in almost strictly positive inductive definitions and almost strictly positive inductive definitions are included in positive inductive definitions. Stronger inductive definitions can represent more properties of programs in a more natural way.

It is natural to study monotone inductive definitions in this line, since they are more powerful than positive inductive definitions and the semantics of inductive definitions is given by the least fixed point that requires only monotonicity.

We study two **q**-realizability interpretation of monotone inductive definitions in this paper. One interpretation is an extension of interpretations of positive inductive definitions[6,7] and it is defined in such a way that a realizer of an inductively defined predicate is the same as a realizer of its expansion. The interpretation can give a sound **q**-realizability interpretation of restricted monotone inductive definitions, but is not sound as a **q**-realizability interpretation of full monotone inductive definitions. Another interpretation is defined by using the well-known higher order coding and **q**-realizability for the second order logic.[10] The interpretation gives a sound **q**-realizability interpretation of full monotone inductive definitions, but requires the framework of the second order logic.

We present untyped predicative theories $\mathbf{TID_0}$ and $\mathbf{TID_1}$ of partial combinatory terms, both of which have monotone inductive definitions of predicates. $\mathbf{TID_1}$ has full monotone inductive definitions, while $\mathbf{TID_0}$ has only restricted monotone inductive definitions. Full monotone inductive definitions give us a way of defining a predicate P which is the least solution of the equation $P \leftrightarrow A(P)$ where a formula $A(P)$ may include the predicate P and is monotone with respect to P. We define a **q**-realizability interpretation for $\mathbf{TID_0}$ and $\mathbf{TID_1}$ so that a realizer of an inductively defined predicate is the same as a realizer of its expansion. This **q**-realizability interpretation is essentially the same as the realizability for $\mathbf{EON} + \mu$.[6,7] We show that the realizability interpretation for $\mathbf{TID_0}$ in $\mathbf{TID_1}$ is sound and that the realizability interpretation for $\mathbf{TID_1}$ in $\mathbf{TID_1}$ is not sound. The soundness proof for $\mathbf{TID_0}$ entails and simplifies the soundness proof for a positive case of $\mathbf{EON} + \mu$[7] as a special case; the proof for $\mathbf{EON} + \mu$ seems much more complicated than that for $\mathbf{TID_0}$. The monotonicity condition for $\mathbf{TID_0}$ is simple and more general than

that of $\mathbf{EON} + \mu$. This enables us to have a simple proof for the soundness for $\mathbf{TID_0}$.

We also present an untyped theory $\mathbf{TID_2}$, which has greatest-lower-bound inductive definitions. $\mathbf{TID_2}$ includes $\mathbf{TID_1}$, since greatest-lower-bound inductive definitions cover monotone inductive definitions as a special case. We define a **q**-realizability interpretation for $\mathbf{TID_2}$, using the well-known higher order coding and **q**-realizability for the second order logic.[10] We prove its soundness. We show that the realizability gives a sound **q**-realizability interpretation for $\mathbf{TID_1}$.

Then we have two sound **q**-realizability interpretations of monotone inductive definitions: one is the interpretation for $\mathbf{TID_0}$ and the other is the interpretation for $\mathbf{TID_1}$ through $\mathbf{TID_2}$. The merits of the former are as follows: (1) a realizer of an inductively defined predicate is the same as a realizer of its expansion; (2) a realizer of an induction principle gives a loop structure to a program; (3) only predicative theories are used. The latter loses them all. On the other hand, the merit of the latter is that the interpretation covers full monotone inductive definitions. We can use both interpretations for program extraction.

In Section 2, we define the theories $\mathbf{TID_0}$ and $\mathbf{TID_1}$. In Section 3, we briefly explain how useful the facility of inductive definitions of predicates are and discuss related works. In Section 4, we discuss models of $\mathbf{TID_1}$ and prove its consistency. In Section 5, we present the **q**-realizability interpretation for $\mathbf{TID_0}$ and $\mathbf{TID_1}$. We explain that this realizability interpretation is natural. We prove the soundness theorem for $\mathbf{TID_0}$. We show that the interpretation for $\mathbf{TID_1}$ is not sound. In Section 6, we present a sound **q**-realizability interpretation of greatest-lower-bound inductive definitions, using the second order coding and **q**-realizability for the second order logic. We show that the interpretation provides a sound **q**-realizability interpretation for $\mathbf{TID_1}$.

This paper is an expanded version of the conference paper[11] in the sense that the discussions and results in Section 2 – Section 5 are based on the paper[11] and the results of Section 6 are new.

2. Theories $\mathbf{TID_0}$ and $\mathbf{TID_1}$

We present the theories $\mathbf{TID_0}$ and $\mathbf{TID_1}$ in this section. Both are the same as Beeson's **EON**[12] except for the rules of inductive definitions of predicates. $\mathbf{TID_0}$ and $\mathbf{TID_1}$ differ only in the rules of inductive definitions of predicates. The name **TID** comes from a Theory with Inductive Definitions.

2.1. Language of $\mathbf{TID_0}$ *and* $\mathbf{TID_1}$

The language of $\mathbf{TID_0}$ and that of $\mathbf{TID_1}$ are the same. This is based on a first order language but extended for inductive definitions of predicates.

The constants are:

$\mathbf{K}$, $\mathbf{S}$, $\mathbf{p}$, $\mathbf{p}_0$, $\mathbf{p}_1$, 0, $\mathbf{s}_{\mathrm{N}}$, $\mathbf{p}_{\mathrm{N}}$, $\mathbf{E}$.

We choose combinators as the target programming language for simplicity. **K** and **S** mean the usual basic combinators. We have natural numbers as primitives,

which are given by 0, the successor function $\mathbf{s}_{\mathrm{N}}$ and the predecessor function $\mathbf{p}_{\mathrm{N}}$. We also have the pairing functions $\mathbf{p}$, $\mathbf{p}_0$ and $\mathbf{p}_1$ as built-in, which correspond to `cons`, `car` and `cdr` in LISP respectively. $\mathbf{E}$ is a combinator judging equality of terms and corresponds to an if-then-else statement in a usual programming language.

We have only one function symbol:

App

whose arity is 2. This means a functional application of combinators.

Terms are defined in the same way as for usual first order logic. For terms s, t, we abbreviate $\mathbf{App}(s,t)$ as st. For terms s, t, we also use abbreviations $\langle s,t\rangle \equiv \mathbf{p}st$, $t_0 \equiv \mathbf{p}_0 t$ and $t_1 \equiv \mathbf{p}_1 t$.

We use a usual coding method of λ-terms by combinatory terms and use $\overline{t}$ to denote the combinatory term corresponding to a λ-term t.[13] In the sequel, a term t which may contain λ constructors always denotes a combinatory term $\overline{t}$. A substituted term $u[x := t]$ denotes a combinatory term $\overline{u}[x := \overline{t}]$ for terms t,u and a variable x.

The predicate symbols are:

$\bot, \quad \downarrow, \quad \mathbf{N}, \quad =$.

$\bot$ means contradiction. $\mathbf{N}(a)$ means that a is a natural number. $a = b$ means that a equals to b.

We adopt logic of partial terms, where some terms may fail to have any values.[12] $a\downarrow$ means that a has a value.

We have predicate variables, which a first order language does not have. The predicate variables are:

$X, Y, Z, \ldots, X^*, Y^*, Z^*, \ldots$.

Each predicate variable has fixed arity. We assume that the arity of X^* is $n+1$ if the arity of X is n.

Definition 1 (Formula). 1. If a, b are terms,

$\bot, \quad a\downarrow, \quad \mathbf{N}(a), \quad a = b$

are formulas. We call them atomic formulas.

2. If X is a predicate variable of arity n and $t_1, \ldots, t_n$ are terms, $X(t_1, \ldots, t_n)$ is a formula.

3. A formula is constructed from logical connectives $\&, \vee, \rightarrow, \forall, \exists$ in the same way as a first order language.

4. $(\lambda x_1 \ldots x_n.A)(t_1, \ldots, t_n)$ is a formula if A is a formula and $t_1, \ldots, t_n$ are terms.

5. $(\mu X.\lambda x_1 \ldots x_n.A)(t_1, \ldots, t_n)$ is a formula where X is a predicate variable of arity n, A is a formula and $t_1, \ldots, t_n$ are terms.

The last case corresponds to inductively defined predicates. Note that X and $x_1, \ldots, x_n$ may occur freely in A. The intuitive meaning of the formula

$(\mu X.\lambda x_1 \ldots x_n.A(X, x_1, \ldots, x_n))(t_1, \ldots, t_n)$

is as follows: Let P be a predicate of arity n such that P is the least solution of the equation

$P(x_1, \ldots, x_n) \leftrightarrow A(P, x_1, \ldots, x_n)$.

Then $(\mu X.\lambda x_1 \ldots x_n.A(X, x_1, \ldots, x_n))(t_1, \ldots, t_n)$ means $P(t_1, \ldots, t_n)$ intuitively.

