データベース・システム 80−13
人 工 知 能 73−13
（1990. 11. 9）

ラベルつきグラフに基づくオブジェクトの意味論

安川 秀樹, 横田 一正
(財) 新世代コンピュータ技術開発機構

本稿の目的は, オブジェクト指向の文脈でオブジェクト識別性として使われる複合オブジェクトを表現するための新しいアプローチを提案し、その宣言的意味論をラベルつきグラフの定義域上に与えることである。

$\mathcal{QUIXOTE}$ [8] は、本稿で提案する複合オブジェクトに基づいた知識表現言語であり、部分情報の表現、部分情報のマージ (単一化)、属性の継承などの、知識表現と推論機能を持っている。

われわれの方法論は、ZFC$^-$/AFA、つまり P. Aczel によって提案された超集合論 [1] に基づいている。それは、オブジェクトの無限構造に対しても、通常の集合論的性質を満たすものである。

Labeled Graphs as Semantics of Objects

Hideki YASUKAWA, Kazumasa YOKOTA
Institute for New Generation Computer Technology (ICOT)
21F., Mita-Kokusai Bldg., 1-4-28, Mita, Minato-ku, Tokyo 108, JAPAN
e-mail: yasukawa@icot.or.jp, kyokota@icot.or.jp

The purposes of the present paper is to propose a new approach to representing complex objects which are used as **object identity** in the context of object-orientation, and give a declarative part of its semantics on the domain of **labeled graphs**.

$\mathcal{QUIXOTE}$ [8] is a knowledge representation language based on the complex objects proposed here, and provides knowledge representation and inference services, including representation of partial information, merging (unification) of partial information, **inheritance of properties**, and so on.

Our methodology is ZFC$^-$/AFA, a hyperset theory proposed by P. Aczel [1] which brings to bear all of the familiar set-theoretic technique to deal with circular phenomena.

1 Introduction

Complex data structure is used for various data and knowledge representation in many application: complex objects in a database area, feature structure in natural language processing, (recursive) record structure in programming languages, and frames and taxonomies in artificial intelligence. Even if different terms are used in those areas, it has been known that there are many similarities among them: partial or extended term representation of objects instead of a first order predicate based one, a subsumption relation among them, an inheritance mechanism based on the relation, and identification of an object. There have been many works focusing on them, where boundaries among database, programming, and knowledge representation languages, and among languages for various applications become to be disappearing gradually.

We have proposed a knowledge representation language, called $\mathcal{QUIXOTE}$, ancestors of which are *Juan* for deductive and object-oriented databases and QUINT for natural language processing applications, and is designed along the above line. In $\mathcal{QUIXOTE}$, an object consists of the identifier and the properties, each attribute of which is a triple of a label, an operator, and a value. We construct it uniformly in the form of **complex objects**, possibly containing circular structure. Our complex objects correspond to ones in a database area, and are considered also as extended feature structure in a natural language area. An object identifier corresponds also to concept description. A subsumption relation is defined among them and used for property inheritance. Furthermore, a concept of a module, the identifier is also in the form of complex objects, is introduced and used for classification of objects and rule inheritance.

One of the distinguished features of $\mathcal{QUIXOTE}$ is the semantics, which is based on ZFC$^-$/AFA, a hyperset theory proposed by P. Aczel [1]. In this paper, we focus mainly on the semantics, the domain of which are a set of labeled graphs in the sense of ZFC$^-$/AFA for treating circular structure. In Section 2, we informally explain objects in $\mathcal{QUIXOTE}$. Then, we define a labeled graph in Section 3, subsumption relation over labeled graphs in Section 4, and discuss about solvability of constraints over them in Section 5. Concerning on the discussions in Section 5, Barwise[2] and Mukai[5],[4] shows important results on constraints over hypersets. Actually, the part on constraints over labeled graphs in the present paper much owes to Mukai's work. As for the context, the related results are explained in Appendix A. Furthermore, we explain property inheritance among objects in Section 6, and a concept of a module and rule inheritance among modules in Section 7.

2 Objects of $\mathcal{QUIXOTE}$

In this section, objects in $\mathcal{QUIXOTE}$ are briefly explained.

Suppose we have a finite set BO of **primitive objects** and a lattice $BO^* = (BO, \preceq, \top, \bot)$. For any $a, b \in BO$, $a \preceq b$ means that b subsumes a, that is, a is a b. $\top$ and $\bot$ be the supremum and infimum of BO^*, respectively. Also, let $a \sqcap b$ and $a \sqcup b$ be the infimum and the supremum of a and b, respectively. Elements of BO^* are called **basic objects**.

An example of the lattice is $BO^* = (\{animal, mammal, human, dog\}, \preceq, \top, \bot)$ where the following holds:

$$mammal \preceq animal$$
$$human \preceq mammal$$
$$dog \preceq mammal$$

Also suppose that a finite set L of (atomic) **labels**. The objects in $\mathcal{QUIXOTE}$ (called ground **object terms**) are defined as follows:

Let o be a basic object, $l_1, l_2, \ldots$ be labels, $o_1, o_2, \ldots$ be object terms, respectively.

- Every basic object is an object.
- A term $o[l_1 = o_1, l_2 = o_2, \ldots]$ is an object term if it contains only one value specification for each label.
- A term is an object only if it can be shown to be an object by the above definition.

For an object term $o[l_1 = o_1, l_2 = o_2, \ldots]$, we say that o be the **head** and o_i be the l_i-**value** of that object term. A head can be omitted only when it is $\top$.

Let $BO = \{human, 20, 30, int, male, female\}$, $20 \preceq int, 30 \preceq int$, and $L = \{age, sex\}$. Then, the following terms are object terms in $\mathcal{QUIXOTE}$:

$$human,$$
$$human[age = 20, sex = male],$$
$$[age = 20].$$

Contrary, the following terms are not object terms:

$$human[age = 20][sex = male],$$
$$human[age = 20, age = 30].$$

An object term is uniquely pictured by a graph such that each node has a basic object as its value, and each arc is associated with a label.

