Ontological Knowledge Base Reasoning with Sort-Hierarchy and Rigidity[*]

Ken Kaneiwa
National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan
kaneiwa@nii.ac.jp

Riichiro Mizoguchi
Osaka University
8-1 Mihogaoka, Ibaraki, Osaka 567-0047, Japan
miz@ei.sanken.osaka-u.ac.jp

Abstract

Although sorts and unary predicates are semantically identical in order-sorted logic, they are classified as different kinds of properties in formal ontology (e.g. sortal and non-sortal). This ontological analysis is an essential notion to deal with properties (or sorts) of objects in knowledge representation and reasoning. In this paper, we propose an extension of an order-sorted logic with the ontological property classification. This logic contains types (rigid sorts), non-rigid sorts and unary predicates to distinguishably express the properties: substantial sorts, non-substantial sorts and non-sortal properties. We define a sorted Horn-clause calculus for such property expressions in a knowledge base. Based on the calculus, we develop a reasoning algorithm for many separated knowledge bases where each knowledge base can extract rigid property information from other knowledge bases (called rigid property derivation).

Introduction

Order-sorted logic (Oberschelp 1962; Cohn 1987; Walther 1988; Weibel 1997) involves many sorts and their hierarchy (called sort-hierarchy). The advantages of order-sorted logic are the following three points. The first is a reduction of the search space, by restricting the domains and ranges of functions, predicates and variables to subsets of the universe (Walther 1985; 1987; 1988). The second is structural knowledge representation by means of partially ordered sorts, whereas the description of first-order logic is flat (Cohn 1989). In the field of artificial intelligence, the use of sorts has facilitated two kinds of knowledge representation: assertions and class-hierarchies, in logic programming languages and deduction systems (Aït-Kaci & Nasr 1986; Kifer, Lausen, & Wu 1995; Smolka 1989; Kaneiwa & Tojo 1999). The third is the detection of sort errors in well-sorted formulas (Oberschelp 1989), like typed programming languages. For example, let $breathing(x\colon animal)$ be well-sorted. The formula $breathing(1\colon int)$ contains a sort error, as opposed to being a false formula. However, in unsorted logic, we cannot detect sort errors even if the sorted formula $breathing(x\colon animal)$ is expressible by $animal(x) \Rightarrow breathing(x)$, since, $animal(1) \Rightarrow breathing(1)$ is regarded as a well-formed formula.

In sort theory (Beierle *et al.* 1992; Frisch 1989) and constraint logic (Bürckert 1994), the subsort relation $s < s'$ in a sort-hierarchy is represented by its equivalent implication form $s(x) \Rightarrow s'(x)$ in first-order logic. This translation motivates us to consider whether or not sorts and unary predicates are logically and semantically identical. In the literature of ontology, these sorts and unary predicates correspond to properties as sets of individuals. Furthermore, in (Guarino, Carrara, & Giaretta 1994), properties are subdivided into sortal and non-sortal (Strawson 1959; Lowe 1989) where sortal is classified as substantial (e.g. apple) and non-substantial (e.g. student), and non-sortal is as pseudo-sortal (e.g. gold) and characterizing (e.g. red). This property classification is important in order to take account of the differences between sorts and unary predicates in structural knowledge systems.

However, the formalization of order-sorted logic does not include an ontological property classification, such as sortal and non-sortal. On the other hand, ontology researchers do not seem to be concerned with incorporating such notions into an alternative logical system, by defining the syntax, semantics and inference system (cf. (Carrara & Giaretta 2001; Kaplan 2001; Gangemi, Guarino, & Oltramari 2001)).

In this paper, we refine an order-sorted logic by combining it with the ontological property classification. Specifically, we take the notions of substantial sorts, non-substantial sorts and non-sortal properties expressed as types τ (Guarino & Welty 2000a), non-rigid sorts σ and unary predicates p (as shown in Table 1). The distinction between properties varies the meaning of *instantiation* of properties (i.e. a term t is an instance of a property). Semantically, the instantiation of a type $\tau(t)$ means that t eternally belongs to τ since all types are rigid. In contrast, the instantiation of a non-rigid sort $\sigma(t)$ and a unary predicate $p(t)$ is not always true since non-rigid sorts and unary predicates are not rigid. For example, let $person$ be a type, $student$ be a non-rigid sort, and $happy$ be a unary predicate. Then, $person(john)$ can be true anytime, but the truth of $student(john)$ and $happy(john)$ is changeable in different situations. Using

[*]This research has been partially supported by the Kayamori Foundation of Informational Science Advancement and the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (14780311).

Table 1: Rigidity of types, sorts and unary predicates

property	expression		instance_of		subsumption	
substantial sort	sort s	type τ	$\tau(t)$	rigid	$\tau_1 < \tau_2$	
non-substantial sort		non-rigid sort σ	$\sigma(t)$	non-rigid	$\sigma_1 < \tau_2$	for all situations
					$\sigma_1 < \sigma_2$	
non-sortal property	unary predicate p		$p(t)$		$p_1(x) \Rightarrow p_2(x)$	for some situations

our logic, *subsumption* between properties (i.e. a property subsumes a property) can be expressed by either of the two forms : *subsort* and *implication*. The subsort relations $\tau_1 < \tau_2$, $\sigma_1 < \tau_2$ and $\sigma_1 < \sigma_2$ are declared to be true in all situations, but the implication form $p_1(x) \Rightarrow p_2(x)$ holds only in some situations. For example, $student < person$ is true in any situation, but the truth of $rich(x) \Rightarrow happy(x)$ depends on each situation. This specification indicates that the implication form $s_1(x) \Rightarrow s_2(x)$ is inadequate to render subsumption between sorts s_1, s_2 rigid[1]. Hence, we should choose the subsort relation $s_1 < s_2$ to express subsumption between sorts. Additionally, any relation of the form $\tau_1 < \sigma_2$ (i.e. a type is a subsort of a non-rigid sort) is not allowed as subsumption due to the consideration of rigid and non-rigid properties in (Guarino & Welty 2000b).

To suitably formalize these notions in logic, we employ an order-sorted logic with sort predicates (Beierle *et al.* 1992; Kaneiwa 2004a) as a basic language. For types τ and non-rigid sorts σ, this logic allows us to express their corresponding unary predicates p_τ and p_σ (called sort predicates)[2] when denoting atomic formulas $p(t)$ of unary predicates p. Our approach carefully addresses the rigidity of types, non-rigid sorts and unary predicates in sorted expressions of the logic. In standard order-sorted logic, a variable x of sort s is denoted by $x\colon s$ and a constant c of sort s is by $c\colon s$. In contrast, we allow the expressions $x\colon \tau$, $x\colon \sigma$ and $c\colon \tau$, but do not allow the expression $c\colon \sigma$. For example, $x_1 : person$ and $x_2 :\ student$ are variables of the type (rigid sort) $person$ and the non-rigid sort $student$, and $john\colon person$ is a constant of the type $person$. However, a constant of a non-rigid sort $c\colon \sigma$ may be meaningless, e.g., $john\colon student$ if John is not a student eternally. Namely, since the elements of non-rigid sorts are not fixed, an element may not belong to a non-rigid sort in a situation. Thus, we require that such a non-rigid sort be used as a sort of variables $x\colon \sigma$ and as a sort predicate $\sigma(t)$ (or $p_\sigma(t)$). For example, we can describe the formulas

$$excellent(x\colon student) \Rightarrow getting\_a\_scholarship(x\colon student)$$

and $student(john)$.

This paper presents a reasoning algorithm for dealing with the rigidity of types, non-rigid sorts and unary predicates and their hierarchies. Following the notion of rigidity, instantiation and subsumption of properties behave meaningfully toward reasoning in many knowledge bases. Figure 1 shows many knowledge bases separately constructed in their respective situations, together with the hierarchies of sorts and types. In this case, each knowledge base can commonly make use of sorts and types in the hierarchies, and furthermore it might be able to extract rigid property information from other knowledge bases. As a motivating example, we consider the following knowledge bases (with the sort and type hierarchies in Figure 2):

Knowledge base 1:

(1a) $male\_student(john\colon person)$,

(1b) $excellent(john\colon person)$,

(1c) $excellent(x\colon student) \Rightarrow getting\_a\_scholarship(x\colon student)$

Knowledge base 2:

(2a) $teacher(mary\colon person)$,

(2b) $likes(mary\colon person, x\colon student)$,

(2c) $likes(mary\colon person, x\colon bird)$

Knowledge base 3:

(3a) $canary(peter\colon animal)$,

(3b) $bird(x\colon animal) \Rightarrow canfly(x\colon animal)$

Knowledge base 4:

(4a) $father(tony\colon animal, peter\colon animal)$,

(4b) $father(y\colon animal, x\colon animal) \wedge bird(x\colon animal) \Rightarrow bird(y\colon animal)$

Instantiation (3a) is true in knowledge base 3, and if the subsumption $canary < bird$ holds in the sort and type hierarchies, then $bird(peter\colon animal)$ can be derived in knowledge base 3. This is also true in any other knowledge bases since the valuation of the type $bird$ does not depend on a particular situation, i.e., it is rigid. In other words, the elements of the type $bird$ can be fixed (Peter is eternally a bird), unlike the elements of the non-rigid sort $student$. Hence, since instantiation as rigid property information must be true in all situations, $bird(peter\colon animal)$ is extensively true in knowledge bases 1, 2 and 4. By this additional information, knowledge base 2 concludes the new fact $likes(mary\colon person, peter\colon animal)$ from (2c),

[1]Note that we here discuss rigidity of subsumption but not rigidity of properties.