We abbreviate a sequence as a bold type symbol, for example, $x_1, \ldots, x_n$ as $\mathbf{x}$. $\mathbf{t}\!\downarrow$ denotes $t_1\!\downarrow \,\&\, \ldots \,\&\, t_n\!\downarrow$ when $\mathbf{t} \equiv t_1 \ldots t_n$.

Example 1. We give an example of a formula. We assume the arity of a predicate variable P is 1. Then

$$(\mu P.\lambda x.x = 0 \vee \exists y(P(y) \,\&\, x = \mathbf{s}_{\mathrm{N}} y))(x)$$

is a formula.

Definition 2 (Abstract).

1. A predicate symbol of arity n is an abstract of arity n.
2. A predicate variable of arity n is an abstract of arity n.
3. If A is a formula, $\lambda x_1 \ldots x_n.A$ is an abstract of arity n.
4. If $(\mu X.\lambda x_1 \ldots x_n.A)(x_1, \ldots, x_n)$ is a formula, $\mu X.\lambda x_1 \ldots x_n.A$ is an abstract of arity n.

2.2. Basic Axioms and Inference Rules of $\mathbf{TID_0}$ *and* $\mathbf{TID_1}$

We give axioms and inference rules except those of inductive definitions. They are almost the same as $\mathbf{EON}^{12}$ and we only list them here.

Axioms for Partial Terms:

$$x\!\downarrow \qquad (x \text{ is a variable}), \tag{T1}$$

$$c\!\downarrow \qquad (c \text{ is a constant}), \tag{T2}$$

$$\mathbf{N}(t) \to t\!\downarrow \qquad (t \text{ is a term}), \tag{T3}$$

$$s = t \to s\!\downarrow \,\&\, t\!\downarrow \qquad (s, t \text{ are terms}), \tag{T4}$$

$$st\!\downarrow \to s\!\downarrow \,\&\, t\!\downarrow \qquad (s, t \text{ are terms}), \tag{T5}$$

$$F(\mathbf{t}) \to \mathbf{t}\!\downarrow. \tag{T6}$$

In (T6), F is an abstract of arity n and $\mathbf{t}$ is a sequence of n terms. $(s\!\downarrow \to s = t) \,\&\, (t\!\downarrow \to s = t)$ is abbreviated as $s \simeq t$.

The logical axioms and inference rules are the same as those of usual intuitionistic logic except for the following quantifier rules:

$$\frac{\forall x A(x) \quad t\!\downarrow}{A(t)}\ (\forall E), \qquad \frac{A(t) \quad t\!\downarrow}{\exists x A(x)}\ (\exists I).$$

Axioms for Equality:

$$x = x, \tag{E1}$$

$$x = y \,\&\, A(x) \to A(y). \tag{E2}$$

Axioms for Combinators:

$$\mathbf{K}xy = x, \tag{C1}$$

$$\mathbf{S}xy\!\downarrow, \tag{C2}$$

$$\mathbf{S}xyz \simeq xz(yz). \tag{C3}$$

Axioms for Pairing:

$\mathbf{p}_0(\mathbf{p}xy) = x$, (P1)

$\mathbf{p}_1(\mathbf{p}xy) = y$, (P2)

$\mathbf{p}_0 x\downarrow$, (P3)

$\mathbf{p}_1 x\downarrow$. (P4)

Axioms for Natural Numbers:

$\mathbf{N}(0)$, (N1)

$\mathbf{N}(x) \to \mathbf{N}(\mathbf{s}_\mathrm{N} x)$, (N2)

$\mathbf{N}(x) \to \mathbf{p}_\mathrm{N}(\mathbf{s}_\mathrm{N} x) = x$, (N3)

$\mathbf{N}(x) \to \mathbf{s}_\mathrm{N} x \neq 0$, (N4)

$A(0) \,\&\, \forall x(\mathbf{N}(x) \,\&\, A(x) \to A(\mathbf{s}_\mathrm{N} x)) \to \forall x(\mathbf{N}(x) \to A(x))$. (N5)

Decidable Equality:

$x = y \vee x \neq y$, (D1)

$x = y \to \mathbf{E}xyzw = z$, (D2)

$x \neq y \to \mathbf{E}xyzw = w$, (D3)

$s \neq t$ (s and t are closed normal terms and $s \not\equiv t$). (D4)

Axioms for Abstracts:

$(\lambda \mathbf{x}.A)(\mathbf{x}) \leftrightarrow A$. (A1)

2.3. Inference Rules of Inductive Definitions

We discuss inference rules of inductive definitions of predicates here. Let $\mu \equiv \mu P.\lambda \mathbf{x}.A(P)$ where $\mathbf{x}$ is a sequence of variables whose length is the same as the arity of a predicate variable P and $A(P)$ is a formula displaying all the free occurrences of P in a formula A. Suppose that

$(\mu 1) \quad \equiv \quad \forall \mathbf{x}(A(\mu) \to \mu(\mathbf{x}))$,

$(\mu 2) \quad \equiv \quad \forall \mathbf{x}(A(C) \to C(\mathbf{x})) \to \forall \mathbf{x}(\mu(\mathbf{x}) \to C(\mathbf{x}))$,

$(\mathrm{MONO}) \equiv \forall \mathbf{x}(X(\mathbf{x}) \to Y(\mathbf{x})) \to \forall \mathbf{x}(A(X) \to A(Y))$,

$(\mathrm{MONO\text{-}Q}) \equiv \forall \mathbf{x} r(X^*(r, \mathbf{x}) \to Y^*(r, \mathbf{x})) \to$

$\forall \mathbf{x} r((r \quad \mathbf{q}_X[X, X^*] \quad A(X)) \to (r \quad \mathbf{q}_X[X, Y^*] \quad A(X)))$

where X, Y, X^*, Y^* are predicate variables, $A(X)$ is a formula displaying all the free occurrences of a predicate variable X in a formula A, $C(\mathbf{x})$ is a formula displaying all the free occurrences of variables $\mathbf{x}$ in a formula C and $\mathbf{x}$ covers all the free variables of $A(X)$. The notation $\mathbf{q}_X[X, X^*]$ is defined in Section 5.

The condition (MONO) states that the formula $A(X)$ is monotone with respect to X.

The additional condition (MONO-Q) states that the formula $(e \quad \mathbf{q}_X[X, X^*] \quad A(X))$ is monotone with respect to X^*.

The condition (MONO-Q) is important, since it guarantees that $(\mu 1)$ and $(\mu 2)$ can be applied to the interpretation of $A(X)$. The condition (MONO-Q) is indispensable for the soundness of the interpretation, which is showed later.

We have 2 inference rules for $\mathbf{TID_0}$ and $\mathbf{TID_1}$ respectively. For $\mathbf{TID_0}$ the inference rules are as follows:

$$\frac{\text{(MONO)} \quad \text{(MONO-Q)}}{(\mu 1)} \ (\mu R1')$$

$$\frac{\text{(MONO)} \quad \text{(MONO-Q)}}{(\mu 2)} \ (\mu R2')$$

where X, Y, X^*, Y^* do not occur freely in open assumptions of deductions of the premises.

For $\mathbf{TID_1}$ the inference rules are as follows:

$$\frac{\text{(MONO)}}{(\mu 1)} \ (\mu R1)$$

$$\frac{\text{(MONO)}}{(\mu 2)} \ (\mu R2)$$

where X, Y do not occur freely in open assumptions of a deduction of the premise.

Rules $(\mu R1)$ and $(\mu R2)$ of $\mathbf{TID_1}$ do not have the additional condition (MONO-Q) as premises, since by comparing $\mathbf{TID_0}$ and $\mathbf{TID_1}$ we will show that the condition (MONO-Q) brings the soundness of the realizability.

We define a theory $\mathbf{TID}^-$ as the theory whose language is the same as $\mathbf{TID_1}$ and whose axioms and inference rules are only basic axioms and inference rules given in Subsection 2.2. That is, the theory $\mathbf{TID}^-$ is the theory $\mathbf{TID_1}$ except for the two inference rules $(\mu R1)$ and $(\mu R2)$. $\mathbf{TID}^-$ is almost the same as Beeson's **EON**.

3. Monotone Inductive Definitions of Predicates

We will explain how useful the inductive definitions of predicates are, and we will discuss related works.

Let $\mu \equiv \mu P.\lambda \mathbf{x}.A(P)$ where $\mathbf{x}$ is a sequence of variables whose length is the same as the arity of a predicate variable P and $A(P)$ is a formula displaying all the free occurrences of P in a formula A. Suppose that (MONO) and (MONO-Q) hold for the formula A. So we can use $(\mu 1)$ and $(\mu 2)$ for A in each theory $\mathbf{TID_0}$ and $\mathbf{TID_1}$. We will discuss $(\mu 1)$ and $(\mu 2)$.

By $(\mu 1)$ and $(\mu 2)$,

$$\mu(\mathbf{x}) \leftrightarrow A(\mu) \qquad (\mu 1')$$

holds. This μ is the least solution of the equation for a predicate variable X:

$$X(\mathbf{x}) \leftrightarrow A(X).$$

Because

$$\forall \mathbf{x}(A(C) \leftrightarrow C(\mathbf{x})) \rightarrow \forall \mathbf{x}(\mu(\mathbf{x}) \rightarrow C(\mathbf{x}))$$

holds by ($\mu 2$) for any formula $C(\mathbf{x})$.