For example, the object term $human[age = 20]$ is pictured by the graph consisting of the pair of the set $\{n_1, n_2\}$ of nodes and the set $\{n_1 \xrightarrow{age} n_2\}$ where the value $n_1 \downarrow$ and $n_2 \downarrow$ of the nodes $n_1$and n_2 are $human$ and 20, respectively.

It is possible to have object terms containing variables ranging over ground object terms as follows:

$$human[age = X, sex = Y].$$

Notice that $X[age = 20]$ is not an object term, since variables are not ranging over basic objects.

Let χ be a collection of variables, and $\mathcal{T}$ be a collection of ground object terms, $\mathcal{T}[\chi]$ be a collection of object terms possibly containing variables.

The subsumption relation is a binary relation over $\mathcal{T} \times \mathcal{T}$ (written $\sqsubseteq$). Though the precise definition of subsumption relation is given in Section 4, intuitive understanding of $\sqsubseteq$-relation will suffice at this point. Intuitively, $o_1 \sqsubseteq o_2$ (we say o_2 subsumes o_1) holds if o_1 has more arcs than o_2 and the value of a node of o_2 is larger than the value of the corresponding node of o_1 with respect to $\preceq$-ordering.

For example, the object term *animal* subsumes $human[age = 20, sex = male]$, since the latter has more arcs than the former, and the value (*animal*) of the root node of the former is larger than the value (*human*) of the root node of the latter, that is, $human \preceq animal$ holds. Similarly, $human[age = 20] \sqsubseteq animal[age = int]$ holds, but $human[age = 20]$ and $human[sex = male]$ can not be compared with respect to $\sqsubseteq$. Moreover, the object term $\top$ be the largest among all the object terms.

The congruence relation (written $\cong$) is defined as follows:

$$o_1 \cong o_2 \stackrel{\text{def}}{=} o_1 \sqsubseteq o_2 \wedge o_2 \sqsubseteq o_1$$

As a collection of object terms is partially ordered by $\sqsubseteq$, $\cong$ is just an equality. But, in $\mathcal{QUIXOTE}$, a basic object sometimes be defined in terms of an object term as:

$$20\_aged \cong [age = 20].$$

This makes $\cong$ be only an equivalence relation. Thus, we are actually working on the quotient $\mathcal{T}/\cong$, on which $\sqsubseteq$ is partial order on $\mathcal{T}/\cong$.

Let u, v be object terms. An **atomic constraint** is a literal if it is in the one of the following forms:

$$u \sqsubseteq v$$
$$u \cong v.$$

Without loss of generality, we can assume that at least u or v be a basic object or a variable. A **constraint** is a set of atomic constraints. A constraint is understood as the conjunction of its elements, that is, the conjunction of atomic constraints in it. Thus, our constraint language is a sublanguage of a quantifier-free first order language in which only conjunction is allowed as logical connectives.

By using constraints, it is possible to extend objects to contain variables ranging over some subset of $\mathcal{T}$. For example, the followings are the object terms in $\mathcal{QUIXOTE}$ [1]:

$$human[affiliation = X] \mid \{X \sqsubseteq company\},$$
$$human[hobby = X] \mid \{tennis \sqsubseteq X\},$$
$$human[age = X] \mid \{X \cong 20\}.$$

Notice that the last one is just same as $human[age = 20]$. Sometimes, we need to have a self-referential object such as "a person who employs himself". Such an object can be defined by using a constraint as follows:

$$X \mid \{X \cong person[employee = X]\}.$$

[1]The *hobby*-value of the second object may seem to be a set. Actually, in $\mathcal{QUIXOTE}$, sets are introduced, but it is beyond the scope of this paper. The details of the treatment of object terms including sets will be shown in [8]. For the moment, assume we have special objects like *tennis_and_ski* or *ball_games*.

The following is the list of syntax-sugaring in $\mathcal{QUIXOTE}$:

$$\begin{aligned}
o[l = o'[\ldots]] &\Leftrightarrow o[l = X] \mid \{X \cong o'[\ldots]\},\\
o[l \rightarrow o'[\ldots]] &\Leftrightarrow o[l = X] \mid \{X \sqsubseteq o'[\ldots]\},\\
o[l \leftarrow o'[\ldots]] &\Leftrightarrow o[l = X] \mid \{X \sqsupseteq o'[\ldots]\},\\
o[l = X@o'[\ldots]] &\Leftrightarrow o[l = Y] \mid \{Y \cong X, X \cong o'[\ldots]\},\\
o[l \rightarrow X@o'[\ldots]] &\Leftrightarrow o[l = Y] \mid \{Y \sqsubseteq X, X \cong o'[\ldots]\},\\
o[l \leftarrow X@o'[\ldots]] &\Leftrightarrow o[l = Y] \mid \{Y \sqsupseteq X. X \cong o'[\ldots]\}.
\end{aligned}$$

For example, the object term representing "a person who employs himself" can be written in $\mathcal{QUIXOTE}$ as $X@person[employee = X]$ is allowed, and called an **annotated variable**.

Due to the existence of self-referential object terms, we must deal with circularity in general. That is the reason why we adopt Aczel's hyperset theory as the semantic domain of $\mathcal{QUIXOTE}$, which brings all of the familiar set-theoretic techniques to deal with circular phenomena.