[2]A sort predicate p_s is simply denoted by s when this will not cause confusion.

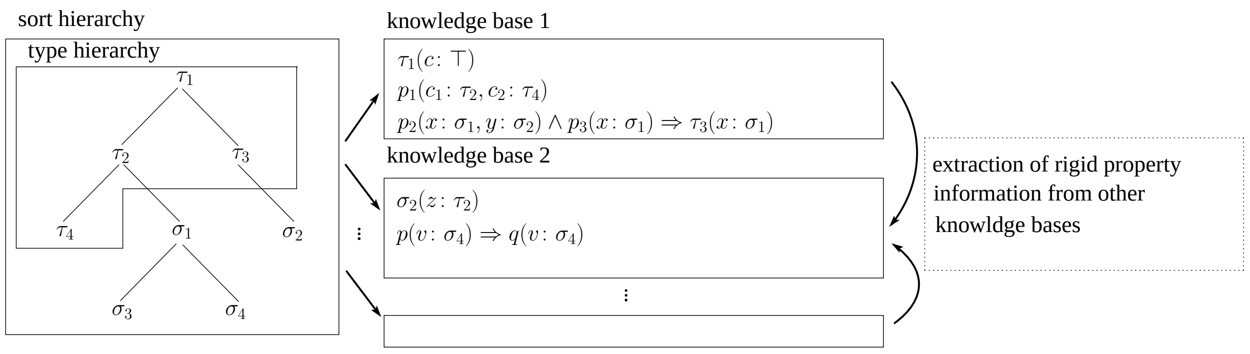
sort hierarchy
type hierarchy
τ_1
τ_2
τ_3
τ_4
σ_1
σ_2
σ_3
σ_4
knowledge base 1
$\tau_1(c\colon \top)$
$p_1(c_1\colon \tau_2, c_2\colon \tau_4)$
$p_2(x\colon \sigma_1, y\colon \sigma_2) \wedge p_3(x\colon \sigma_1) \Rightarrow \tau_3(x\colon \sigma_1)$
knowledge base 2
$\sigma_2(z\colon \tau_2)$
$p(v\colon \sigma_4) \Rightarrow q(v\colon \sigma_4)$
extraction of rigid property information from other knowldge bases

Figure 1: Many knowledge bases with sort and type hierarchies

which cannot be derived without importing the rigid property information. Moreover, knowledge base 4 can derive $bird(tony\colon animal)$ (Tony is a bird) from fact (4a) and rule (4b). This instantiation $bird(tony\colon animal)$ (as rigid property information) recursively[3] and furthermore results in $canfly(tony\colon animal)$ in knowledge base 3 and $likes(mary\colon person, tony\colon animal)$ in knowledge base 2. However, when instantiation (1a) and the subsumption $male\_student < student$ hold, $student(john\colon person)$ is true only in knowledge base 1 because $student$ is not rigid. Hence, (2b) does not derive $likes(mary\colon person, john\colon student)$ in knowledge base 3.

This paper is organized as follows. First, we formalize an order-sorted logic extended with types, non-rigid sorts and sort predicates. Next, we develop a Horn-clause calculus with sorted and unsorted substitutions and inference rules of sort predicates. Based on the calculus, we present a derivation method of rigid properties for many separated knowledge bases. Then, we prove the completeness of the sorted Horn-clause calculus and the rigid property derivation. Finally, we discuss the conclusions and future work.

An Order-sorted Logic with Rigidity and Sort Predicates

We introduce the classified property expressions: types, non-rigid sorts and unary predicates in the syntax and semantics of an order-sorted logic with sort predicates (based on (Socher-Ambrosius & Johann 1996; Schmidt-Schauss 1989; Manzano 1993)).

Syntax

Definition 1 *The alphabet of a sorted first-order language $\mathcal{L}$ with rigidity and sort predicates contains the following symbols:*

1. *T: a countable set of type symbols $\tau_1, \tau_2, \ldots$ including the greatest type $\top$*
2. *N: a countable set of non-rigid sort symbols $\sigma_1, \sigma_2, \ldots$ with $T \cap N = \emptyset$*
3. *C: a countable set of constant symbols*
4. *F_n: a countable set of n-ary function symbols*
5. *P_n: a countable set of n-ary predicate symbols*
6. *$\leftarrow, (,)$: the connective and auxiliary symbols*

[3]First, knowledge base 3 exports rigid property information to knowledge base 4. Then, a new fact can be derived in knowledge base 4, and inversely, knowledge base 3 imports the new fact as rigid property information from knowledge base 4.

We generally call type symbols (denoted $\tau, \tau_1, \tau_2, \ldots$) or non-rigid sort symbols (denoted $\sigma, \sigma_1, \sigma_2, \ldots$) *sort symbols* (denoted $s, s_1, s_2, \ldots$). Namely, $T \cup N$ is the set of sort symbols. V_s denotes an infinite set of variables $x\colon s, y\colon s, z\colon s, \ldots$ of sort s. The set of variables of all sorts is denoted by $V = \bigcup_{s \in T \cup N} V_s$. For all sorts $s \in T \cup N$, the unary predicates $p_s \in P_1$ indexed by the sorts s (called *sort predicates*) are introduced, and the set of sort predicates is denoted by $P_{T \cup N} = \{p_s \mid s \in T \cup N\}$. In particular, the predicate p_τ indexed by a type τ is called a *type predicate*, and the predicate p_σ indexed by a non-rigid sort σ is called a *non-rigid sort predicate*. In what follows, we assume that the language $\mathcal{L}$ contains all the sort predicates in $P_{T \cup N}$.

Definition 2 (Sorted Signatures with Rigidity) *A signature of a sorted first-order language $\mathcal{L}$ with rigidity and sort predicates (called sorted signature) is a tuple $\Sigma = (T, N, \Omega, \leq, \leq_+)$ such that:*

1. *$(T, \leq)$ is a partially ordered set of types (called a type hierarchy) where T is the same as the set of type symbols in $\mathcal{L}$;*
2. *$(T \cup N, \leq_+)$ is a partially ordered set of sorts (called a sort hierarchy) where*
   - *(a) $T \cup N$ is the union of the set of type symbols and the set of non-rigid symbols in $\mathcal{L}$,*
   - *(b) each ordered pair is not of the form $\tau \leq_+ \sigma$ (i.e. it is of the form $\sigma_i \leq_+ \sigma_j$, $\sigma \leq_+ \tau$ or $\tau_k \leq_+ \tau_l$), and*
   - *(c) if $\tau_i \leq \tau_j$, then $\tau_i \leq_+ \tau_j$;*
3. *if $c \in C$, then there is a unique constant declaration of the form $c\colon \rightarrow \tau \in \Omega$ (which means $c\colon \emptyset \rightarrow \tau \in \Omega$);*
4. *if $f \in F_n$, then there is a unique function declaration of the form $f\colon \tau_1 \times \cdots \times \tau_n \rightarrow \tau \in \Omega$;*
5. *if $p \in P_n$, then there is a unique predicate declaration of the form $p\colon s_1 \times \cdots \times s_n \in \Omega$ (in particular, for all sort predicates $p_s \in P_{T \cup N}$, $p_s\colon \top \in \Omega$).*

Constants and functions are required to be rigidly sorted. Specifically, constant and function declarations are defined by types τ_i in order to avoid the non-rigid domains and ranges of constants and functions. In contrast, predicate declarations are defined by any sorts s_i (types or non-rigid sorts), since the domains of predicates can be not rigid. For instance, consider the constant and function declarations:

$$john\colon\ \rightarrow person \in \Omega$$
$$father\colon person \rightarrow person \in \Omega$$

for $john \in C$, $father \in F_1$ and $person \in T$, and the predicate declaration:

$$getting\_a\_scholarship\colon student \in \Omega$$

for $getting\_a\_scholarship \in P_1$ and $student \in N$. If unrigidly sorted constants and functions are allowed for, then the following declarations:

$$john\colon\ \rightarrow student \in \Omega$$
$$father\colon person \rightarrow teacher \in \Omega$$

for $student, teacher \in N$ give rise to the sorted terms:

$$john\colon student$$
$$father(john\colon student)\colon teacher.$$

The first expresses the constant $john$ of the non-rigid sort $student$, and the second represents the function $father$ with the argument $john\colon student$ the range of which is the non-rigid sort $teacher$. From the perspective of rigidity, we regard these expressions as meaninglessly sorted because John and John's father are not a student and a teacher eternally.