We often want to define a predicate inductively. For example, we will define the predicate *Nat* which judges whether the argument is a natural number or not. One way to define it is as follows:

$$\mathit{Nat}(x) \equiv x = 0 \vee \exists y(\mathit{Nat}(y) \,\&\, x = \mathbf{s}_{\mathrm{N}} y).$$

But it is not a definition in a usual sense since *Nat* appears also in the right hand side which is the body of the definition of the predicate. What we want to present by this equation is the least solution of the recursive equation for a predicate variable X:

$$X(x) \leftrightarrow x = 0 \vee \exists y(X(y) \,\&\, x = \mathbf{s}_{\mathrm{N}} y).$$

In $\mathbf{TID_0}$ and $\mathbf{TID_1}$ this solution can be represented by the formula

$$(\mu P.\lambda x.x = 0 \vee \exists y(P(y) \,\&\, x = \mathbf{s}_{\mathrm{N}} y))(x)$$

and ($\mu 1$) and ($\mu 2$) state that this formula is actually the least solution of the equation. By this facility we can define *Nat* in $\mathbf{TID_0}$ and $\mathbf{TID_1}$ as follows:

$$\mathit{Nat}(x) \equiv (\mu P.\lambda x.x = 0 \vee \exists y(P(y) \,\&\, x = \mathbf{s}_{\mathrm{N}} y))(x).$$

For this example, ($\mu 2$) represents the mathematical induction for natural numbers classified by *Nat*. ($\mu 2$) represents the generalized induction principle in general.

By the facility of inductive definitions of predicates, many useful recursive data structures such as natural numbers, lists and trees can be defined inside the theory.[6] The induction principle corresponding to each data structure is derived from ($\mu 2$) for each case. The list induction is derived from ($\mu 2$) for a definition of lists and the tree induction is derived from ($\mu 2$) for a definition of trees.

The theory is introduced to write specification formulas of programs in a natural setting. Inductively defined predicates are also useful to specify recursive calls, which are one of the most important control structures of programs. For example, predicates defined recursively by Horn clauses in Prolog can be represented directly in the theory.[6]

Recursive definitions or inductive definitions have been studied in untyped predicative theories.[5,6,7] The restricting condition for $A(P)$ varies for the theories. In **PX**,[5] we can use $(\mu P.\lambda x.A(P))(t)$ only when every occurrence of P in $A(P)$ is positive and $A(P)$ is a Harrop formula. In $\mathbf{EON} + \mu$,[6] we can use it only when every occurrence of P in $A(P)$ is positive and for each occurrence P one of the following 2 conditions holds:

1. the occurrence of P appears in some Harrop subformula of $A(P)$;
2. the occurrence of P is strictly positive in $A(P)$.

In $\mathbf{EON} + \mu$,[7] we can use it only when every occurrence of P in $A(P)$ is positive. In $\mathbf{TID_0}$ we have presented, we can use it only when both of the following 2 conditions hold:

1. (MONO), which means that $A(P)$ is monotone with respect to P;
2. (MONO-Q), which means that $(r \;\mathbf{q}\; A(P))$ is monotone with respect to P^*.

$\mathbf{TID_0}$ has the weakest restrictions on $A(P)$ among the above theories. We will show it in Proposition 6 in Section 5.

We want to have weaker restriction on inductively defined predicates. But the theory will be inconsistent if there are no restrictions, because without any restrictions we could use $\mu P.\neg P$ and $(\mu P.\neg P) \leftrightarrow \neg(\mu P.\neg P)$ would hold if we require $(\mu 1')$. From the point of view of an intended model, an inductively defined predicate, which is the least solution of an equation, means the least fixed point. In this model construction we need only monotonicity. Therefore we may raise a question: can we use inductively defined predicates on the condition that a formula $A(P)$ is monotone with respect to a predicate variable P? From the point of view of consistency of the theory, the answer is yes. In Section 4 we establish Theorem 2 to state that $\mathbf{TID_1}$ is consistent. But the restrictions of **PX** and $\mathbf{EON} + \mu$ were introduced also for the soundness of the realizability. So we may raise another question: is realizability of monotone inductive definitions sound? The answer varies according to the interpretations. In Section 5 we show that the answer is no for our realizability interpretation defined there. In Section 5 we establish Theorem 3 to state that the realizability interpretation for $\mathbf{TID_1}$ is not sound. We relax the restriction on inductive definitions in $\mathbf{TID_1}$, which is weaker than positivity, but we need some restrictions that come from the realizability interpretation. $\mathbf{TID_0}$ suffices this purpose since the realizability interpretation for $\mathbf{TID_0}$ is sound. On the other hand, the answer is yes for another realizability interpretation defined in Section 6, where the **q**-realizability interpretation for $\mathbf{TID_1}$ is sound.

4. Models of $\mathbf{TID_1}$

We will briefly explain semantics of $\mathbf{TID_1}$ by giving its intended model. Since $\mathbf{TID_1}$ includes $\mathbf{TID_0}$, this model also explains semantics of $\mathbf{TID_0}$.

We will use the well-known least fixed point theorem for model construction.

Least Fixed Point Theorem. Let S be a set and $p(S)$ be the power set of S. If $f : p(S) \to p(S)$ is a monotone function, there exists a such that $a \in p(S)$ and

1. $f(a) = a$,
2. For any $b \in p(S)$, if $f(b) \subset b$, then $a \subset b$.

a is abbreviated as $\mathrm{lfp}(f)$.

We will construct a model of $\mathbf{TID_1}$ extending an arbitrary model of $\mathbf{TID}^-$. Our intended model of $\mathbf{TID}^-$ is the closed normal term model whose universe is the set of closed normal terms and in which equality of elements is decidable. This model construction of $\mathbf{TID}^-$ is based on Beeson's method.[12] We denote the universe by U.

We will define $\rho \models A$ in almost the same way as for first order logic, where A is a formula and ρ is an environment which assigns an element of U to a first order variable and a subset of U^n to a predicate variable of arity n and which covers all the free first order variables and all the free predicate variables of A. We present only the definition for the case $P(\mathbf{t})$ for a predicate variable P and the case $(\mu P.\lambda \mathbf{x}.A(P))(\mathbf{t})$.

Case $P(\mathbf{t})$.

$\rho \models P(\mathbf{t})$ is defined as $\mathbf{t} \in \rho(P)$.

Case $(\mu P.\lambda \mathbf{x}.A(P))(\mathbf{t})$.

Define F as follows:

$F : p(U^n) \to p(U^n)$,

$F(X) = \{\mathbf{x} \in U^n \mid \rho[P := X] \models A(P)\}$,

where $|\mathbf{x}| = n$ and $\rho[P := X]$ is defined as follows:

$\rho[P := X](P) = X$.

$\rho[P := X](x) = \rho(x)$, if x is not P.

Then $\rho \models (\mu P.\lambda \mathbf{x}.A(P))(\mathbf{t})$ is defined as $\mathbf{t} \in \mathrm{lfp}(F)$ if F is monotone. $\rho \models (\mu P.\lambda \mathbf{x}.A(P))(\mathbf{t})$ is defined as true if F is not monotone.

Theorem 1. If $\mathbf{TID_1} \vdash A$ holds, then $\rho \models A$ holds for any environment ρ which covers all the free variables of A.

Proof. The claim is proved by induction on the proof of A. Cases are classified by the last rule applied to the proof. We give only two cases here, since other cases are the same as a usual proof for the first order logic.

Case $(\mu R1)$.

We must show $\rho \models \forall \mathbf{x}(A(\mu) \to \mu(\mathbf{x}))$. By induction hypothesis, we have $\rho \models \forall \mathbf{x}(X(\mathbf{x}) \to Y(\mathbf{x})) \to \forall \mathbf{x}(A(X) \to A(Y))$. Define $F(X) = \{\mathbf{x} \in U^n \mid \rho[P := X] \models A(P)\}$. Then F is monotone. Fix $\mathbf{t} \in U^n$ and assume $\rho \models A(\mu)[\mathbf{x} := \mathbf{t}]$. By this, we have $\rho[P := \mathrm{lfp}(F)] \models A(P)[\mathbf{x} := \mathbf{t}]$. Hence $\mathbf{t} \in F(\mathrm{lfp}(F))$. Therefore $\mathbf{t} \in \mathrm{lfp}(F)$. Then we have $\rho \models \mu(\mathbf{t})$. This shows the claim for this case.

Case $(\mu R2)$.

We must show $\rho \models \forall \mathbf{x}(A(C) \to C(\mathbf{x})) \to \forall \mathbf{x}(\mu(\mathbf{x}) \to C(\mathbf{x}))$. By induction hypothesis, we have $\rho \models \forall \mathbf{x}(X(\mathbf{x}) \to Y(\mathbf{x})) \to \forall \mathbf{x}(A(X) \to A(Y))$. Define $F(X) = \{\mathbf{x} \in U^n \mid \rho[P := X] \models A(P)\}$. Then F is monotone. Assume $\rho \models \forall \mathbf{x}(A(C) \to C(\mathbf{x}))$.