Our conception of objects is to see object terms as identifiers for objects (**object identifiers**). For example, $human[age = 20, sex = male]$ is a term uniquely denoting the concept of "20-aged male human", but does not mean that a *human* is 20 years old and *male*. The term (object) $human[age = 20, sex = male]$ possibly has labels other than *age* and *sex*, such as *name* and *occupation*, but such labels are not essential for defining the object according to our conception of objects. For example, the following description (called an **attribute term**) can be used to represent the fact that "20-aged male human" is married (by default) in $\mathcal{QUIXOTE}$:

$$human[age = 20, sex = male]/[married = yes].$$

The lefthand side of "/" is an object term and the righthand side is **attribution** of the object term which specifies the attributes of the object term.

As we are working on the domain $\mathcal{T}[\chi]$ of object terms containing variables and constraints over them. Thus, in $\mathcal{QUIXOTE}$, **object identity** is defined in terms of the equivalence relation on $\mathcal{T}[\chi]$. What is needed is to solve constraints or at least to check the solvability of constraints. Barwise[2] shows an important result on solvability of constraints over hypersets in terms of the existence of **Simulation Pair**, that is, a set of equivalences and subsumptions among hypersets, in the case that each variable is instantiated. Mukai[5],[4] extends the Barwise's result to the case of constraints containing disjunctions and negations, and also shows that a subclass of Aczel's hyperset theory satisfies the criteria of CLP-schema proposed by Jaffar and Lassez[3].

In Section 5, the solvability condition of our constraint language is discussed based on the Mukai's result on a constraint language over hypersets.

3 Labeled Graphs

In this section, a subclass of the domain of hypersets is defined in order to give the domain for interpretation of object terms.

First of all, we will give a subclass $\mathcal{G}$ of the domain of Aczel's hyperset. An element of $\mathcal{G}$ is called a **labeled graph**, since it is a set-theoretical encoding of a graph each of whose arcs is associated with a label.[2] The collection of (ground) labeled graphs is the largest collection $\mathcal{G}$ such that every $G \in \mathcal{G}$ be the pair (a, b) where $a \in BO^*$ and b be an empty set or a (finite) function with a subset of L as a domain and a subset of $\mathcal{G}$ as a codomain.

It is easy to see that mapping from ground object terms to ground labeled graphs is bijective, i.e., the correspondence between $\mathcal{T}$ and $\mathcal{G}$ is one-to-one. For example, the object terms *human*, $human[age = 20, sex = male]$, and $[age = 20]$ correspond to the following labeled graphs, respectively:

$$(human, \emptyset)$$
$$(human, \{(age, (20, \emptyset)), (sex, (male, \emptyset))\})$$
$$(\top, \{(age, (20, \emptyset))\}).$$

For the sake of simplicity, let $\mathcal{I}$ be the interpretation function which takes an object term and returns the corresponding ground labeled graph.

$\mathcal{G}$ is a subclass of the domain V_A of Aczel's hypersets with atoms. As in the case of object terms, it is possible to have the domain $\mathcal{G}[\chi_\mathcal{G}]$ of labeled graphs containing variables. Labeled graphs are set-theoretical constructs in the sense of Aczel's hypersets, and uniquely defined as the solution of a **system of equations**. This is what Aczel's Solution Lemma says. For example, the labeled graph $(human, \{(age, \{(20, \emptyset)\}), (sex, \{(male, \emptyset)\})\})$ is the solution of the variable x for the following system of equations:

$$x = (human, \{x_1, x_2\})$$
$$x_1 = (age, \{x_3\})$$
$$x_2 = (sex, \{x_4\})$$
$$x_3 = (20, \emptyset)$$
$$x_4 = (male, \emptyset).$$

To be more precise, an **assignment** f is to be defined as a function from a subset of $\chi_\mathcal{G}$ to a subset of $\mathcal{G}$. An assignment f with a set $X \subset \chi_\mathcal{G}$ of variables as its domain is the solution of the the system of equations $x_i = a_{x_i} (x_i \in X)$ iff for any $x_i \in X$

$$f(x_i) = f(a_{x_i})$$

holds.

The two important points of Solution Lemma are:

(1) existence of the solution,

(2) uniqueness of the solution.

Consider, for example, the interpretation (labeled graph) of the object term $X@human[employee = X]$. It is given by the following system of equations:

$$x = (human, \{x_1\})$$
$$x_1 = (employee, x).$$

According to the Aczel's lemma, the solution of x uniquely exists, call it Ω_1. Ω_1 is a hyperset which satisfies the following condition:

$$\Omega_1 = (human, \{(employee, \Omega_1)\}).$$

4 Subsumption Relation over Labeled Graphs

First, a binary relation $\sqsubseteq_\mathcal{G}$ is defined to be the largest relation satisfying the following conditions:
If $(a_1, b_1) \sqsubseteq_\mathcal{G} (a_2, b_2)$ where a_1, a_2 are basic objects and b_1, b_2 are functions from a subset of atomic labels onto a subset of $\mathcal{G}$, then

- $a_1 \preceq a_2$,
- for any pair $(l, g_2) \in b_2$, there exists a pair $(l, g_1) \in b_1$ such that $g_1 \sqsubseteq_\mathcal{G} g_2$.

For example, the following holds:

$$(human, \{(age, \{(20, \emptyset)\}), (sex. \{(male, \emptyset)\})\})$$
$$\sqsubseteq_\mathcal{G} (human, \emptyset).$$
$$(human, \{(age, \{(20, \emptyset)\}), (sex. \{(male, \emptyset)\})\})$$
$$\sqsubseteq_\mathcal{G} (\top, \{(age, \{(20, \emptyset)\})\}).$$

Furthermore, as in the case of Ω_1 above, we get the interpretation Ω_2 of the object term $X@human[employee = X, name = \text{"}John\text{"}]$ as:

$$\Omega_2 = (human, \{(employee, \Omega_2), (name, \{(\text{"}John\text{"}, \emptyset)\})\}).$$

It is not easy to see that $\Omega_2 \sqsubseteq_\mathcal{G} \Omega_1$ holds in usual set theory. But, it holds in Aczel's hyperset theory. Intuitively speaking, sets including Ω_1 and Ω_2 are coinductively defined in Aczel's hyperset theory, which means that if we compare Ω_1 and Ω_2 then it suffices to unfold only one time and compare them without Ω_1 and Ω_2. Thus, we can conclude that $\Omega_2 \sqsubseteq_\mathcal{G} \Omega_1$ holds by comparing the following two labeled graphs:

$$(human, \{(employee, \_)\}), \text{ and}$$
$$(human, \{(employee, \_), (name. \{(\text{"}John\text{"}, \emptyset)\})\}),$$

where $\_$ is supposed to be a special atom.