The type hierarchy in a sorted signature corresponds to the backbone taxonomy consisting only of rigid properties (Welty & Guarino 2001). By the consideration of subsumption in (Guarino & Welty 2000b), any relation of the form $\tau \leq_+ \sigma$ is not allowed in the sort hierarchy, since types τ are rigid but non-rigid sorts σ are not rigid. Next, we define terms, atoms (atomic formulas), goals and clauses of a sorted first-order language with rigidity and sort predicates.

Definition 3 (Typed Terms) *Let $\Sigma = (T, N, \Omega, \leq, \leq_+)$ be a sorted signature. The set $\mathcal{T}^-_\tau$ of terms of type τ is defined by the following:*

1. *If $x\colon \tau \in V_\tau$, then $x\colon \tau \in \mathcal{T}^-_\tau$.*
2. *If $c \in C$ and $c\colon\ \rightarrow \tau \in \Omega$, then $c\colon \tau \in \mathcal{T}^-_\tau$.*
3. *If $t_1 \in \mathcal{T}^-_{\tau_1}, \ldots, t_n \in \mathcal{T}^-_{\tau_n}$, $f \in F_n$ and $f\colon \tau_1 \times \cdots \times \tau_n \rightarrow \tau \in \Omega$, then $f(t_1, \ldots, t_n)\colon \tau \in \mathcal{T}^-_\tau$.*
4. *If $t \in \mathcal{T}^-_{\tau'}$ and $\tau' \leq \tau$, then $t \in \mathcal{T}^-_\tau$.*

Note that the set $\mathcal{T}^-_\tau$ of terms of type τ contains terms of subtypes τ' with $\tau' \leq \tau$ (by the forth clause in Definition 3). For example, let $person, animal$ be types and let $person \leq animal \in \Omega$. Then, $john\colon person$ belongs to the set $\mathcal{T}^-_{animal}$ of terms of the type $animal$.

Definition 4 (Non-rigid Sorted Terms) *Let $\Sigma = (T, N, \Omega, \leq, \leq_+)$ be a sorted signature. The set $\mathcal{T}^-_\sigma$ of terms of non-rigid sort σ is defined by the following:*

1. *If $x\colon \sigma \in V_\sigma$, then $x\colon \sigma \in \mathcal{T}^-_\sigma$.*
2. *If $t \in \mathcal{T}^-_{\sigma'}$ and $\sigma' \leq_+ \sigma$, then $t \in \mathcal{T}^-_\sigma$.*

By this definition, the only non-rigid sorted terms are variables. We call a typed term or a non-rigid sorted term a *sorted term*.

Definition 5 (Sorted Terms) *Let $\Sigma = (T, N, \Omega, \leq, \leq_+)$ be a sorted signature. The set $\mathcal{T}_s$ of terms of sort s is defined by the following:*

1. *If $t \in \mathcal{T}^-_s$, then $t \in \mathcal{T}_s$.*
2. *If $t \in \mathcal{T}_{s'}$ and $s' \leq_+ s$, then $t \in \mathcal{T}_s$.*

The set of terms of all sorts is denoted by $\mathcal{T} = \bigcup_{s \in T \cup N} \mathcal{T}_s$. The function $sort$ is a mapping from sorted terms to their sorts, defined by (i) $sort(x\colon s) = s$, (ii) $sort(c\colon \tau) = \tau$ and (iii) $sort(f(t_1, \ldots, t_n)\colon \tau) = \tau$.

Definition 6 *The function $Var\colon \mathcal{T} \rightarrow 2^V$ is defined by the following:*

1. *$Var(x\colon s) = \{x\colon s\}$,*
2. *$Var(c\colon \tau) = \emptyset$ for $c \in C$, and*
3. *$Var(f(t_1, \ldots, t_n)\colon \tau) = \bigcup_{1 \leq i \leq n} Var(t_i)$ for $f \in F_n$.*

A sorted term is called *ground* if it is without variables. $\mathcal{T}_0 = \{t \in \mathcal{T} | Var(t) = \emptyset\}$ is the set of ground sorted terms. The set of ground terms of sort s is denoted by $\mathcal{T}_{0,s} = \mathcal{T}_0 \cap \mathcal{T}_s$. In the following, Horn clauses (Lloyd 1987; Doets 1994) are defined by sorted formulas of the language.

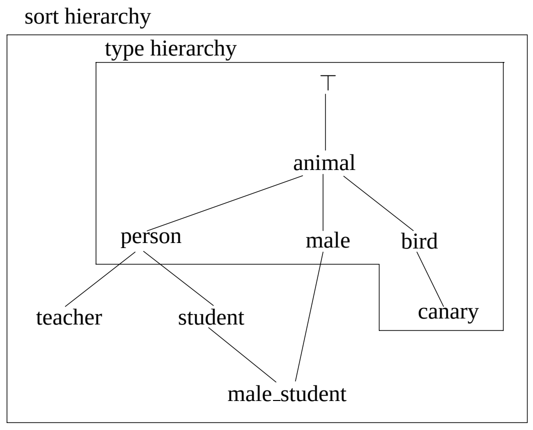

Figure 2: Sort and type hierarchies in a sorted signature

Definition 7 *Let $\Sigma = (T, N, \Omega, \leq, \leq_+)$ be a sorted signature. The set $\mathcal{A}$ of atoms, the set $\mathcal{G}$ of goals and the set $\mathcal{C}$ of clauses are defined by the following:*

1. *If $t_1 \in \mathcal{T}_{s_1}, \ldots, t_n \in \mathcal{T}_{s_n}$, $p \in P_n$ and $p\colon s_1 \times \cdots \times s_n \in \Omega$, then $p(t_1, \ldots, t_n) \in \mathcal{A}$.*
2. *If $L_1, \ldots, L_n \in \mathcal{A}$ $(n \geq 0)$, then $\{L_1, \ldots, L_n\} \in \mathcal{G}$.*
3. *If $G \in \mathcal{G}$ and $L \in \mathcal{A}$, then $L \leftarrow G \in \mathcal{C}$.*

An atom $p_s(t)$ with a sort predicate is simply denoted by $s(t)$ when this will not cause confusion. The clauses $L \leftarrow G$ are denoted by $L \leftarrow$ if $G = \emptyset$.

Definition 8 *The function $EVar\colon \mathcal{A} \cup \mathcal{G} \cup \mathcal{C} \rightarrow 2^V$ is defined by the following:*

1. $EVar(p(t_1, \dots, t_n)) = Var(t_1) \cup \cdots \cup Var(t_n)$
2. $EVar(\{L_1, \dots, L_n\}) = EVar(L_1) \cup \cdots \cup EVar(L_n)$
3. $EVar(L \leftarrow G) = EVar(L) \cup EVar(G)$

Example 1 *Let us consider the sorted signature $\Sigma = (T, N, \Omega, \leq^*, \leq^*_+)$ such that*

$$\begin{array}{rcl} T & = & \{\, male,\ person,\ canary,\ bird,\ animal,\ \top \,\}, \\ N & = & \{\, teacher,\ student,\ male\_student \,\}, \\ \Omega & = & \{\, john\colon \to person,\ mary\colon \to person, \\ & & peter\colon \to animal,\ tony\colon \to animal, \\ & & excellent\colon person,\ likes\colon animal \times \top, \\ & & getting\_a\_scholarship\colon student, \\ & & father\colon \top \times \top,\ canfly\colon \top \,\} \cup \\ & & \{\, p_s\colon \top \mid s \in T \cup N \,\}, \\ \leq & = & \{\, (canary, bird), (bird, animal), (animal, \top), \\ & & (person, animal), (male, animal) \,\}, \\ \leq_+ & = & \leq \cup \{\, (male\_student, male), \\ & & (teacher, person), (student, person), \\ & & (male\_student, student) \,\}. \end{array}$$

and $\leq^*, \leq^*_+$ are the reflexive and transitive closures of $\leq$, $\leq_+$. This sorted signature declares the sort and type hierarchies in Figure 2. The expressions:

$$student(john\colon person) \leftarrow,$$
$$canfly(x\colon animal) \leftarrow \{bird(x\colon animal)\}$$

are clauses in the sorted first-order language.

Semantics

The semantics of an order-sorted logic with rigidity and sort predicates is defined by a Kripke model. This characterizes the rigidity of types and non-rigid sorts by interpreting them in the set of worlds.