Assume $\mathbf{t} \in F(\{\mathbf{x} \in U^n \mid \rho \models C(\mathbf{x})\})$. Then $\rho \models A(C)[\mathbf{x} := \mathbf{t}]$. Hence $\rho \models C(\mathbf{t})$. Therefore $\mathbf{t} \in \{\mathbf{x} \in U^n \mid \rho \models C(\mathbf{x})\}$. Hence we have $F(\{\mathbf{x} \in U^n \mid \rho \models C(\mathbf{x})\}) \subset \{\mathbf{x} \in U^n \mid \rho \models C(\mathbf{x})\}$ and $\mathrm{lfp}(F) \subset \{\mathbf{x} \in U^n \mid \rho \models C(\mathbf{x})\}$.

We will show $\rho \models \forall \mathbf{x}(\mu(\mathbf{x}) \to C(\mathbf{x}))$. Fix $\mathbf{t} \in U^n$ and assume $\rho \models \mu(\mathbf{t})$. Then $\mathbf{t} \in \mathrm{lfp}(F)$. Therefore $\mathbf{t} \in \{\mathbf{x} \in U^n \mid \rho \models C(\mathbf{x})\}$. That is, $\rho \models C(\mathbf{t})$ holds. This shows the claim of this case. □

Theorem 2. $\mathbf{TID_0}$ and $\mathbf{TID_1}$ are consistent.

Proof. Assume $\vdash \bot$ in $\mathbf{TID_1}$. Then the previous theorem shows that $\models \bot$. This is a contradiction. Therefore $\vdash \bot$ does not hold in $\mathbf{TID_1}$.

The consistency of $\mathbf{TID_0}$ is immediate from the consistency of $\mathbf{TID_1}$, since $\mathbf{TID_0} \vdash A$ implies $\mathbf{TID_1} \vdash A$ for any formula A. □

5. q-Realizability Interpretation for $\mathbf{TID_0}$ and $\mathbf{TID_1}$

We give an intuitive explanation of our realizability. We start with usual **q**-realizability and try to interpret $(\mu P.\lambda x.A(P,x))(x)$. Let μ be $\mu P.\lambda x.A(P,x)$ and suppose that $(\mu 1)$ and $(\mu 2)$ hold for this μ. Then $\mu(x) \leftrightarrow A(\mu,x)$ holds. We want to treat $\mu(x)$ and $A(\mu,x)$ in the same manner. So we require $(e \ \mathbf{q} \ \mu(x)) \leftrightarrow$

$(e \quad \mathbf{q} \quad A(\mu, x))$. Therefore it is natural to define $(e \quad \mathbf{q} \quad \mu(x))$ as $\mu^*(e, x)$ where $\mu^*(e, x)$ is the least solution of the recursive equation for a predicate variable X^*:

$$X^*(e, x) \leftrightarrow (e \quad \mathbf{q} \quad A(\mu, x))[(r \quad \mathbf{q} \quad \mu(y)) := X^*(r, y)].$$

where $[(r \quad \mathbf{q} \quad \mu(y)) := X^*(r, y)]$ of the right hand side means replacing each subformula $(r \quad \mathbf{q} \quad \mu(y))$ by a subformula $X^*(r, y)$ in the formula $(e \quad \mathbf{q} \quad A(\mu, x))$. We get the following definition of our realizability by describing this idea in an exact way. This interpretation is an immediate extension of interpretations of positive inductive definitions.[6,7]

Our **q**-realizability interpretation for $\mathbf{TID_0}$ and that for $\mathbf{TID_1}$ are the same, since the language of $\mathbf{TID_0}$ and that of $\mathbf{TID_1}$ are also the same.

Definition 3 (q-Realizability Interpretation). Suppose A is a formula, $n \geq 0$, $P_1, \ldots, P_n$ is a sequence of predicate variables whose arities are $m_1, \ldots, m_n$ respectively and $F_1, G_1, \ldots, F_n, G_n$ is a sequence of abstracts whose arities are $m_1, m_1 + 1, \ldots, m_n, m_n + 1$ respectively. A formula

$$(e \quad \mathbf{q}_{P_1,\ldots,P_n}[F_1, G_1, \ldots, F_n, G_n] \quad A)$$

is defined by induction on the construction of A as follows.

We use the following abbreviations.

$(e \quad \mathbf{q}' \quad A) \equiv (e \quad \mathbf{q}_{P_1,\ldots,P_n}[F_1, G_1, \ldots, F_n, G_n] \quad A)$,

$(e \quad \mathbf{q}'_P[F, G] \quad A) \equiv (e \quad \mathbf{q}_{P_1,\ldots,P_n,P}[F_1, G_1, \ldots, F_n, G_n, F, G] \quad)$,

$\mathbf{F} \equiv F_1, \ldots, F_n$,

$\mathbf{P} \equiv P_1, \ldots, P_n$.

1. $(e \quad \mathbf{q}' \quad A) \equiv e = 0 \,\&\, A$ where A is an atomic formula.
2. $(e \quad \mathbf{q}' \quad P_i(\mathbf{t})) \equiv G_i(e, \mathbf{t})$.
3. $(e \quad \mathbf{q}' \quad Q(\mathbf{t})) \equiv Q^*(e, \mathbf{t})$ where $Q \not\equiv P_i \quad (1 \leq i \leq n)$.
4. $(e \quad \mathbf{q}' \quad A \,\&\, B) \equiv (e_0 \quad \mathbf{q}' \quad A) \,\&\, (e_1 \quad \mathbf{q}' \quad B)$.
5. $(e \quad \mathbf{q}' \quad A \vee B) \equiv \mathbf{N}(e_0) \,\&$
   $(e_0 = 0 \rightarrow (e_1 \quad \mathbf{q}' \quad A) \,\&\, A_{\mathbf{P}}[\mathbf{F}]) \,\&$
   $(e_0 \neq 0 \rightarrow (e_1 \quad \mathbf{q}' \quad B) \,\&\, B_{\mathbf{P}}[\mathbf{F}])$.
6. $(e \quad \mathbf{q}' \quad A \rightarrow B) \equiv \forall q((q \quad \mathbf{q}' \quad A) \,\&\, A_{\mathbf{P}}[\mathbf{F}] \rightarrow (eq \quad \mathbf{q}' \quad B)) \,\&\, e{\downarrow}$.
7. $(e \quad \mathbf{q}' \quad \forall x A(x)) \equiv \forall x(ex \quad \mathbf{q}' \quad A(x))$.
8. $(e \quad \mathbf{q}' \quad \exists x A(x)) \equiv (e_1 \quad \mathbf{q}' \quad A(e_0)) \,\&\, A(e_0)_{\mathbf{P}}[\mathbf{F}]$.
9. $(e \quad \mathbf{q}' \quad (\lambda \mathbf{x}.A)(\mathbf{t})) \equiv (\lambda r \mathbf{x}.(r \quad \mathbf{q}' \quad A))(e, \mathbf{t})$.
10. $(e \quad \mathbf{q}' \quad \mu(\mathbf{t})) \equiv (\mu X^*.\lambda r \mathbf{x}.(r \quad \mathbf{q}'_X[\mu_{\mathbf{P}}[\mathbf{F}], X^*] \quad A(X)))(e, \mathbf{t})$ where $\mu \equiv \mu X.\lambda \mathbf{x}.A(X)$.

We write simply $(e \quad \mathbf{q} \quad A)$ if $n = 0$.

In the above definition, $_{P_1,\ldots,P_n}[F_1, G_1, \ldots, F_n, G_n]$ means a substitution. Our realizability interpretation is something like a realizability interpretation with substitution.

We will show that this interpretation works well for $\mathbf{TID_0}$ but goes wrong for $\mathbf{TID_1}$. That is, it will be shown that the interpretation for $\mathbf{TID_0}$ is sound but the interpretation for $\mathbf{TID_1}$ is not sound. The difference is brought by the additional condition (MONO-Q). In general, we cannot conclude that $(e \quad \mathbf{q}_X[X, X^*] \quad A)$ is monotone with respect to X^* even if A is monotone with respect to X. For such A, we in general cannot use $(\mu 1)$ and $(\mu 2)$ for $(e \quad \mathbf{q} \quad (\mu X.\lambda x.A)(t))$ in $\mathbf{TID_1}$. On the other hand, we can always use $(\mu 1)$ and $(\mu 2)$ for $(e \quad \mathbf{q} \quad (\mu X.\lambda x.A)(t))$ in $\mathbf{TID_0}$ since the additional (MONO-Q) guarantees that $(e \quad \mathbf{q}_X[X, X^*] \quad A)$ is monotone with respect to X^*.

Theorem 3. The $\mathbf{q}$-realizability interpretation is not sound for $\mathbf{TID_1}$. That is, there is a formula A such that $\mathbf{TID_1} \vdash A$ holds but $\mathbf{TID_1} \vdash (e \quad \mathbf{q} \quad A)$ does not hold for any e.