Notice that g_2 be unique if it exists in the second condition above, since b_2 is a function. It is easy to check that $\sqsubseteq_\mathcal{G}$ is a pre-order. Furthermore, $\sqsubseteq_\mathcal{G}$ is a partial order, i.e., both $g_1 \sqsubseteq_\mathcal{G} g_2$ and $g_2 \sqsubseteq_\mathcal{G} g_1$ hold then $g_1 = g_2$, without considering any additional equational theory on $\mathcal{G}$.[3] For the present purpose, we assume that $\sqsubseteq_\mathcal{G}$ be a partial order. Thus, $\cong_\mathcal{G}$ defined below is the equality on $\mathcal{G}$:

$$g_1 \cong_\mathcal{G} g_2 \stackrel{\text{def}}{=} g_1 \sqsubseteq_\mathcal{G} g_2 \wedge g_2 \sqsubseteq_\mathcal{G} g_1.$$

The subsumption relation $\sqsubseteq$ over object terms is defined in terms of $\sqsubseteq_\mathcal{G}$ as:

$$o_1 \sqsubseteq o_2 \stackrel{\text{def}}{=} \mathcal{I}(o_1) \sqsubseteq_\mathcal{G} \mathcal{I}(o_2).$$

[2] Aczel[1] uses the term "label" in different meaning. On his use, "label" is associated with each node of a graph, and corresponds to the value of a node in our treatment.

[3] That means we may add equality axioms in such a way that $woman \cong human[sex = female]$. But, it is not clear whether we could have this kind of equational theories over the domain of hypersets, since the basic principle of Aczel's hyperset theory is that every hyperset has a unique picture. This principle seems to be violated by adding an equational theory.

In general, a definition of an object term is associated with a constraint as shown in Section 2. $\sqsubseteq$-relation between object terms is interpreted as $\sqsubseteq_{\mathcal{G}}$-relation between solutions of a constraint, for example, an object term $human[age = Y] \mid \{Y \sqsubseteq int\}$ is defined as the solution of X in the following set of equations and subsumption:

$$X \cong_{\mathcal{G}} (human, \{X_1\})$$
$$X_1 \cong_{\mathcal{G}} (age, \{Y\})$$
$$X_2 \cong_{\mathcal{G}} (int, \emptyset)$$
$$Y \sqsubseteq_{\mathcal{G}} X_2.$$

The problem is to have conditions on solvability of constraints such as the above. Section 5 shows the condition.

Next, the meet and the join operations over $\mathcal{G}$ are introduced. For any pair of two labeled graphs (say, G_1 and G_2), it is possible to have the following description, where $0 \leq i, j, k$, o_1 and o_2 are basic objects, l_i, l'_j, and l''_k are labels, and x_i, x'_j, y_k, and y'_i are all labeled graphs:

$$G_1 = (o_1, \{(l_1, x_1), \ldots, (l_i, x_i), (l'_1, x'_1), \ldots, (l'_j, x'_j)\})$$
$$G_2 = (o_2, \{(l_1, y_1), \ldots, (l_i, y_i), (l''_1, y'_1), \ldots, (l''_k, y'_k)\}).$$

The meet and the join of two labeled graphs G_1 and G_2 above are defined recursively as follows[4]:

meet (written $G_1 \downarrow G_2$)

$$G_1 \downarrow G_2 \stackrel{\text{def}}{=} (o_1 \sqcap o_2, \{(l_1, (x_1 \downarrow y_1)), \ldots, (l_i, (x_i \downarrow y_i)), (l'_1, x'_1), \ldots, (l'_j, x'_j), (l''_1, y'_1), \ldots, (l''_k, y'_k)\})$$

join (written $G_1 \uparrow G_2$)

$$G_1 \uparrow G_2 \stackrel{\text{def}}{=} (o_1 \sqcup o_2, \{(l_1, (x_1 \uparrow y_1)), \ldots, (l_i, (x_i \uparrow y_i))\})$$

The following holds:

$$G \downarrow G \cong_{\mathcal{G}} G$$
$$G_1 \downarrow G_2 \cong_{\mathcal{G}} G_2 \downarrow G_1$$
$$G_1 \downarrow G_2 \sqsubseteq_{\mathcal{G}} G_1$$
$$G_1 \uparrow G_1 \cong_{\mathcal{G}} G_1$$
$$G_1 \uparrow G_2 \cong_{\mathcal{G}} G_2 \uparrow G_1$$
$$G_1 \sqsubseteq_{\mathcal{G}} G_1 \uparrow G_2$$

It seems to be the easy consequence of the above propositions that:

Proposition 1

$$G_1 \downarrow G_2 \Leftrightarrow inf\{G_1, G_2\},$$
$$G_1 \uparrow G_2 \Leftrightarrow sup\{G_1, G_2\},$$

since $\sqsubseteq_{\mathcal{G}}$ is a partial order on $\mathcal{G}$.