Definition 9 (Σ-model) *Let Σ be a sorted signature. A sorted Σ-model M is a tuple (W, U, I) such that*

1. W is a non-empty set of worlds,
2. U is a non-empty set of individuals,
3. $I = \{I_w \mid w \in W\}$ is the set of interpretation functions I_w for all worlds $w \in W$ with the following conditions:
   (a) if $s \in T \cup N$, then $I_w(s) \subseteq U$ (in particular, $I_w(\top) = U$),
   (b) if $s_i \leq_+ s_j$ for $s_i, s_j \in T \cup N$, then $I_w(s_i) \subseteq I_w(s_j)$,
   (c) if $c \in C$ and $c\colon \to \tau \in \Omega$, then $I_w(c) \in I_w(\tau)$,
   (d) if $f \in F_n$ and $f\colon \tau_1 \times \cdots \times \tau_n \to \tau \in \Omega$, then $I_w(f)\colon I_w(\tau_1) \times \cdots \times I_w(\tau_n) \to I_w(\tau)$,
   (e) if $p \in P_n$ and $p\colon s_1 \times \cdots \times s_n \in \Omega$, then $I_w(p) \subseteq I_w(s_1) \times \cdots \times I_w(s_n)$.

By restricting Σ-models, we give the class of rigid Σ-models as follows.

Definition 10 (Rigid Σ-model) *Let Σ be a sorted signature. A rigid sorted Σ-model is a sorted Σ-model $M = (W, U, I)$ such that for all $w_i, w_j \in W$, $I_{w_i}(\tau) = I_{w_j}(\tau)$, $I_{w_i}(c) = I_{w_j}(c)$ and $I_{w_i}(f) = I_{w_j}(f)$.*

A (rigid) sorted Σ-model $M = (W, U, I)$ is said to be a (rigid) sorted Σ^+-model if the following conditions hold:

1. If $s \in T \cup N$, then $I_w(s) = I_w(p_s)$.
2. If $s_i \leq_+ s_j$, then $I_w(p_{s_i}) \subseteq I_w(p_{s_j})$.

In the sorted Σ^+-models, the interpretation of sorts s is equivalent to the interpretation of the sort predicates p_s. A variable assignment on a Σ-model $M = (W, U, I)$ is a function $\alpha_w\colon V \to U$ where $\alpha_w(x\colon s) \in I_w(s)$. The variable assignment $\alpha_w[x\colon s/d]$ is defined by $\alpha_w \backslash \{(x\colon s, \alpha_w(x\colon s))\} \cup \{(x\colon s, d)\}$. A Σ-interpretation $\mathcal{I}$ is a pair (M, α) of a Σ-model M and a set of variable assignments $\alpha = \{\alpha_w \mid w \in W\}$ on M. Let $\mathcal{I} = (M, \alpha)$. The Σ-interpretation $(M, \alpha \backslash \{\alpha_w\} \cup \{\alpha_w[x\colon s/d]\})$ is denoted by $\mathcal{I}\alpha_w[x\colon s/d]$.

Definition 11 *Let $\mathcal{I} = (M, \alpha)$ be a Σ-interpretation. The denotation $[\![\ ]\!]_{w,\alpha}\colon \mathcal{T} \to U$ is defined by the following:*

1. $[\![x\colon s]\!]_{w,\alpha} = \alpha_w(x\colon s)$,
2. $[\![c\colon \tau]\!]_{w,\alpha} = I_w(c)$,
3. $[\![f(t_1, \dots, t_n)\colon \tau]\!]_{w,\alpha} = I_w(f)([\![t_1]\!]_{w,\alpha}, \dots, [\![t_n]\!]_{w,\alpha})$.

The satisfiability of atoms, goals and clauses are defined for a Σ-interpretation $\mathcal{I}$ and a world $w \in W$.

Definition 12 (Σ-satisfiability Relation) *Let $\mathcal{I} = (M, \alpha)$ with $M = (W, U, I)$ be a Σ-interpretation, let $F \in \mathcal{A} \cup \mathcal{G} \cup \mathcal{C}$ and $w \in W$. The Σ-satisfiability relation $\mathcal{I}, w \models F$ is defined inductively as follows:*

1. $\mathcal{I}, w \models p(t_1, \dots, t_n)$ iff $([\![t_1]\!]_{w,\alpha}, \dots, [\![t_n]\!]_{w,\alpha}) \in I_w(p)$.
2. $\mathcal{I}, w \models \{L_1, \dots, L_n\}$ iff $\mathcal{I}, w \models L_1, \dots, \mathcal{I}, w \models L_n$.
3. $\mathcal{I}, w \models L \leftarrow G$ iff for all $d_1 \in I_w(s_1)$, … ,$d_n \in I_w(s_n)$, $\mathcal{I}\alpha_w[x_1\colon s_1/d_1, \dots, x_n\colon s_n/d_n], w \models G$ implies $\mathcal{I}\alpha_w[x_1\colon s_1/d_1, \dots, x_n\colon s_n/d_n], w \models L$ where $EVar\ (L \leftarrow G) = \{x_1\colon s_1, \dots, x_n\colon s_n\}$.

Let Γ be a set of formulas in $\mathcal{A} \cup \mathcal{G} \cup \mathcal{C}$. We write $\mathcal{I}, w \models \Gamma$ if, for every formula $F \in \Gamma$, $\mathcal{I}, w \models F$. A formula F is said to be Σ-satisfiable if for some Σ-interpretation $\mathcal{I}$ and world w, $\mathcal{I}, w \models F$. Otherwise, it is Σ-unsatisfiable. F is a consequence of Γ in the class of Σ-interpretations (denoted $\Gamma \models F$) if for any Σ-interpretation $\mathcal{I}$ and $w \in W$, $\mathcal{I}, w \models \Gamma$ implies $\mathcal{I}, w \models F$. A Σ-interpretation $\mathcal{I} = (M, \alpha)$ is a Σ^+-interpretation if M is a Σ^+-model. Analogously, Σ^+-satisfiability and consequences in the class of Σ^+-interpretations (denoted $\Gamma \models^+ F$) can be defined.

Knowledge Base Reasoning with Rigid Properties

In this section, we develop a knowledge base reasoning system for our order-sorted logic with rigid properties. We first extend a Horn-clause calculus (Hanus 1992) by incorporating sorted and unsorted substitutions and inference rules of type predicates. Based on the calculus, we define a rigid property derivation method for many separated knowledge bases.

Extended Horn-clause Calculus

We define a sorted substitution such that each sorted variable $x\colon s$ is replaced with a well-sorted term in $\mathcal{T}_s$.

Definition 13 (Sorted Substitution) *A sorted substitution is a partial function $\theta\colon V \rightarrow \mathcal{T}$ such that $\theta(x\colon s) \in \mathcal{T}_s - \{x\colon s\}$ and $Dom(\theta)$ is finite.*

Moreover, an unsorted substitution is defined as a substitution operationally ignoring the sort of each variable which may lead to ill-sorted terms.

Definition 14 (Unsorted Substitution) *An unsorted substitution is a partial function $\theta^u\colon V \rightarrow \mathcal{T}$ such that $\theta^u(x\colon s) \in \mathcal{T} - \{x\colon s\}$ and $Dom(\theta^u)$ is finite.*

For example, for $student \leq_+ person$, $\theta^u(x\colon student) = john\colon person$ is an unsorted substitution, but not a sorted substitution. Each of sorted and unsorted substitutions can be represented by $\{x_1\colon s_1/t_1, \dots, x_n\colon s_n/t_n\}$. Let θ be a sorted substitution. θ is called a ground sorted substitution if for every variable $x\colon s \in Dom(\theta)$, $\theta(x\colon s)$ is a ground sorted term. The restriction of a sorted substitution θ to $V'(\subseteq V)$ is defined by $\theta\!\uparrow\! V' = \{x\colon s/\theta(x\colon s) \mid x\colon s \in V' \cap Dom(\theta)\}$. θ is a ground sorted substitution for V' if $V' \subseteq Dom(\theta)$ and $\theta\!\uparrow\! V'$ is a ground sorted substitution. The identity substitution is denoted by ϵ (i.e. $Dom(\epsilon) = \emptyset$). Similarly, these notions are defined for unsorted substitutions θ^u.

In the usual manner of logic, sorted and unsorted substitutions are extended to terms, atoms, goals and clauses. Let E be an expression, θ be a sorted substitution and θ^u be an unsorted substitution. The sorted and unsorted substitutions to variables in E are denoted by $E\theta$ and $E\theta^u$.