Proof. Let X,Y be predicate variables of arity 0 and

$\mu \equiv \mu X.(X \to X) \,\&\, Y$,

$\mu^*(r) \equiv (r \quad \mathbf{q} \quad \mu)$,

$A \equiv \mu \to Y$.

Then clearly $\mathbf{TID_1} \vdash A$ holds. But $\mathbf{TID_1} \vdash (e \quad \mathbf{q} \quad A)$ does not hold for any e. We will show it by the model given in Section 4. By Definition 3 We have

$$\begin{aligned}(e \quad \mathbf{q} \quad A) &\equiv (e \quad \mathbf{q} \quad \mu \to Y) \\ &\equiv \forall r((r \quad \mathbf{q} \quad \mu) \,\&\, \mu \to (er \quad \mathbf{q} \quad Y)) \,\&\, e\!\downarrow \\ &\equiv \forall r(\mu^*(r) \,\&\, \mu \to Y^*(er)) \,\&\, e\!\downarrow,\end{aligned}$$

where

$$\begin{aligned}\mu^* &\equiv \mu X^*.\lambda r.(r \quad \mathbf{q}_X[\mu, X^*] \quad (X \to X) \,\&\, Y) \\ &\equiv \mu X^*.\lambda r.(r_0 \quad \mathbf{q}_X[\mu, X^*] \quad X \to X) \,\&\, (r_1 \quad \mathbf{q}_X[\mu, X^*] \quad Y) \\ &\equiv \mu X^*.\lambda r.\forall q((q \quad \mathbf{q}_X[\mu, X^*] \quad X) \,\&\, \mu \to (r_0 q \quad \mathbf{q}_X[\mu, X^*] \quad X)) \\ &\qquad \,\&\, r_0\!\downarrow \,\&\, Y^*(r_1) \\ &\equiv \mu X^*.\lambda r.\forall q(X^*(q) \,\&\, \mu \to X^*(r_0 q)) \,\&\, r_0\!\downarrow \,\&\, Y^*(r_1).\end{aligned}$$

Choose ρ such that $\rho \models Y$ and $\rho[Y^*] = \phi$. Define F as follows:

$F : p(U) \to p(U)$,

$F(V) = \{r \in U \mid \rho[X^* := V] \models \forall q(X^*(q) \,\&\, \mu \to X^*(r_0 q)) \,\&\, r_0\!\downarrow \,\&\, Y^*(r_1)\}$.

Since F is not monotone, $\rho' \models \mu^*(r)$ holds for any $u \in U$ and $\rho' = \rho[r := u]$. Put $u = 0$ and $\rho' = \rho[r := 0]$. Then we have

$\rho' \models \mu^*(r)$,

$\rho' \models \mu$,

$\rho' \not\models Y^*(er)$

for any e. Therefore we have

$\rho' \not\models \mu^*(r) \,\&\, \mu \to Y^*(er)$.

Hence

$\rho \not\models \forall r(\mu^*(r) \,\&\, \mu \to Y(er))$.

Therefore $\mathbf{TID_1} \not\vdash (e \quad \mathbf{q} \quad A)$ for any e. □

We will discuss the realizability interpretation for $\mathbf{TID_0}$ in $\mathbf{TID_1}$.

Proposition 1.

$\mathbf{TID_1} \vdash (e \quad \mathbf{q} \quad A) \to e{\downarrow}$

holds.

Proof. By induction on the construction of a formula A. □

Proposition 2. For a formula $A(X)$ and an abstract C,

$(e \quad \mathbf{q} \quad A(X))[X := C, X^* := \lambda r\mathbf{x}.(r \quad \mathbf{q} \quad C(\mathbf{x}))] \equiv (e \quad \mathbf{q} \quad A(C))$

holds, where $[X := C, X^* := \lambda r\mathbf{x}.(r \quad \mathbf{q} \quad C(\mathbf{x}))]$ denotes a substitution of abstracts for predicate variables.

This claim is proved by induction on A.

Proposition 3. Let $\mu \equiv \mu P.\lambda\mathbf{x}.A(P)$ and suppose that (MONO-Q) holds for $A(P)$. Then

$\mathbf{TID_1} \vdash (e \quad \mathbf{q} \quad \mu(\mathbf{x})) \leftrightarrow (e \quad \mathbf{q} \quad A(\mu))$

holds.

This proposition shows that the definition of the realizability satisfies our first motivation.

Proof. From (μ1) and (μ2). □

We give realizers of (μ1) and (μ2) in the next two propositions.

Proposition 4. Suppose that $\mathbf{TID_0} \vdash$ (MONO) and $\mathbf{TID_0} \vdash$ (MONO-Q) hold for a formula $A(X)$. Let $\mu \equiv \mu X.\lambda\mathbf{x}.A(X)$. Then

$\mathbf{TID_1} \vdash \lambda\mathbf{x}r.r \quad \mathbf{q} \quad (\mu 1)$

holds.

Proof. Immediate from Proposition 3. □

Proposition 5.

$\mathbf{TID_1} \vdash \lambda q.\mu f.\lambda\mathbf{x}r.q\mathbf{x}(mf\mathbf{x}(m(\lambda\mathbf{x}r.r)\mathbf{x}r)) \quad \mathbf{q} \quad (\mu 2)$

holds where

$\mathbf{TID_1} \vdash m \quad \mathbf{q} \quad$ (MONO).

We prove it by using the next lemma.

Suppose that

$m \quad \mathbf{q} \quad \forall\mathbf{x}(X(\mathbf{x}) \to Y(\mathbf{x})) \to \forall\mathbf{x}(A(X) \to A(Y))$.

By expanding it, we get

$\forall f(\forall\mathbf{x}r(X^*(r,\mathbf{x}) \,\&\, X(\mathbf{x}) \to Y^*(f\mathbf{x}r,\mathbf{x})) \,\&\, \forall\mathbf{x}(X(\mathbf{x}) \to Y(\mathbf{x})) \,\&\, \forall\mathbf{x}(f\mathbf{x}{\downarrow})$
$\to \forall\mathbf{x}r((r \quad \mathbf{q}_X[X, X^*] \quad A(X)) \,\&\, A(X) \to (mf\mathbf{x}r \quad \mathbf{q}_X[Y, Y^*] \quad A(X)))).$ (1)

Lemma 1. Suppose that

$m \quad \mathbf{q} \quad \forall\mathbf{x}(X(\mathbf{x}) \to Y(\mathbf{x})) \to \forall\mathbf{x}(A(X) \to A(Y))$.

Then

$(r \quad \mathbf{q}_X[F, G] \quad A(X)) \,\&\, A(F) \to$
$(m(\lambda\mathbf{x}r.r)\mathbf{x}r \quad \mathbf{q}_X[F, \lambda r\mathbf{x}.(F(\mathbf{x}) \,\&\, G(r,\mathbf{x}))] \quad A(X))$

holds.

Proof. Put

$X := F$,

$Y := F$,

$X^* := G$,

$Y^* := \lambda r\mathbf{x}.(F(\mathbf{x}) \,\&\, G(r,\mathbf{x}))$,

$f := \lambda \mathbf{x} r.r$

in (1). Then we get the conclusion of the lemma. □

Proof. (Proof of Proposition 5) We assume that

$$q \quad \mathbf{q} \quad \forall \mathbf{x}(A(C) \to C(\mathbf{x})), \tag{2}$$

$$\forall \mathbf{x}(A(C) \to C(\mathbf{x})). \tag{3}$$

Let

$f \equiv \mu f.\lambda \mathbf{x} r.q\mathbf{x}(mf\mathbf{x}(m(\lambda \mathbf{x} r.r)\mathbf{x}r))$.

We will show

$f \quad \mathbf{q} \quad \forall \mathbf{x}(\mu(\mathbf{x}) \to C(\mathbf{x}))$

under these conditions. This is equivalent to

$\forall \mathbf{x} r((\mu P^*.\lambda r\mathbf{x}.(r \quad \mathbf{q}_P[\mu, P^*] \quad A(P)))(r,\mathbf{x}) \,\&\, \mu(\mathbf{x}) \to (f\mathbf{x}r \quad \mathbf{q} \quad C(\mathbf{x})))$.

Therefore by ($\mu 2$) it is sufficient to show

$\forall \mathbf{x} r((r \quad \mathbf{q}_P[\mu, \lambda r\mathbf{x}.(\mu(\mathbf{x}) \to (f\mathbf{x}r \quad \mathbf{q} \quad C(\mathbf{x})))] \quad A(P)) \to$

$\mu(\mathbf{x}) \to (f\mathbf{x}r \quad \mathbf{q} \quad C(\mathbf{x})))$.