5 Solvability of Constraints over Labeled Graphs

Barwise[2] shows the solvability conditions on a set of hyperset-equations and hyperset-subsumptions over parametric hypersets (hypersets containing variables). In that paper, Barwise defines the notion of Simulation Pair which is consists of a pair of **bisimulation** relation and **simulation** relation with the same field, and show that a set of hyperset-equations and hyperset-subsumptions over a set of parametric hypersets has a solution iff a simulation pair satisfying obvious conditions exists for the same set.

This means that if we have two relations which satisfy the conditions on simulation pair, we could define a constraint by means of the two relations, and moreover we could check the solvability of the constraint over the domain of hypersets.

To get the conditions on solvability of our constraints, we must show that our constraint relation $\cong_{\mathcal{G}}$ and $\sqsubseteq_{\mathcal{G}}$ satisfies the conditions on the definitions of bisimulation relation and simulation relation.

First of all, notice that the domain $\mathcal{G}$ of labeled graphs are different from the ones in [2] in two ways:

(1) our domain is a subclass of the domain of hypersets, i.e., a domain of labeled graphs,

(2) our class of atoms (basic objects) is partially ordered.

The first point causes no problem. The second point may seems to cause difficult problems. But, we have no variables over basic objects, i.e., variables are ranging over labeled graphs, and we suppose a fixed algebraic structure on basic objects. Thus, we only need the comparison on two basic objects which can be deterministically understood by seeing the fixed domain of basic objects.

Furthermore, Barwise's result presupposes that each variable in a constraint is **instantiated**, that is, for each variable x, only one equation $x = u$[5] is in the constraint where u is a hyperset or an atom. In general, some variables might be uninstantiated. For such a case, we must make sure the way to extend a constraint without changing the solution space of the constraint.

[4] To make this definition presice, it should be noticed that there might be more than one definitional equations for one object, since we have self-referential structure such as Ω_1 above. In such case, the meet of two labeled graphs could not be defined by one (definitional) equation. Consider the following two labeled graphs:

$$X \cong_{\mathcal{G}} (human, \{(employee, \{X\})\})$$
$$Y \cong_{\mathcal{G}} (human, \{(employee, \{(john, \emptyset)\})\}).$$

To get the meet $X \downarrow Y$, we must define it as the following system of equations:

$$X \downarrow Y \cong_{\mathcal{G}} Z_1 \downarrow Z_2,$$
$$Z_1 \cong_{\mathcal{G}} (human, \{(employee, \{Z_2\})\}),$$
$$Z_2 \cong_{\mathcal{G}} X \downarrow (john, \emptyset) \cong_{\mathcal{G}} ((human \sqcap john), \{(employee, \{X\})\}).$$

In this case, $X \downarrow Z_2 \cong_{\mathcal{G}} Z_2$and $Z_1 \downarrow Z_2 \cong_{\mathcal{G}} Z_2$ hold apparently. Thus, we get

$$X \downarrow Y \cong_{\mathcal{G}} ((human \sqcap john), \{(employee, \{X \downarrow Y\})\}).$$

If $Z \cong_{\mathcal{G}} (john, \{(employee, \{Z\})\})$, then we have

$$X \downarrow Z \cong_{\mathcal{G}} ((human \sqcap john), \{(employee, \{X \downarrow Z\})\}).$$

This equation has a unique solution.

[5] $x \cong_{\mathcal{G}} u$ where u is a labeled graph for our case.

Mukai[5], [4] extends the Barwise's result and shows that constraints over a hereditary finite set is satisfaction-complete and solution-compact, which suggests that a subclass of an AFA-universe can be used as a domain for CLP(X). That is also true for our domain of labeled graphs.

In the following, we will show two results on the solvability of the constraints over labeled graphs. One is on the solvability conditions of a constraint similar to the one in [2]. The approarch is to give obvious set of constraint rules on $\cong_{\mathcal{G}}$ and $\sqsubseteq_{\mathcal{G}}$ and show that they satisfy the defining condition of bisimulation and simulation relation under the set of constraint rules. The other is on the extensionability of a given constraint so that each variable becomes instantiated.

These two results are given by Mukai's work[5] directly. Thus, we only show the results here with some additional comments. For details, see [7] (in preparation).

First of all, we will give a definition of our constraints and their solutions, similar to the one in [5] .

An **atomic constraint** is an equation $u \cong_{\mathcal{G}} v$ or a subsumption $u \sqsubseteq_{\mathcal{G}} v$, where u, v are elements of $\mathcal{G}[\chi] \cup \chi$.

A **constraint** is a set of atomic constraints. A constraint is conjunction of the atomic constraints in it.

Satisfiability relation ($\models$) is defined as usual, i.e., a binary relation between an assignment f and a constraint c (written $f \models c$) with obvious clauses.

An assignment f is a **solution** of a constraint c if $f \models c$.

To extend a given constraint, we use a set of rules on constraints (call it **constraint rules**). The constraint rules contains usual rules on equality (for $\cong_{\mathcal{G}}$) and usual rules on partial order (for $\sqsubseteq_{\mathcal{G}}$). Besides those, we have the following rules:

- If $(a_1, b_1) \cong_{\mathcal{G}} (a_2, b_2)$ then $a_1 = a_2$ and for each $(l, g_1) \in b_1$ there exists $(l, g_2) \in b_2$ such that $g_1 = g_2$,
- if $(a_1, b_1) \sqsubseteq_{\mathcal{G}} (a_2, b_2)$ then $a_1 \preceq a_2$ and for each $(l, g_2) \in b_2$ there exists (l, g_1) such that $g_1 \sqsubseteq_{\mathcal{G}} g_2$,
- if $x \sqsubseteq_{\mathcal{G}} y$ and $x \sqsubseteq_{\mathcal{G}} z$ then $x \sqsubseteq_{\mathcal{G}} (y \downarrow z)$,
- if $y \sqsubseteq_{\mathcal{G}} x$ and $z \sqsubseteq_{\mathcal{G}} x$ then $(y \uparrow z) \sqsubseteq_{\mathcal{G}} x$.