Definition 15 *Let $E \in \mathcal{T} \cup \mathcal{A} \cup \mathcal{G} \cup \mathcal{C}$ and let γ be a sorted substitution θ or an unsorted substitution θ^u. The expression $E\gamma$ is defined by the following:*

1. $x\colon s\gamma = \gamma(x\colon s)$ *if* $x\colon s \in Dom(\gamma)$,
2. $x\colon s\gamma = x\colon s$ *if* $x\colon s \notin Dom(\gamma)$,
3. $f(t_1, \dots, t_n)\colon \tau\gamma = f(t_1\gamma, \dots, t_n\gamma)\colon \tau$,
4. $p(t_1, \dots, t_n)\gamma = p(t_1\gamma, \dots, t_n\gamma)$,
5. $\{L_1, \dots, L_n\}\gamma = \{L_1\gamma, \dots, L_n\gamma\}$,
6. $(L \leftarrow G)\gamma = L\gamma \leftarrow G\gamma$.

Let t be a sorted term. The term $t\theta^u$ is called an ill-sorted term if $t\theta^u \notin \mathcal{T}$. θ (resp. θ^u) is a ground sorted substitution (resp. ground unsorted substitution) for E if $E\theta$ (resp. $E\theta^u$) is ground. The composition $\theta_1\theta_2$ of sorted substitutions θ_1 and θ_2 (resp. unsorted substitutions θ_1^u and θ_2^u) is defined by $(x\colon s)\theta_1\theta_2 = ((x\colon s)\theta_1)\theta_2$ (resp. $(x\colon s)\theta_1^u\theta_2^u = ((x\colon s)\theta_1^u)\theta_2^u$).

Lemma 1 *Let $\mathcal{I} = (M, \alpha)$ with $M = (W, U, I)$ be a Σ-interpretation, $L \leftarrow G$ be a clause, θ be a sorted substitution and let $w \in W$. If $\mathcal{I}, w \models L \leftarrow G$, then $\mathcal{I}, w \models (L \leftarrow G)\theta$.*

Proof. Suppose $\mathcal{I}, w \models L \leftarrow G$. By Definition 12, for all $d_1 \in I_w(s_1), \dots, d_n \in I_w(s_n)$, $\mathcal{I}\alpha_w[x_1\colon s_1/d_1, \dots, x_n\colon s_n/d_n], w \models G$ implies $\mathcal{I}\alpha_w[x_1\colon s_1/d_1, \dots, x_n\colon s_n/d_n], w \models L$. Let $d_1' \in I_w(s_1'), \dots, d_m' \in I_w(s_m')$ where $EVar((L \leftarrow G)\theta) = \{y_1\colon s_1', \dots, y_m\colon s_m'\}$, and let $d_i = [\![\theta(x_i\colon s_i)]\!]_{w,\alpha'}$ for $\alpha' = \alpha \backslash \{\alpha_w\} \cup \{\alpha_w[y_1\colon s_1'/d_1', \dots, y_m\colon s_m'/d_m']\}$. This derives $\mathcal{I}, w \models (L \leftarrow G)\theta$. ∎

Definition 16 (Knowledge Base) *Let $\Sigma = (T, N, \Omega, \leq, \leq_+)$ be a sorted signature. A knowledge base $\mathcal{K}$ is a finite set of clauses in $\mathcal{C}$.*

Let θ be a sorted substitution and C be a clause. $C\theta$ is called an instance of C. The set of all ground instances of C is denoted by $ground(C)$. In the following, we define inference rules in an extended Horn-clause calculus with sorted and unsorted substitutions.

Definition 17 (Sorted Horn-clause Calculus) *Let C be a ground clause and $\mathcal{K}$ be a knowledge base. A derivability of C from $\mathcal{K}$ (denoted $\mathcal{K} \vdash C$) in the sorted Horn-clause calculus is defined as follows:*

- **Sorted substitution rule:** *Let $L \leftarrow G \in \mathcal{K}$ and θ is a ground substitution for $L \leftarrow G$. $\mathcal{K} \vdash (L \leftarrow G)\theta$*
- **Cut rule:** *Let $L \leftarrow G$ and $L' \leftarrow G' \cup \{L\}$ be ground clauses. If $\mathcal{K} \vdash L \leftarrow G$ and $\mathcal{K} \vdash L' \leftarrow G' \cup \{L\}$, then $\mathcal{K} \vdash L' \leftarrow G \cup G'$*
- **Subsort rule:** *Let $p_s(t) \leftarrow G$ and $p_{s'}(t) \leftarrow G$ be ground clauses. If $\mathcal{K} \vdash p_s(t) \leftarrow G$ and $s <_+ s'$, then $\mathcal{K} \vdash p_{s'}(t) \leftarrow G$.*
- **Type predicate rule:** *Let t be a ground sorted term. If $sort(t) \leq_+ \tau$, then $\mathcal{K} \vdash p_\tau(t) \leftarrow$.*
- **Unsorted type predicate rule:** *Let t be a sorted term where $EVar(t) = \{x_1\colon s_1, \dots, x_n\colon s_n\}$ and let $p_{s_i}(t_i) \leftarrow G_i$ be a ground clause. If $sort(t) \leq_+ \tau$ and $\mathcal{K} \vdash p_{s_1}(t_1) \leftarrow G_1, \dots, \mathcal{K} \vdash p_{s_n}(t_n) \leftarrow G_n$, then $\mathcal{K} \vdash p_\tau(t)\{x_1\colon s_1/t_1, \dots, x_n\colon s_n/t_n\} \leftarrow G_1 \cup \dots \cup G_n$.*
- **Unsorted substitution rule:** *Let $L \leftarrow G \in \mathcal{K}$ where $\{x_1\colon s_1, \dots, x_n\colon s_n\} \subseteq EVar(L \leftarrow G)$ and let $p_{s_i}(t_i) \leftarrow G_i$ be a ground clause. If $\mathcal{K} \vdash p_{s_1}(t_1) \leftarrow G_1, \dots, \mathcal{K} \vdash p_{s_n}(t_n) \leftarrow G_n$, then $\mathcal{K} \vdash (L \leftarrow G \cup G_1 \cup \dots \cup G_n)\{x_1\colon s_1/t_1, \dots, x_n\colon s_n/t_n\}\theta$ where θ is a ground sorted substitution for $(L \leftarrow G \cup G_1 \cup \dots \cup G_n)\{x_1\colon s_1/t_1, \dots, x_n\colon s_n/t_n\}$.*

We write $\mathcal{K} \vdash L$ if $\mathcal{K} \vdash L \leftarrow$. The sorted substitution rule and the cut rule serve as sorted inference rules in ordinary order-sorted logic. The subsort rule actualizes an inference of the implication form $p_s(t) \Rightarrow p_{s'}(t)$ with respect to the subsort relation $s <_+ s'$. The type predicate rule derives axioms $p_\tau(t)$ of type predicates with $sort(t) \leq_+ \tau$. For example, $animal(john\colon person)$ is valid if $person \leq_+ animal$. Furthermore, the unsorted type predicate rule and the unsorted substitution rule are unsorted variants of the type predicate rule and the sorted substitution rule respectively. These inference rules are based on the sorted resolution system with sort predicates in (Beierle *et al.* 1992). In order to deal with the rigidity of sorts, we distinguish types and non-rigid sorts in the calculus. In particular, the unsorted type predicate rule and the unsorted substitution rule are necessary for reasoning over non-rigid sorted terms. For a non-rigid sort $student \in N$, if $student(john\colon person)$ is true, then $x\colon student$ can be unsortedly substituted with a sorted term $john\colon person$ while the sort of $john\colon person$ is not a subsort of $student$.

Example 2 *Suppose we have the sorted signature* $\Sigma = (T, N, \Omega, \leq^*, \leq^*_+)$ *of Example 1. Consider the following knowledge bases* $\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_3, \mathcal{K}_4$*:*

$$\begin{aligned}
\mathcal{K}_1 &= \{\, male\_student(john\colon person) \leftarrow, \\
&\quad excellent(john\colon person) \leftarrow, \\
&\quad getting\_a\_scholarship(x\colon student) \\
&\qquad\qquad \leftarrow \{excellent(x\colon student)\}\, \} \\
\mathcal{K}_2 &= \{\, teacher(mary\colon person) \leftarrow, \\
&\quad likes(mary\colon person, x\colon student) \leftarrow, \\
&\quad likes(mary\colon person, x\colon bird) \leftarrow\ \} \\
\mathcal{K}_3 &= \{\, canary(peter\colon animal) \leftarrow, \\
&\quad canfly(x\colon animal) \leftarrow \{bird(x\colon animal)\}\, \} \\
\mathcal{K}_4 &= \{\, father(tony\colon animal, peter\colon animal) \leftarrow, \\
&\quad bird(y\colon animal) \leftarrow \{bird(x\colon animal), \\
&\quad father(y\colon animal, x\colon animal)\}\, \}
\end{aligned}$$

Figure 3 shows derivations from $\mathcal{K}_1,\ \mathcal{K}_3$ *in the sorted Horn-clause calculus. In the first we can derive the ill-sorted expression*

$$getting\_a\_scholarship(john\colon person)$$

where for $person \in T$ *and* $student \in N$*,* $person \not\leq_+ student$ *and* $getting\_a\_scholarship\colon student \in \Omega$*. This is obtained by an application of the unsorted substitution rule to the clauses*

$$\begin{aligned}
&student(john\colon person) \leftarrow, \\
&getting\_a\_scholarship(x\colon student) \\
&\qquad\qquad \leftarrow \{excellent(x\colon student)\}
\end{aligned}$$

with the unsorted substitution

$$\theta^u = \{x\colon student/john\colon person\}.$$

Rigid Property Derivation in Knowledge Bases

Normally, the conclusions in each knowledge base are not derivable from another knowledge base, since knowledge bases are separately constructed for their respective situations. Nevertheless, each knowledge base can derive something from rigid property information in other knowledge bases. For this idea, we present a derivation method of rigid properties in a finite set of knowledge bases (called *rigid property derivation*).