From Lemma 1 and (MONO-Q) it is sufficient to show

$\forall \mathbf{x} r((m(\lambda \mathbf{x} r.r)\mathbf{x}r \quad \mathbf{q}_P[\mu, \lambda r\mathbf{x}.(f\mathbf{x}r \quad \mathbf{q} \quad C(\mathbf{x}))] \quad A(P)) \,\&\, A(\mu) \to$

$(f\mathbf{x}r \quad \mathbf{q} \quad C(\mathbf{x})))$. (4)

Put

$X := \mu$,

$Y := C$,

$X^* := \lambda r\mathbf{x}.(f\mathbf{x}r \quad \mathbf{q} \quad C(\mathbf{x}))$,

$Y^* := \lambda r\mathbf{x}.(r \quad \mathbf{q} \quad C(\mathbf{x}))$,

$f := f$

in (1), then we get

$\forall \mathbf{x} r((r \quad \mathbf{q}_X[\mu, \lambda r\mathbf{x}.(f\mathbf{x}r \quad \mathbf{q} \quad C(\mathbf{x}))] \quad A(X)) \,\&\, A(\mu) \to$

$(mf\mathbf{x}r \quad \mathbf{q} \quad A(C)))$. (5)

We will show (4). Fix $\mathbf{x}$ and r and assume that

$$m(\lambda \mathbf{x} r.r)\mathbf{x}r \quad \mathbf{q}_P[\mu, \lambda r\mathbf{x}.(f\mathbf{x}r \quad \mathbf{q} \quad C(\mathbf{x}))] \quad A(P), \tag{6}$$

$$A(\mu). \tag{7}$$

We will show $(f\mathbf{x}r \quad \mathbf{q} \quad C(\mathbf{x}))$ under these conditions. By (5), (6) and (7), we have

$mf\mathbf{x}(m(\lambda \mathbf{x} r.r)\mathbf{x}r) \quad \mathbf{q} \quad A(C)$.

By (2),

$q\mathbf{x}(mf\mathbf{x}(m(\lambda \mathbf{x} r.r)\mathbf{x}r)) \quad \mathbf{q} \quad C(\mathbf{x})$.

By the definition of f, we have

$f\mathbf{x}r \quad \mathbf{q} \quad C(\mathbf{x})$.

Then (4) holds. □

Theorem 4 (Soundness Theorem). If $\mathbf{TID_0} \vdash A$, we can get a term e from a proof and $\mathbf{TID_1} \vdash (e \ \ \mathbf{q} \ \ A)$ holds where all the free variables of e are included in all the free variables of A.

Proof. By induction on the proof of $\mathbf{TID_0} \vdash A$. Cases are classified by the last rule applied to the proof. Cases for basic axioms and inference rules are proved in the same way as the paper.[6] Case $(\mu R1')$ is proved by using Proposition 4. Case $(\mu R2')$ is proved by using Proposition 5. □

By the soundness of the realizability, we get the term existence property, the disjunction property, the program extraction theorem and consistency with some choice axioms in a usual way.

Theorem 5 (Term Existence Property). If $\mathbf{TID_0} \vdash \exists x A(x)$, we can get a term t from the proof and $\mathbf{TID_1} \vdash A(t)$ holds.

Proof. Suppose $\mathbf{TID_0} \vdash \exists x A(x)$. By Theorem 4, we can get a term e from the proof and $\mathbf{TID_1} \vdash (e \ \ \mathbf{q} \ \ \exists x A(x))$. By Definition 3, it implies $\mathbf{TID_1} \vdash A(e_0)$. Put $t := e_0$, then the claim holds. □

Theorem 6 (Program Extraction). If there exists a term j such that $\mathbf{TID_1} \vdash \forall x(A(x) \to (j(x) \ \ \mathbf{q} \ \ A(x)))$ and $\mathbf{TID_0} \vdash \forall x(A(x) \to \exists y B(x,y))$, we can get a program f from the proof and $\mathbf{TID_1} \vdash \forall x(A(x) \to f(x)\downarrow \,\&\, B(x, f(x)))$ holds.

Proof. Suppose that there exists a term j such that $\mathbf{TID_1} \vdash \forall x(A(x) \to (j(x) \ \ \mathbf{q} \ \ A(x)))$ and $\mathbf{TID_0} \vdash \forall x(A(x) \to \exists y B(x,y))$. By Theorem 4, we can get a term e from the proof and $\mathbf{TID_1} \vdash (e \ \ \mathbf{q} \ \ \forall x(A(x) \to \exists y B(x,y)))$. By Definition 3 and Proposition 1, it implies that $\mathbf{TID_1} \vdash \forall x(A(x) \to (ex(jx))_0\downarrow \,\&\, B(x, (ex(jx))_0))$. Put $f := \lambda x.(ex(jx))_0$, then the claim holds. □

We show the discussion of the realizability given in the papers[6,7] is a special case of the above discussion.

Proposition 6. If P occurs only positively in $A(P)$, (MONO) and (MONO-Q) hold.

Corollary 1. $\mathbf{TID_0}$ is an extension of $\mathbf{EON} + \mu$ where $\mathbf{EON} + \mu$ is presented in the paper.[6,7]

If we choose an appropriate monotone realizer m for $A(P)$ where P is positive in $A(P)$, we get almost the same soundness proof for $\mathbf{EON} + \mu$ as in the paper.[6,7] But the soundness proof of this paper is simpler than those for $\mathbf{EON} + \mu$ in the paper[6,7] because of the monotonicity condition.

We explain it more by comparing the proof of this paper and that of the paper.[6] The main difference of two proofs is in the proof of the case of the generalized induction. The proof of the case of the paper[6] is 7 pages long, which is written from Proposition 4.3.4 to Proof 4.3.15, while that of this paper is only 2 pages long, which is written in Proposition 5 and Lemma 1. This simplicity comes from the fact that we pay attention to the realizer m of the monotonicity (MONO) in this paper, while in the paper[6] we calculate concretely $\sigma^{X,f}_{A(X)}$, which corresponds to the realizer $mf\mathbf{x}$, when the formula $A(X)$ is positive with respect to X. The main property of σ stated by Proposition 4.3.12, and the soundness proof of the case of the generalized induction stated by Proposition 4.3.14 in the paper[6] correspond respectively to the formula (5) of the proof of Proposition 5, and Proposition 5 in this paper.

6. Theory $\mathbf{TID_2}$ and Its q-Realizability

In this section, we present another **q**-realizability interpretation. This gives a sound interpretation for $\mathbf{TID_1}$, which has full monotone inductive definitions. This interpretation is based on the well-known higher order coding and **q**-realizability for the second order logic.[10] The interpretation is defined for the theory $\mathbf{TID_2}$, which has greatest-lower-bound inductive definitions and includes $\mathbf{TID_1}$.

Definition 4 (Theory $\mathbf{TID_2}$). The language of the theory $\mathbf{TID_2}$ is the same as that of $\mathbf{TID_0}$, $\mathbf{TID_1}$ and $\mathbf{TID}^-$.

The axioms and inference rules of $\mathbf{TID_2}$ are those of $\mathbf{TID}^-$ and the following inference rule ($\mu 3$) and axiom ($\mu 2$):

$$\frac{\forall \mathbf{x}(A(X) \to X(\mathbf{x})) \to \forall \mathbf{x}(C(\mathbf{x}) \to X(\mathbf{x}))}{\forall \mathbf{x}(C(\mathbf{x}) \to \mu(\mathbf{x}))}\ (\mu 3),$$

$$\forall \mathbf{x}(A(C) \to C(\mathbf{x})) \to \forall \mathbf{x}(\mu(\mathbf{x}) \to C(\mathbf{x})) \qquad (\mu 2)$$

where $\mu \equiv \mu X.\lambda \mathbf{x}.A(X)$ and in the rule ($\mu 3$) the predicate variable X does not occur freely in formulas $C(\mathbf{x})$ and $A(C)$ and open assumptions of a deduction of the premise.

The intended meaning of $\mu(\mathbf{x})$ is the greatest lower bound of the set $\{X | \forall \mathbf{x}(A(X) \to X(\mathbf{x}))\}$ with respect to the order relation $\sqsubseteq$ given by the implication connective $\to$. We call the definitions of $\mu(\mathbf{x})$ by ($\mu 3$) and ($\mu 2$) *greatest-lower-bound inductive definitions*. Note that monotonicity of $A(X)$ is not required here.

Theorem 7. $\mathbf{TID_2}$ is consistent.

Proof. We will construct a model of $\mathbf{TID_2}$ extending an arbitrary model of $\mathbf{TID}^-$. Our intended model of $\mathbf{TID}^-$ is the closed normal term model whose universe is the set of closed normal terms in the same way as that of $\mathbf{TID_1}$. We denote the universe by U.

We will define $\rho \models A$ in almost the same way as for first order logic where A is a formula and ρ is an environment which assigns an element of U to a first order variable and a subset of U^n to a predicate variable of arity n and which covers all the free first order variables and all the free predicate variables of A. We present only the definition for the case $\mu(\mathbf{t})$ where $\mu \equiv \mu P.\lambda \mathbf{x}.A(P)$.

We define $\rho \models \mu(\mathbf{t})$ as

$\mathbf{t} \in \cap\{S \in p(U^n) | \rho[X := S] \models \forall \mathbf{x}(A(X) \to X(\mathbf{x}))\}$

where $\rho[X := S]$ is defined as follows:

$\rho[X := S](X) = S$;

$\rho[X := S](x) = \rho(x)$, if x is not X.