The first rule states the property (defining condition) of $\cong_{\mathcal{G}}$, the second rule states the property of $\sqsubseteq_{\mathcal{G}}$, the third and the fourth rules introduce the infimum and the supremum of two labeled graphs into a constraint. For $\downarrow$ is the infimum of two labeled graphs, $(x \downarrow y)$ means that the variable z such that $z \sqsubseteq_{\mathcal{G}} x, z \sqsubseteq_{\mathcal{G}} y$ for the case where x or y is a variable.

Notice that the correspondence between (l, g_1) and (l, g_2) is unique if it exists.

By applying the constraint rules to a given constraint, a constraint (a set of atomic constraints) closed under the constraint rules is obtained. Call it a **closure** of a given constraint.

From the definition of constraint rules shown above, $\cong_{\mathcal{G}}$ satisfies the defining condition of bisimulation relation and the converse of $\sqsubseteq_{\mathcal{G}}$ satisfies the defining condition on simulation relation, and they constitutes a simulation pair, while the treatment of atoms (basic objects) is different.

The appendix A shows the definition of bisimulation relation, simulation relation, and simulation pair. For simplicity, the domain for their definition is changed to the domain $\mathcal{G}[\chi] \cup \chi$.

As we noticed, basic objects are atoms in our domain and partially ordered, while in [2] a collection of atoms is descrete. But, we have no variable ranging over atoms (basic objects). So, the difference of the treatment of atoms causes no problem in understanding $(\sqsubseteq_{\mathcal{G}}, \cong_{\mathcal{G}})$ as simulation pair[6].

The following is the restatement of the Unification Lemma of Mukai[5] for our constraints:

Proposition 2 *(cf. Mukai[5]) the following two clauses are equivalent,*

(1) a constraint c has a normal closure,

(2) a constraint c has a solution in $\mathcal{G}$.

This proposition shows that our constraints over $\mathcal{G}$ is decidable.

In general, the closure of a given constraint might not be normal. But, it is guaranteed that any closure has a normal closure as it extension (See [5] and [7]) with a procedure to extend a closure.

Unification over object terms is given as the problem of checking the solvability of a constraint over $\mathcal{G}$. For the purpose of defining unification of two object terms, the following restriction is posed on the solution:

Definition 1 *(Condition on Unification)*
Unification succeeds only if a normal closure of a constraint contains no labeled graph g such that $g \sqsubseteq (\bot, \emptyset)$.

Following is the example of a constraint which corresponds to the unification of the two object terms X and Y such that $X \sqsubseteq human[name = N_1]$ and $Y \sqsubseteq animal[name = N_2, age = 20] \mid \{N_2 \sqsubseteq string\}$, where $human \preceq animal$:

$$
\begin{aligned}
&X \cong_{\mathcal{G}} Y \\
&X \sqsubseteq_{\mathcal{G}} (human, \{(name, \{N_1\})\}) \\
&N_1 \sqsubseteq_{\mathcal{G}} (\top, \emptyset) \\
&Y \sqsubseteq_{\mathcal{G}} (animal, \{(name, \{N_2\}), (age, \{Z\})\}) \\
&N_2 \sqsubseteq_{\mathcal{G}} (string, \emptyset) \\
&Z \cong_{\mathcal{G}} (20, \emptyset).
\end{aligned}
$$

[6] It seems to be convenient to use Aczel's notion of a labeled system to show that the treatment of atoms causes no problem. In a labeled system, each node (corresponding to a variable in a system of equations) has a unique set called label of it. Be careful that the word "label" is used in a different way we used. It is possible to suppose one-to-one mapping from a set of basic objects to a set of labels.

Starting from this constraint, the following equations and subsumptions are obtained by applying the rules on constraints and closure extension procedure.

$$X \sqsubseteq_{\mathcal{G}} (human, \{(name\{N\}), (age, \{Z\})\})$$
$$N \sqsubseteq_{\mathcal{G}} N_1$$
$$N \sqsubseteq_{\mathcal{G}} N_2$$
$$X \cong_{\mathcal{G}} (human, \{(name\{N\}), (age, \{Z\})\})$$
$$N_1 \cong_{\mathcal{G}} (\top, \emptyset)$$
$$N_2 \cong_{\mathcal{G}} (string, \emptyset)$$
$$N \cong_{\mathcal{G}} (string, \emptyset).$$

This result show that the original constraint is solvable, since all the variables in the original constraint becomes instantiated. This also show that the assignment f in the following is a possible solution of the constraint:

$$f(X) = (human, \{(name, \{(string, \emptyset)\}), (age, \{(20, \emptyset)\})\})$$
$$f(Y) = (human, \{(name, \{(string, \emptyset)\}), (age, \{(20, \emptyset)\})\})$$
$$f(Z) = (20, \emptyset)$$
$$f(N_1) = (string, \emptyset)$$
$$f(N_2) = (string, \emptyset).$$

The solvability of constraints is essential in our notion of object identity and inheritance of attributes. In the next section, we will show how constraints are related to inheritance of attributes.

6 Inheritance of Attributes

In this section, inheritance of the attributes among object terms is explained in terms of the general rule for inheritance and constraints.

First of all, we define an **attribute term**, intuitively explained in Section 2. Let $o, o_1, \ldots, o_n$ be object terms and $l_1, \ldots, l_n$ be labels. An attribute term is defined as follows:

$$o/[l_1\ op_1\ o_1, \ldots, l_n\ op_n\ o_n],$$

where op_i $(1 \leq i \leq n)$ is $\rightarrow$, $\leftarrow$, or $=$. Each $l_1\ op_i\ o_i$ is called an attribute (specification) of o. Each attribute can be written in the form of a constraint as object terms in Section 2 and furthermore constraints of an object term can be gathered into a set of constraint of an attribute term by renaming variables appropriately.