Let $\mathcal{S} = \{\mathcal{K}_1, \ldots, \mathcal{K}_n\}$ be a finite set of knowledge bases where $\mathcal{K}_1, \ldots, \mathcal{K}_n$ have the same sorted signature Σ. A Σ-model $M = (W_{\mathcal{S}}, U, I)$ is said to be a knowledge base Σ-model (or simply KB Σ-model) for $\mathcal{S}$ if $W_{\mathcal{S}} = \{w_{\mathcal{K}_1}, \ldots, w_{\mathcal{K}_n}\}$ is the set of knowledge base worlds of $\mathcal{K}_1, \ldots, \mathcal{K}_n$. Afterwards we assume that every model is a KB Σ-model. A (rigid) Σ-interpretation is called a (rigid) KB Σ-interpretation if its model is a (rigid) KB Σ-model. We denote $\mathcal{K}_i \models^+_{\mathcal{S}} F$ if for every rigid KB Σ^+-interpretation $\mathcal{I}$ for $\mathcal{S}, \mathcal{I}, w_{\mathcal{K}_1} \models \mathcal{K}_1, \ldots, \mathcal{I}, w_{\mathcal{K}_n} \models \mathcal{K}_n$ implies $\mathcal{I}, w_{\mathcal{K}_i} \models F$. Let us denote by $\mathcal{A}_0$ the set of ground atoms. We define the theory of $\mathcal{K}$ as $Th(\mathcal{K}) = \{L \in \mathcal{A}_0 \mid \mathcal{K} \vdash L\}$.

In order to define rigid property derivation, an expansion of knowledge bases is introduced. The expansion of each knowledge base is obtained from other knowledge bases by extracting rigid property information.

Definition 18 (Expansion of Knowledge Bases) *Let* $\mathcal{S} = \{\mathcal{K}_1, \ldots, \mathcal{K}_n\}$ *be a finite set of knowledge bases. The expanded knowledge bases* $\mathcal{K}^m_i$ *of* $\mathcal{K}_i$ *in* $\mathcal{S}$ *are defined by the following:*

$$\begin{aligned}
\mathcal{K}^0_i &= \mathcal{K}_i \\
\mathcal{K}^{m+1}_i &= \mathcal{K}^m_i \cup \Delta(Th(\mathcal{K}^m_1) \cup \cdots \cup Th(\mathcal{K}^m_n))
\end{aligned}$$

where $\Delta(X) = \{p_\tau(t) \in \mathcal{A}_0 \mid \tau \in T \text{ and } p_\tau(t) \in X\}$

Each knowledge base $\mathcal{K}_i$ is expanded to $\mathcal{K}^0_i, \mathcal{K}^1_i, \ldots$ by adding rigid atomic formulas $p_\tau(t)$ derivable in one of the knowledge bases $\mathcal{K}_1, \ldots, \mathcal{K}_n \in \mathcal{S}$. Note that the expansion is not dependent upon an ordering of $\mathcal{S}$, namely, the final expansion of all elements of $\mathcal{S}$ is uniquely obtained, regardless of the ordering. Using this expansion, we define rigid property derivation in a finite set of knowledge bases.

Definition 19 (Rigid Property Derivation in $\mathcal{S}$) *Let* $\mathcal{S} = \{\mathcal{K}_1, \ldots, \mathcal{K}_n\}$ *be a finite set of knowledge bases. A ground atom* L *is derivable from* $\mathcal{K}_i$ *in* $\mathcal{S}$ *(denoted* $\mathcal{K}_i \vdash_{\mathcal{S}} L$ *) if there exists an expanded knowledge base* $\mathcal{K}^m_i$ *of* $\mathcal{K}_i$ *such that* $\mathcal{K}^m_i \vdash L$*.*

Let us consider rigid property derivation in the knowledge bases $\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_3, \mathcal{K}_4$ of Example 2.

Example 3 *Suppose we have the sorted signature* $\Sigma = (T, N, \Omega, \leq^*, \leq^*_+)$ *of Example 1 and the knowledge bases* $\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_3, \mathcal{K}_4$ *of Example 2. By the expanded knowledge bases* $\mathcal{K}^0_i, \mathcal{K}^1_i, \ldots$ *of each* $\mathcal{K}_i$ *in* $\mathcal{S} = \{\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_3, \mathcal{K}_4\}$*, the following derivations can be obtained.*

Initially, for the knowledge bases $\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_3, \mathcal{K}_4$*, we have the theories* $Th(\mathcal{K}_1), Th(\mathcal{K}_2), Th(\mathcal{K}_3), Th(\mathcal{K}_4)$ *as follows:*

$$\begin{aligned}
Th(\mathcal{K}_1) &= \Gamma \cup \{\, excellent(john\colon person), \\
&\quad getting\_a\_scholarship(john\colon person), \\
&\quad male\_student(john\colon person), \\
&\quad student(john\colon person), \\
&\quad male(john\colon person)\, \} \\
Th(\mathcal{K}_2) &= \Gamma \cup \{\, teacher(mary\colon person)\, \} \\
Th(\mathcal{K}_3) &= \Gamma \cup \{\, canfly(peter\colon animal), \\
&\quad bird(peter\colon animal), \\
&\quad canary(peter\colon animal)\, \} \\
Th(\mathcal{K}_4) &= \Gamma \cup \\
&\quad \{\, father(tony\colon animal, peter\colon animal)\, \}
\end{aligned}$$

where

$$\begin{aligned}
\Gamma = \{\, & person(john\colon person),\ animal(john\colon person), \\
& person(mary\colon person),\ animal(mary\colon person), \\
& animal(peter\colon animal),\ animal(tony\colon animal)\, \}.
\end{aligned}$$

Note that the theories $Th(\mathcal{K}_1)$ *and* $Th(\mathcal{K}_3)$ *contain the rigid atomic formulas* $male(john\colon person)$*; and* $bird(peter\colon animal)$*,* $canary(peter\colon animal)$ *respectively. In the following steps, the knowledge bases* $\mathcal{K}^0_1$*,* $\mathcal{K}^0_2$*,* $\mathcal{K}^0_3, \mathcal{K}^0_4$ *are expanded by adding these rigid atomic formulas:*

$\mathcal{K}_1 \vdash \mathit{getting\_a\_scholarship}(\mathit{john}\colon \mathit{person})$:

$$\dfrac{\mathit{excellent}(\mathit{john}\colon \mathit{person}) \leftarrow \qquad \dfrac{\dfrac{\mathit{male\_student}(\mathit{john}\colon \mathit{person}) \leftarrow}{\mathit{student}(\mathit{john}\colon \mathit{person}) \leftarrow}}{\mathit{getting\_a\_scholarship}(\mathit{john}\colon \mathit{person}) \leftarrow \{\mathit{excellent}(\mathit{john}\colon \mathit{person})\}}}{\mathit{getting\_a\_scholarship}(\mathit{john}\colon \mathit{person}) \leftarrow}$$

$\mathcal{K}_3 \vdash \mathit{canfly}(\mathit{peter}\colon \mathit{animal})$:

$$\dfrac{\mathit{canfly}(\mathit{peter}\colon \mathit{animal}) \leftarrow \{\mathit{bird}(\mathit{peter}\colon \mathit{animal})\} \qquad \dfrac{\mathit{canary}(\mathit{peter}\colon \mathit{animal}) \leftarrow}{\mathit{bird}(\mathit{peter}\colon \mathit{animal}) \leftarrow}}{\mathit{canfly}(\mathit{peter}\colon \mathit{animal}) \leftarrow}$$

Figure 3: An example of derivations

Step 1:
$\mathcal{K}_i^0 = \mathcal{K}_i \ (1 \le i \le 4)$

Step 2:
$\mathcal{K}_i^1 = \mathcal{K}_i^0 \cup \Delta(T^0) \ (1 \le i \le 4)$ *where*
$T^0 = Th(\mathcal{K}_1) \cup Th(\mathcal{K}_2) \cup Th(\mathcal{K}_3) \cup Th(\mathcal{K}_4)$ *and*
$\Delta(T^0) = \{ \ \mathit{male}(\mathit{john}\colon \mathit{person}),$
$\quad \mathit{bird}(\mathit{peter}\colon \mathit{animal}),\ \mathit{canary}(\mathit{peter}\colon \mathit{animal}) \ \}$