Then ($\mu 3$) and ($\mu 2$) hold in this model. □

We use a theory $\mathbf{T_2}$ to give **q**-realizability for $\mathbf{TID_2}$.

Definition 5 (Theory $\mathbf{T_2}$). A theory $\mathbf{T_2}$ is defined as the theory obtained from $\mathbf{TID_1}$ by adding the second order universal quantifiers.

If A is a formula and X is a predicate variable, then $\forall X\, A$ is a formula.

The axioms and inference rules of $\mathbf{T_2}$ are those of $\mathbf{TID_1}$ and the following inference rules:

$$\frac{A(X)}{\forall X A(X)}\ (\forall^2 I), \qquad \frac{\forall X A(X)}{A(F)}\ (\forall^2 E)$$

where $A(X)$ denotes a formula displaying all the free occurrences of a predicate variable X, F denotes an abstract of the same arity as the predicate variable X and in the rule $(\forall^2 I)$ X does not occur freely in open assumptions of a deduction of the premise.

Theorem 8. $\mathbf{T_2}$ is consistent.

Proof. A theory $\mathbf{T_2}'$ is defined as the theory obtained from $\mathbf{T_2}$ by removing the formula constructor μ.

$\mathbf{T_2}'$ is clearly consistent by the well-known fact of consistency of the second order logic and the models of $\mathbf{TID}^-$.

It is sufficient to give an interpretation for $\mathbf{T_2}$ in $\mathbf{T_2}'$. An interpretation A° of the formula A is defined by induction on the construction of A as follows:

1. if A is either an atomic formula or $X(\mathbf{t})$, then $A^\circ \equiv A$;
2. $(A \,\&\, B)^\circ \equiv A^\circ \,\&\, B^\circ$;
3. $(A \to B)^\circ \equiv A^\circ \to B^\circ$;
4. $(A \vee B)^\circ \equiv A^\circ \vee B^\circ$;
5. $(\forall x\, A)^\circ \equiv \forall x(A^\circ)$;
6. $(\exists x\, A)^\circ \equiv \exists x(A^\circ)$;
7. $((\lambda \mathbf{x}.B)(\mathbf{t}))^\circ \equiv (\lambda \mathbf{x}.B^\circ)(\mathbf{t})$;
8. $((\mu X.\mathbf{x}.A(X))(\mathbf{t}))^\circ \equiv \forall X(\forall \mathbf{x}(A^\circ(X) \to X(\mathbf{x})) \to X(\mathbf{t}))$;
9. $(\forall X\, A)^\circ \equiv \forall X(A^\circ)$.

If a formula A is proved under assumptions $A_1, \ldots, A_n$ in $\mathbf{T_2}$, then the formula A° is proved under assumptions $A_1^\circ, \ldots, A_n^\circ$ in $\mathbf{T_2}'$. This claim is proved by induction on a proof of a formula A. □

Proposition 7. If a formula $A(X)$ is monotone with respect to a predicate variable X, that is,

$$\forall \mathbf{x}(X(\mathbf{x}) \to Y(\mathbf{x})) \to \forall \mathbf{x}(A(X) \to A(Y))$$

holds in $\mathbf{TID_2}$, then $\mathbf{TID_2}$ proves

$$\forall \mathbf{x}(A(\mu) \to \mu(\mathbf{x})) \qquad (\mu 1)$$

where $\mu \equiv \mu X.\lambda \mathbf{x}.A(X)$.

This proposition shows that greatest-lower-bound inductive definitions cover monotone inductive definitions.

Proof. Assume $\forall \mathbf{x}(A(X) \to X(\mathbf{x}))$. By $(\mu 2)$, $\forall \mathbf{x}(\mu(\mathbf{x}) \to X(\mathbf{x}))$ holds. By monotonicity of $A(X)$, $\forall \mathbf{x}(A(\mu) \to A(X))$ holds. Combining it with the assumption, we have $\forall \mathbf{x}(A(\mu) \to X(\mathbf{x}))$. Hence we have $\forall \mathbf{x}(A(X) \to X(\mathbf{x})) \to \forall \mathbf{x}(A(\mu) \to X(\mathbf{x}))$ without assumptions. By $(\mu 3)$, $\forall \mathbf{x}(A(\mu) \to \mu(\mathbf{x}))$ holds. □

Theorem 9. If a formula A is provable in $\mathbf{TID_1}$, then A is provable in $\mathbf{TID_2}$.

This claim is proved immediately by Proposition 7.

We give $\mathbf{q}$-realizability for $\mathbf{TID_2}$ in $\mathbf{T_2}$.

Definition 6 (q-Realizability for $\mathbf{TID_2}$). A $\mathbf{q}$-realizability interpretation $(e \quad \mathbf{q} \quad A)$ of the formula A is defined by induction on the construction of A as follows:

1. $(e \quad \mathbf{q} \quad A) \equiv e = 0 \,\&\, A$ where A is an atomic formula.
2. $(e \quad \mathbf{q} \quad X(\mathbf{t})) \equiv X(\mathbf{t}) \,\&\, X^*(e, \mathbf{t})$ where X is a predicate variable.
3. $(e \quad \mathbf{q} \quad A \,\&\, B) \equiv (e_0 \quad \mathbf{q} \quad A) \,\&\, (e_1 \quad \mathbf{q} \quad B)$.
4. $(e \quad \mathbf{q} \quad A \vee B) \equiv \mathbf{N}(e_0) \,\&\, (e_0 = 0 \to (e_1 \quad \mathbf{q} \quad A)) \,\&\, (e_0 \neq 0 \to (e_1 \quad \mathbf{q} \quad B))$.
5. $(e \quad \mathbf{q} \quad A \to B) \equiv \forall q((q \quad \mathbf{q} \quad A) \to (eq \quad \mathbf{q} \quad B)) \,\&\, (A \to B) \,\&\, e{\downarrow}$.
6. $(e \quad \mathbf{q} \quad \forall x A(x)) \equiv \forall x(ex \quad \mathbf{q} \quad A(x))$.
7. $(e \quad \mathbf{q} \quad \exists x A(x)) \equiv (e_1 \quad \mathbf{q} \quad A(e_0))$.
8. $(e \quad \mathbf{q} \quad (\lambda \mathbf{x}.A)(\mathbf{t})) \equiv (\lambda r \mathbf{x}.(r \quad \mathbf{q} \quad A))(e, \mathbf{t})$.
9. $(e \quad \mathbf{q} \quad \mu(\mathbf{t})) \equiv \forall X X^* r((r \quad \mathbf{q} \quad \forall \mathbf{x}(A(X) \to X(\mathbf{x}))) \to X^*(er, \mathbf{t})) \,\&\, \mu(\mathbf{t}) \,\&\, e{\downarrow}$ where $\mu \equiv \mu X.\lambda \mathbf{x}.A(X)$.

Proposition 8. $(e \quad \mathbf{q} \quad A) \to A$ holds in $\mathbf{T_2}$.

This claim is proved by induction on the construction of A.

We give realizers of $(\mu 3)$ and $(\mu 2)$ in the next proposition.

Proposition 9. Suppose that $\mu \equiv \mu X.\lambda \mathbf{x}.A(X)$.

1. A rule
$$\frac{q \quad \mathbf{q} \quad \forall \mathbf{x}(A(X) \to X(\mathbf{x})) \to \forall \mathbf{x}(C(\mathbf{x}) \to X(\mathbf{x}))}{\lambda \mathbf{x} r p.qp\mathbf{x}r \quad \mathbf{q} \quad \forall \mathbf{x}(C(\mathbf{x}) \to \mu(\mathbf{x}))}$$
is derivable in $\mathbf{T_2}$ where X and X^* do not occur freely in open assumptions of a deduction of the premise.

2. $(\lambda q \mathbf{x} r.rq \quad \mathbf{q} \quad \forall \mathbf{x}(A(C) \to C(\mathbf{x})) \to \forall \mathbf{x}(\mu(\mathbf{x}) \to C(\mathbf{x})))$ holds in $\mathbf{T_2}$.

Proof.

1. Assume
$$q \quad \mathbf{q} \quad \forall \mathbf{x}(A(X) \to X(\mathbf{x})) \to \forall \mathbf{x}(C(\mathbf{x}) \to X(\mathbf{x})). \tag{8}$$
We will show
$$\forall \mathbf{x} r((r \quad \mathbf{q} \quad C(\mathbf{x})) \to (\lambda p.qp\mathbf{x}r \quad \mathbf{q} \quad \mu(\mathbf{x})) \,\& \tag{9}$$
$$\forall \mathbf{x}(C(\mathbf{x}) \to \mu(\mathbf{x})) \,\&\, \lambda r p.qp\mathbf{x}r{\downarrow}. \tag{10}$$
(10) is proved from (8), Proposition 8 and $(\mu 3)$.