For example, consider the following:

$$o[l = X] | \{X \sqsubseteq o'\} / [l_1 = X, l_2 = Y] | \{X \cong o_1, Y \sqsupseteq o_2\}.$$

By renaming X in the attribution to Z, we can get

$$o[l = X] / [l_1 = Z, l_2 = Y] | \{X \sqsubseteq o', Z \cong o_1, Y \sqsupseteq o_2\}.$$

Next, it should be noticed that the treatment of labels are different in object terms and attribution.

Consider, for example, the following description which specifies the attribution of an object term:

$$o[l = x] / [l_1 \rightarrow x_1, l_2 \leftarrow x_2, l_3 = x_3].$$

As shown above, $o[l = x]$ is an object term and interpreted as a labeled graph, that is, labels are used as the name of the arcs in a labeled graph. To the contrary, in attribution, labels are interpreted as unary functions over $\mathcal{G}[\chi]$. The attribution in the above description are interpreted as the following constraint:

$$l_1((o, \{(l, x)\})) \sqsubseteq_{\mathcal{G}} x_1$$
$$x_2 \sqsubseteq_{\mathcal{G}} l_2((o, \{(l, x)\}))$$
$$l_3((o, \{(l, x)\})) \cong_{\mathcal{G}} x_3.$$

To refer to the value of an attribute, say l_1 in the above, a term $o[l = x].l_1$ is used, and called a **dotted term**. The interpretation of the dotted term $o[l = x].l_1$ is $l_1((o, \{(l, x)\}))$.

For the labels which are not explicitly specified, we assume that their values exists but not be constrained. Let l_4 be a label. The value of the attribute l_4 of the above example is constrained as:

$$l_4((o, \{(l, x)\})) \sqsubseteq_{\mathcal{G}} (\top, \emptyset).$$

Thus, in $\mathcal{QUIXOTE}$, every label is interpreted as a function defined for each object term.

It is natural to assume that attribution is inherited among object terms with respect to $\sqsubseteq$-ordering. For example, consider the following example:

$$swallow \sqsubseteq bird.$$
$$bird/[canfly \rightarrow yes].$$

Since, *swallow* is *bird* and *bird* has the attribution $[canfly \rightarrow yes]$, *swallow* should have the same attribution by default.

General rule for inheritance of attributes among objects is:

Definition 2 *(Rule for inheritance)*

$$o_1 \sqsubseteq o_2 \ \Rightarrow\ o_1.l \sqsubseteq o_2.l.$$

By this rule, the following holds:
if $o_1 \sqsubseteq o_2$, then

- if o_2 has the attribution $[l \rightarrow o']$, then o_1 also has the same attribution,
- if o_1 has the attribution $[l \leftarrow o']$, then o_2 also has the same attribution.

Notice that the both hold for the attribution $[l = o']$ by definition.

Furthermore, it is possible to introduce the notion of exceptions on inheritance of attribution by assuming the additional rule for inheritance.

The rule for exception is stated as follows:

Definition 3 *(Rule for exception)*
The specification of labels in an object term is overridden against the attribution of the object term.

For example, consider the attribution of the object term $bird[canfly \rightarrow no]$ with respect to the following definition:

$$bird/[canfly \rightarrow yes].$$

By the rule for inheritance, $bird[canfly \rightarrow no]$ inherits the attribution $[canfly \rightarrow yes]$. But, $bird[canfly \rightarrow no]$ has the specification on the label $canfly$ in it. Thus, $bird[canfly \rightarrow no]$ has the attribution $[canfly \rightarrow no]$ by the rule for exception.

As a consequence of the rule for exception, the following holds:

$$o[l_1\ op_1\ x_1, \ldots, l_n\ op_n\ x_n]/[l_1\ op_1\ x_1, \ldots, l_n\ op_n\ x_n],$$

where $op_i \in \{\rightarrow, \leftarrow, =\}(1 \leq i \leq n)$.

Taking into accounts of inheritance and exceptions, the attribution of an object term o having the specifications of labels $l_1 = x_1, \ldots, l_n = x_n$ is defined as follows:

$$((C_o \cup C_\downarrow^o \cup C_\uparrow^o) \setminus \{l_1, \ldots, l_n\}) \cup \{o.l_1 \cong x_1, \ldots, o.l_n \cong x_n\},$$

where C_o be the set of constraints corresponding to the attribution of the object term o which o itself has, $C_\downarrow^o$ be the set of constraints corresponding to the attribution of the object term o inherited from the object terms which are larger than o, $C_\uparrow^o$ be the set of constraints corresponding to the attribution of the object term o inherited from the object terms which are smaller than o. Also, $C \setminus \{l_1, \ldots, l_n\}$ is the constraint which is the result of removing the atomic constraints corresponding to the $l_1, \ldots, l_n$ attributes from C.

Sometimes, the attribution of an object term might have no solution. In such a case, the definition of the object term is said to be **inconsistent**. For example, consider the following:

$$bird/[canfly = yes]$$
$$penguin/[canfly = no]$$
$$penguin \sqsubseteq bird.$$

Here, as $penguin$ inherits an attribute $[canfly \rightarrow yes]$ from $bird$, $penguin$ has a constraint $[canfly \sqsubseteq yes \downarrow no] = [canfly \rightarrow \bot]$, which is inconsistent.

7 Modules

In $\mathcal{QUIXOTE}$, special kind of object terms called **module identifiers** are introduced to modularize a set of object terms.