Step 3:
$Th(\mathcal{K}_1^1) = Th(\mathcal{K}_1) \cup \Delta(T^0)$
$Th(\mathcal{K}_2^1) = Th(\mathcal{K}_2) \cup \Delta(T^0)$
$\quad \cup \{ \ \mathit{likes}(\mathit{mary}\colon \mathit{person}, \mathit{peter}\colon \mathit{animal}) \ \}$
$Th(\mathcal{K}_3^1) = Th(\mathcal{K}_3) \cup \Delta(T^0)$
$Th(\mathcal{K}_4^1) = Th(\mathcal{K}_4) \cup \Delta(T^0) \cup \{ \ \mathit{bird}(\mathit{tony}\colon \mathit{animal}) \ \}$

In Step 3, as a result of the expansion, the new conclusions $\mathit{likes}(\mathit{mary}\colon \mathit{person}, \mathit{peter}\colon \mathit{animal})$ *in* $Th(\mathcal{K}_2^1)$ *and* $\mathit{bird}(\mathit{tony}\colon \mathit{animal})$ *in* $Th(\mathcal{K}_4^1)$ *can be derived in the Horn-clause calculus. Moreover, the rigid atomic formula* $\mathit{bird}(\mathit{tony}\colon \mathit{animal})$ *expands the knowledge bases* $\mathcal{K}_1^1, \mathcal{K}_2^1, \mathcal{K}_3^1, \mathcal{K}_4^1$ *as follows:*

Step 4:
$\mathcal{K}_i^2 = \mathcal{K}_i^1 \cup \Delta(T^1) \ (1 \le i \le 4)$ *where*
$T^1 = Th(\mathcal{K}_1^1) \cup Th(\mathcal{K}_2^1) \cup Th(\mathcal{K}_3^1) \cup Th(\mathcal{K}_4^1)$ *and*
$\Delta(T^1) = \Delta(T^0) \cup \{ \ \mathit{bird}(\mathit{tony}\colon \mathit{animal}) \ \}$

Step 5:
$Th(\mathcal{K}_1^2) = Th(\mathcal{K}_1^1) \cup \Delta(T^1)$
$Th(\mathcal{K}_2^2) = Th(\mathcal{K}_2^1) \cup \Delta(T^1)$
$\quad \cup \{ \ \mathit{likes}(\mathit{mary}\colon \mathit{person}, \mathit{tony}\colon \mathit{animal}) \ \}$
$Th(\mathcal{K}_3^2) = Th(\mathcal{K}_3^1) \cup \Delta(T^1)$
$\quad \cup \{ \ \mathit{canfly}(\mathit{tony}\colon \mathit{animal}) \ \}$
$Th(\mathcal{K}_4^2) = Th(\mathcal{K}_4^1) \cup \Delta(T^1)$

In Step 5, the further new results $\mathit{likes}(\mathit{mary}\colon \mathit{person}, \mathit{tony}\colon \mathit{animal})$ *in* $Th(\mathcal{K}_2^2)$ *and* $\mathit{canfly}(\mathit{tony}\colon \mathit{animal})$ *in* $Th(\mathcal{K}_3^2)$ *are generated from the expanded knowledge bases* $\mathcal{K}_1^2, \mathcal{K}_2^2, \mathcal{K}_3^2, \mathcal{K}_4^2$.

Step 6:
$\mathcal{K}_i^3 = \mathcal{K}_i^2 \cup \Delta(T^2) \ (1 \le i \le 4)$ *where*
$T^2 = Th(\mathcal{K}_1^2) \cup Th(\mathcal{K}_2^2) \cup Th(\mathcal{K}_3^2) \cup Th(\mathcal{K}_4^2)$ *and*
$\Delta(T^2) = \Delta(T^1)$.

The derivation terminates in Step 6 because $\mathcal{K}_1^2, \mathcal{K}_2^2, \mathcal{K}_3^2, \mathcal{K}_4^2$ *are not expanded anymore.*

In this example, the following conclusions are derivable from each knowledge base in $\mathcal{S}$.

$$\begin{array}{lll} \mathcal{K}_1 & \vdash_{\mathcal{S}} & \mathit{student}(\mathit{john}\colon \mathit{person}) \\ \mathcal{K}_2 & \not\vdash_{\mathcal{S}} & \mathit{likes}(\mathit{mary}\colon \mathit{person}, \mathit{john}\colon \mathit{person}) \end{array}$$

However, the knowledge base $\mathcal{K}_2$ cannot extract the instantiation $\mathit{student}(\mathit{john}\colon \mathit{person})$ from the knowledge base $\mathcal{K}_1$ because the sort $\mathit{student}$ is not rigid. This means that we do not find whether or not John is a student in the situation of the knowledge base $\mathcal{K}_2$.

$$\begin{array}{rll} \mathcal{K}_2 & \vdash_{\mathcal{S}} & \mathit{likes}(\mathit{mary}\colon \mathit{person}, \mathit{peter}\colon \mathit{animal}) \\ (\text{but } \mathcal{K}_2 & \not\vdash & \mathit{likes}(\mathit{mary}\colon \mathit{person}, \mathit{peter}\colon \mathit{animal})) \\ \mathcal{K}_2 & \vdash_{\mathcal{S}} & \mathit{likes}(\mathit{mary}\colon \mathit{person}, \mathit{tony}\colon \mathit{animal}) \\ (\text{but } \mathcal{K}_2 & \not\vdash & \mathit{likes}(\mathit{mary}\colon \mathit{person}, \mathit{tony}\colon \mathit{animal})) \\ \mathcal{K}_3 & \vdash_{\mathcal{S}} & \mathit{canfly}(\mathit{tony}\colon \mathit{animal}) \\ (\text{but } \mathcal{K}_3 & \not\vdash & \mathit{canfly}(\mathit{tony}\colon \mathit{animal})) \end{array}$$

These were not derivable in the Horn-clause calculus without rigid property derivation. Our method can derive them from the knowledge bases $\mathcal{K}_2^2, \mathcal{K}_3^2$ expanded by means of extracting the rigid property information that Tony and Peter are birds, i.e.,

$$\begin{array}{lll} \mathcal{K}_4 & \vdash_{\mathcal{S}} & \mathit{bird}(\mathit{tony}\colon \mathit{animal}) \\ \mathcal{K}_3 & \vdash_{\mathcal{S}} & \mathit{bird}(\mathit{peter}\colon \mathit{animal}). \end{array}$$

Completeness of Rigid Property Derivation

In this section, we prove the completeness of the sorted Horn-clause calculus and the rigid property derivation for many knowledge bases.

Theorem 1 (Soundness of the Horn-clause Calculus) *Let $\mathcal{S} = \{\mathcal{K}_1, \dots, \mathcal{K}_n\}$ be a finite set of knowledge bases and L be a ground sorted or ill-sorted atom. If $\mathcal{K}_i \vdash L$, then $\mathcal{K}_i \models^+ L$.*

The set of derivable sorted and ill-sorted terms for sort s in $\mathcal{K}_i$ is defined as $\mathcal{T}_{0,s}^{\mathcal{K}_i} = \{t \mid \mathcal{K}_i \vdash p_s(t)\}$. We define the set of derivable terms for all sorts in $\mathcal{K}_i$ by $\mathcal{T}_0^{\mathcal{K}_i} = \bigcup_{s \in T \cup N} \mathcal{T}_{0,s}^{\mathcal{K}_i}$. To prove the completeness of the Horn-clause calculus and the rigid property derivation, we construct a Herbrand model for a finite set of knowledge bases.

Definition 20 (Herbrand Model) *Let $\mathcal{S} = \{\mathcal{K}_1, \dots, \mathcal{K}_n\}$ be a finite set of knowledge bases. A Herbrand model M_H for $\mathcal{S}$ is a tuple $(W_{\mathcal{S}}, U_H, I_H)$ such that*

1. $W_{\mathcal{S}} = \{w_{\mathcal{K}_1}, \dots, w_{\mathcal{K}_n}\}$,
2. $U_H = \bigcup_{\mathcal{K}_i \in \mathcal{S}} \mathcal{T}_0^{\mathcal{K}_i}$,
3. $I_H = \{I_{w_{\mathcal{K}_i}} \mid w_{\mathcal{K}_i} \in W_{\mathcal{S}}\}$ *is the set of interpretation functions for all* $w_{\mathcal{K}_i} \in W_{\mathcal{S}}$ *with the following conditions:*
   (a) $I_{w_{\mathcal{K}_i}}(s) = \mathcal{T}_{0,s}^{\mathcal{K}_i}$,
   (b) *if* $c \in C$ *and* $c\colon \to \tau \in \Omega$, *then* $I_{w_{\mathcal{K}_i}}(c) = c\colon \tau$,
   (c) *if* $f \in F_n$ *and* $f\colon \tau_1 \times \cdots \times \tau_n \to \tau \in \Omega$, *then* $I_{w_{\mathcal{K}_i}}(f)(t_1, \dots, t_n) = f(t_1, \dots, t_n)\colon \tau$ *where* $t_k \in I_{w_{\mathcal{K}_i}}(\tau_k)$ *for* $1 \leq k \leq n$, *and*
   (d) *if* $p \in P_n$ *and* $p\colon s_1 \times \cdots \times s_n \in \Omega$, *then* $I_{w_{\mathcal{K}_i}}(p) \subseteq I_{w_{\mathcal{K}_i}}(s_1) \times \cdots \times I_{w_{\mathcal{K}_i}}(s_n)$.