We will show (9). Fix $\mathbf{x}$ and r and assume
$$(r \quad \mathbf{q} \quad C(\mathbf{x})). \tag{11}$$

Under these assumptions, we will show

$$\forall X X^* p((p \quad \mathbf{q} \quad \forall \mathbf{x}(A(X) \to X(\mathbf{x}))) \to X^*(qp\mathbf{x}r, \mathbf{x})) \,\&\, \tag{12}$$

$$\mu(\mathbf{x}) \,\&\, \lambda p.qp\mathbf{x}r{\downarrow}. \tag{13}$$

(13) is proved from (11), Proposition 8 and (10).

We will show (12). Fix X, X^* and p and assume $(p \quad \mathbf{q} \quad \forall \mathbf{x}(A(X) \to X(\mathbf{x})))$.

From (8), $(qp \quad \mathbf{q} \quad \forall \mathbf{x}(C(\mathbf{x}) \to X(\mathbf{x})))$ holds. From (11), $(qp\mathbf{x}r \quad \mathbf{q} \quad X(\mathbf{x}))$ holds. □

2. We will show

$$\forall q((q \quad \mathbf{q} \quad \forall \mathbf{x}(A(C) \to C(\mathbf{x}))) \to (\lambda \mathbf{x}r.rq \quad \mathbf{q} \quad \forall \mathbf{x}(\mu(\mathbf{x}) \to C(\mathbf{x})))) \,\&\, \tag{14}$$

$$\forall \mathbf{x}(A(C) \to C(\mathbf{x})) \to \forall \mathbf{x}(\mu(\mathbf{x}) \to C(\mathbf{x})) \,\&\, \lambda q\mathbf{x}r.rq{\downarrow}. \tag{15}$$

(15) is proved by ($\mu 2$).

Fix q and assume

$$q \quad \mathbf{q} \quad \forall \mathbf{x}(A(C) \to C(\mathbf{x})). \tag{16}$$

We will show

$$\forall \mathbf{x}r((r \quad \mathbf{q} \quad \mu(\mathbf{x})) \to (rq \quad \mathbf{q} \quad C(\mathbf{x}))) \,\& \tag{17}$$

$$\forall \mathbf{x}(\mu(\mathbf{x}) \to C(\mathbf{x})) \,\&\, \lambda r.rq{\downarrow}. \tag{18}$$

(18) is proved from (16), Proposition 8 and ($\mu 2$).

We will show (17). Fix $\mathbf{x}$ and r and assume $(r \quad \mathbf{q} \quad \mu(\mathbf{x}))$. This assumption is identical to

$$\forall X X^* q((q \quad \mathbf{q} \quad \forall \mathbf{x}(A(X) \to X(\mathbf{x}))) \to X^*(rq, \mathbf{x})) \,\&\, \mu(\mathbf{x}) \,\&\, r{\downarrow}. \tag{19}$$

We will show $(rq \quad \mathbf{q} \quad C(\mathbf{x}))$. Put $X := C$, $X^* := \lambda r\mathbf{x}.(r \quad \mathbf{q} \quad C(\mathbf{x}))$ and $q := q$ in (19). From (16), we have $(rq \quad \mathbf{q} \quad C(\mathbf{x}))$. □

Theorem 10 (Soundness of the Realizability for $\mathbf{TID_2}$). If a formula A is provable in $\mathbf{TID_2}$, then we can get a term e from the proof and $(e \quad \mathbf{q} \quad A)$ holds in $\mathbf{T_2}$.

This theorem is proved by induction on the proof of A by using Proposition 9.

We give a realizer of ($\mu 1$) in the next proposition.

Proposition 10. If m is a realizer of monotonicity of $A(X)$, that is,

$$m \quad \mathbf{q} \quad \forall \mathbf{x}(X(\mathbf{x}) \to Y(\mathbf{x})) \to \forall \mathbf{x}(A(X) \to A(Y))$$

holds in $\mathbf{T_2}$, then

$$\lambda \mathbf{x}rp.p\mathbf{x}(m(\lambda \mathbf{x}r.rp)\mathbf{x}r) \quad \mathbf{q} \quad \forall \mathbf{x}(A(\mu) \to \mu(\mathbf{x}))$$

holds in $\mathbf{T_2}$ where $\mu \equiv \mu X.\lambda \mathbf{x}.A(X)$.

This claim is proved by calculating the realizer according to the proof of Proposition 7.

Theorem 11 (Soundness of the Realizability for $\mathbf{TID_1}$). If a formula A is provable in $\mathbf{TID_1}$, then we can get a term e from the proof and $(e \quad \mathbf{q} \quad A)$ holds in $\mathbf{T_2}$.

This theorem is proved by induction on the proof of A by using Theorem 10 and Proposition 10.

By the soundness of the realizability, we get the term existence property, the disjunction property, the program extraction theorem and consistency with some choice axioms in a usual way.

Theorem 12. The program extraction theorem and the term existence property hold for $\mathbf{TID_1}$ and $\mathbf{TID_2}$ in $\mathbf{T_2}$.

Remark 1. The realizability in this section presents a sound interpretation of full monotone inductive definitions. On the other hand, the realizability given in Section 5 can only provide an interpretation of restricted monotone inductive definitions.

Remark 2. Neither the definition of the **q**-realizability for $\mathbf{TID_2}$ nor the proof of its soundness given in this section uses monotonicity. On the other hand, both the definition of the **q**-realizability for $\mathbf{TID_0}$ and the proof of its soundness given in Section 5 need monotonicity.

Neither the realizers of ($\mu 3$) and ($\mu 2$) given in Proposition 9 nor the realizer of ($\mu 1$) given in Proposition 10 has loop structures. It is realizers of $\mu(x)$ that provide loop structures to extracted programs.

Both the consistency proof of $\mathbf{T_2}$ and the soundness proof of the realizability for $\mathbf{TID_2}$ use the second order logic. On the other hand, both the consistency proof of $\mathbf{TID_1}$ and the soundness proof of the realizability for $\mathbf{TID_0}$ use only the fixed point theorem and do not use the second order logic.

7. Concluding Remarks

We have presented two **q**-realizability interpretations of monotone inductive definitions. One is the interpretation given in Section 5, which satisfies that a realizer of an inductively defined predicate is the same as a realizer of its expansion. The other is the interpretation given in Section 6, which is based on the higher order coding and realizability for the second order logic. The former can provide only a sound interpretation of restricted monotone inductive definitions but is not involved in the second order system. On the other hand, the latter can present a sound interpretation of full monotone inductive definitions but is involved in the second order system.

It will be interesting to study the following problems on monotone inductive definitions and **q**-realizability interpretations introduced in this paper. (1) Which interpretation is fit better to program extraction? For this purpose, we need to analyze practical examples of program extraction using the interpretations. (2) Find new application of monotone inductive definitions to programming. Up to now only application of positive inductive definitions to programming is known. We could find new notions in programming by using monotone inductive definitions. (3) Study proof-theoretic strength of $\mathbf{TID_2}$. We conjecture that its strength would be equal to that of $\mathbf{TID_1}$.

Acknowledgements

I'm deeply grateful to Prof. Masahiko Sato for invaluable discussions. I would like to thank Prof. Takayasu Ito for useful comments.

References

1. R.L. Constable et al, *Implementing Mathematics with the Nuprl Proof Development System* (Prentice-Hall, 1986).
2. T. Coquand and C. Paulin, "Inductively Defined Types", LNCS **417**, 1990.
3. C. Paulin-Mohring, "Extracting F_ω's programs from proofs in the Calculus of Constructions", *Proc.* 16^{th} *Symp. Principles of Programming Languages,* 1989, pp.89–104.
4. F. Pfenning and C. Paulin-Mohring, "Inductively Defined Types in the Calculus of Constructions", Technical Report CMU-CS-89-209, School of Computer Science, Carnegie Melon University, 1989.
5. S. Hayashi and H. Nakano, **PX***: A Computational Logic* (MIT Press, 1988).
6. M. Tatsuta, "Program Synthesis Using Realizability", *Theoretical Computer Science* **90** (1991) 309–353.
7. S. Kobayashi and M. Tatsuta, "Realizability Interpretation of Generalized Inductive Definitions", *Theoretical Computer Science* (to appear).
8. M. Sato, and Y. Kameyama, "Constructive Programming in $\mathcal{SST}$", *Proceedings of the Japanese-Czechoslovak Seminar on Theoretical Foundations of Knowledge Information Processing,* 1990, pp. 23–30.
9. M. Tatsuta, "Realizability of Inductive Definitions for Constructive Programming", Doctoral Thesis, University of Tokyo, 1993.
10. H. Friedman, "On the derivability of instantiation properties", *Journal of Symbolic Logic* **42** (4) (1977) 506–514.
11. M. Tatsuta, "Monotone Recursive Definition of Predicates and Its Realizability Interpretation", *Proceedings of International Conference on Theoretical Aspects of Computer Software,* LNCS **526** (1991) pp. 38–52.
12. M. Beeson, *Foundations of Constructive Mathematics* (Springer, 1985).
13. H.P. Barendregt, *The Lambda Calculus: Its Syntax and Semantics* (North-Holland, Amsterdam, 1984).