The description of the form

$$m :: \sigma \| C$$

is called a **(unit) rule**, where m is a module identifier and σ is an attribute term, and C is a constraint. This unit rule says that σ is in m. Corresponding to the unit rule $s :: \sigma$, the **proposition** $m : \sigma$ is defined as:

$$(m : \sigma) \text{ is true iff } m :: \sigma.$$

In $\mathcal{QUIXOTE}$, a (non-unit) rule is also available:

$$m :: \sigma <= m_1 : \tau_1, m_2 : \tau_2, \ldots, m_n : \tau_n \| C.$$

where $m, m_1, \ldots, m_n$ are module identifiers, and $\sigma, \tau_1, \ldots, \tau_n$ are attribute terms, and C is a constraint.

This rule says that if $m_i : \tau_i$ $(1 \leq i \leq n)$ are all true, then σ is in m ($m : \sigma$ is true). This rule also says that this rule is in m (this rule is accessible from m only).

General rule for inheritance of rules among modules is:

$$m \sqsubseteq m', m :: t \Rightarrow m' :: t$$

For the purpose of the present paper, we restrict attention to unit rules, and the relationship between modules and inheritance of attribution.

Thus, for example, if

$$m_1 :: john \cong human[name = \text{"}John\text{"}].$$
$$m_2 :: john/[age \rightarrow 20].$$
$$m_3 :: john/[age \rightarrow 30].$$
$$m_1 \sqsubseteq m_2.$$
$$m_1 \sqsubseteq m_3.$$

then $john$ is 20 years old or 30 years old depending on whether it is in m_2 or m_3, while its name is "$John$" in both m_2 and m_3. Note that the existence of an object term is global over the concept of modules, while the attribution of object terms is local and is inherited through subsumption relation between module identifiers. If we specify $m_2 \sqsubseteq m_4$ and $m_3 \sqsubseteq m_4$, m_4 has the constraint $[age \rightarrow 20 \downarrow 30] = [age \rightarrow \bot]$, that is, $john$ becomes to have inconsistent definition.

As a module identifier is defined as an object trem, an module can be parameterized, i.e., abstraction. For example, consider the following:

$$m[l = X] :: john[age \rightarrow X].$$

The variable X could be instantiated into some integer during processing.

Acknowledgments

The authors would like to express special thanks to Kuniaki Mukai who gives us a guideline to Aczel's hyperset thoery and a basic ideas on constraints over hypersets.

The authors also would like to thank Atsushi Ohori for his valuable comments on an earlier draft of this paper.

Finally, the authors would like to thank members of $\mathcal{QUIXOTE}$ meeting for their stimulus discussions and collaborations on designing $\mathcal{QUIXOTE}$.

References

[1] P. Aczel: Non-Well-Founded Set Theory, CSLI Lecture Notes No. 14, 1988.

[2] J. Barwise: AFA and the Unification of Information, in *The Situation in Logic*, CSLI Lecture Notes No. 17, 1989.

[3] J. Jaffar and J.-L. Lassez: Constraint Logic Programming, In *Proceedings of the 14th ACM Symposium on Principle of Programming Languages*, 1987.

[4] K. Mukai: Coinductive Semantics of Horn Clauses with Compact Constraint, ICOT Technical Report TR-562, 1990.

[5] K. Mukai: CLP(AFA): Coinductive Semantics of Horn Clauses with Compact Constraint, *The Second Conference on Situation Theory and Its Applications*, Kinloch Rannoch Scotland, Sep., 1990.

[6] K. Yokota and S. Nishio: Towards Integration of Deductive Databases and Object-Oriented Databases: A Limited Survey, *Advanced Database System Symposium*, Kyoto, Dec.7-8, 1989.

[7] H. Yasukawa and K. Mukai: Constraints over Complex Objects, in preparation.

[8] H. Yasukawa and K. Yokota: An Overview of a Knowledge Representation Language $\mathcal{QUIXOTE}$, draft, 1990.

A Bisimulation and Simulation

Here, we will give a definition of bisimulation relation and simulation pair of [2]. For simplicity, we use the domain of labeled graph instead of the one of hypersets.

Definition 4 *A* bisimulation relation $(\sim)$ *is an equivalence relation* $\sim$ *on some subclass of* $\mathcal{G}[\chi] \cup \mathcal{BO}^* \cup \chi$ *satisfying the following condition.*

If $(a_1, b_1) \sim (a_2, b_2)$, *then*

(1) $a_1 = a_2$,

(2) for every $(l, g_1) \in b_1$, *there is a* $(l, g_2) \in b_2$ *such that* $g_1 \sim g_2$,

and if $x \sim u$ *where* x *is a variable, then*

(3) there is some labeled graph u *such that* $x \sim u$.

For the condition (2), symmetric condition follows, since $\sim$ is an equivalence relation.

It is easy to see that $\cong_{\mathcal{G}}$ satisfies the conditions (1) and (2) for ground labeled graphs.

Next, we will give a definition of simulation pair, also in the way suitable for some subclass $\mathcal{G}[\chi] \cup \mathcal{BO}^* \cup \chi$.

Definition 5 *A* simulation pair $(\prec, \sim)$ *is a pair of relations* $\prec, \sim$ *with the same field which satisfies the followings:*

(1) $\sim$ *is a bisimulation relation.*

(2) $\prec$ *is a simulation relation:*

If $(a_1, b_1) \prec (a_2, b_2)$ *where* $(a_1, b_1), (a_2, b_2)$ *are two labeled graphs, then*

- $a_1 = a_2$,
- *for all* $(l, g_1) \in b_1$, *there is a* $(l, g_2) \in b_2$ *such that* $g_1 \prec g_2$.

(3) $\sim$ *is a congruence relation with respect to* $\prec$. *That is,*

- $u \sim v$ *implies* $u \prec v$,
- $u \prec v$, $u \sim u'$, *and* $v \sim v'$ *imply* $u' \prec v'$.