The set of rigid derivable sorted and ill-sorted terms for sort s in $\mathcal{K}_i$ is defined as $\mathcal{RT}_{0,s}^{\mathcal{K}_i} = \{t \mid \mathcal{K}_i \vdash_{\mathcal{S}} p_s(t)\}$. We define the set of rigid derivable terms for all sorts in $\mathcal{K}_i$ by $\mathcal{RT}_0^{\mathcal{K}_i} = \bigcup_{s \in T \cup N} \mathcal{RT}_{0,s}^{\mathcal{K}_i}$.

Definition 21 (Rigid Herbrand Model) *A rigid Herbrand model M_H for $\mathcal{S}$ is a tuple $(W_{\mathcal{S}}, U_H, I_H)$ with $U_H = \bigcup_{\mathcal{K}_i \in \mathcal{S}} \mathcal{RT}_0^{\mathcal{K}_i}$, $I_{w_{\mathcal{K}_i}}(s) = \mathcal{RT}_{0,s}^{\mathcal{K}_i}$ and the other conditions of Herbrand models.*

A (rigid) Herbrand interpretation $\mathcal{I}_H$ for $\mathcal{S}$ is a pair (M_H, α) such that M_H is a (rigid) Herbrand model for $\mathcal{S}$ and α is a set of variable assignments on M_H. We define $ground^+(L \leftarrow G) = ground(L \leftarrow G) \cup \bigcup_{\theta^u} ground((L \leftarrow G)\theta^u)$ with $\theta^u = \{x_1\colon s_1/t_1, \dots, x_n\colon s_n/t_n\}$ where $\mathcal{K}_i \vdash p_{s_k}(t_k)$ for $1 \leq k \leq n$ and $\{x_1\colon s_1, \dots, x_n\colon s_n\} \subseteq EVar(L \leftarrow G)$. In the following, we define a canonical interpretation for a finite set of knowledge bases.

Definition 22 (Canonical Interpretation for $\mathcal{S}$) *Let $\mathcal{S} = \{\mathcal{K}_1, \dots, \mathcal{K}_n\}$ be a finite set of knowledge bases and L be a ground atom. A canonical interpretation (resp. rigid canonical interpretation) for $\mathcal{S}$ is a Herbrand interpretation (resp. rigid Herbrand interpretation) $\mathcal{I}_{\mathcal{S}} = (M_H, \alpha)$ for $\mathcal{S}$ such that*

$\mathcal{I}_{\mathcal{S}}, w_{\mathcal{K}_i} \models L$ iff $\mathcal{K}_i \vdash L$ (resp. $\mathcal{K}_i \vdash_{\mathcal{S}} L$).

We show that this canonical interpretation is a (KB) Σ^+-interpretation and satisfies each knowledge base $\mathcal{K}_i$ in $\mathcal{S}$.

Lemma 2 *Let $\mathcal{S} = \{\mathcal{K}_1, \dots, \mathcal{K}_n\}$ be a finite set of knowledge bases. Let $\mathcal{I}_{\mathcal{S}}$ be a canonical interpretation for $\mathcal{S}$ and $L \leftarrow G$ be a clause. Then, the following statements hold:*

1. *$\mathcal{I}_{\mathcal{S}}, w_{\mathcal{K}_i} \models L \leftarrow G$ if and only if $\mathcal{I}_{\mathcal{S}}, w_{\mathcal{K}_i} \models ground^+ (L \leftarrow G)$.*
2. *$\mathcal{I}_{\mathcal{S}}$ is a (KB) Σ^+-interpretation of $\mathcal{K}_i$.*

Proof. 1. ($\Rightarrow$) By Lemma 1, $\mathcal{I}_{\mathcal{S}}, w_{\mathcal{K}_i} \models (L \leftarrow G)\theta$ ($\in ground(L \leftarrow G)$). Let $(L \leftarrow G)\theta^u\theta \in ground^+((L \leftarrow G)\theta^u)$ with $\theta^u = \{x_1\colon s_1/t_1, \dots, x_n\colon s_n/t_n\}$. By Definition 20, we have $t_1 \in I_{w_{\mathcal{K}_i}}(s_1), \dots, t_n \in I_{w_{\mathcal{K}_i}}(s_n)$. Hence, $\mathcal{I}_{\mathcal{S}}, w_{\mathcal{K}_i} \models (L \leftarrow G)\theta^u$. Therefore, by Lemma 1, $\mathcal{I}_{\mathcal{S}}, w_{\mathcal{K}_i} \models (L \leftarrow G)\theta^u\theta$. ($\Leftarrow$) Let $\theta_0(x_k\colon s_k) = t_k$ for any ground term t_k in $I_{w_{\mathcal{K}_i}}(s_k)$ where $Dom(\theta_0) = EVar(L \leftarrow G) (= \{x_1\colon s_1, \dots, x_n\colon s_n\})$. We divide it by $\theta = \{(x_k\colon s_k, \theta_0(x_k\colon s_k)) \mid \theta_0(x_k\colon s_k) \in \mathcal{T}_{0,s_k}\}$ and $\theta^u = \theta_0 - \theta$. So, we obtain $(L \leftarrow G)\theta^u\theta \in ground^+(L \leftarrow G)$. By assumption, for all $t_1 \in I_{w_{\mathcal{K}_i}}(s_1), \dots, t_n \in I_{w_{\mathcal{K}_i}}(s_n)$, $\mathcal{I}\alpha_w[x_1\colon s_1/t_1, \dots, x_n\colon s_n/t_n], w_{\mathcal{K}_i} \models G$ implies $\mathcal{I}\alpha_w[x_1\colon s_1/t_1, \dots, x_n\colon s_n/t_n], w_{\mathcal{K}_i} \models L$.

2. We show that $\mathcal{I}_{\mathcal{S}}$ is a Σ-interpretation. By Definition 20, the conditions 1,2,3-(a) and 3-(e) of Σ-models (Definition 9) hold. The condition 3-(c) is shown by the following. If $c \in C$ and $c\colon \to \tau \in \Omega$, then we have $I_{w_{\mathcal{K}_i}}(c) = c\colon \tau$ (by Definition 20 3-(b)). By the type predicate rule in the Horn-clause calculus, $\mathcal{K}_i \vdash p_\tau(c\colon \tau)$. So, by Definition 20 3-(a), $c\colon \tau \in I_{w_{\mathcal{K}_i}}(\tau)$. Thus, $I_{w_{\mathcal{K}_i}}(c) \in I_{w_{\mathcal{K}_i}}(\tau)$. The conditions 3-(b) and 3-(d) can be shown by the subsort rule and unsorted type predicate rule. Furthermore, by the definition of $\mathcal{T}_{0,s}^{\mathcal{K}_i}$ and Definition 22, we can derive that $\mathcal{I}_{\mathcal{S}}$ is a Σ^+-interpretation.

Next, we prove that $\mathcal{I}_{\mathcal{S}}$ satisfies $\mathcal{K}_i$. Let $L \leftarrow G \in \mathcal{K}_i$ where $EVar(L \leftarrow G) = \{x_1\colon s_1, \dots, x_n\colon s_n\}$. So we want to show $\mathcal{I}_{\mathcal{S}}, w_{\mathcal{K}_i} \models ground^+(L \leftarrow G)$. Case 1: let $(L \leftarrow G)\theta \in ground^+(L \leftarrow G)$ where $\theta = \{x_1\colon s_1/t_1, \dots, x_n\colon s_n/t_n\}$ is a ground sorted substitution for $L \leftarrow G$. Suppose $\mathcal{I}_{\mathcal{S}}, w_{\mathcal{K}_i} \models \{L_1, \dots, L_n\}\theta$ where $G = \{L_1, \dots, L_n\}$. By definition 22, $\mathcal{K}_i \vdash L_1\theta, \dots, \mathcal{K}_i \vdash L_n\theta$. So, we have $\mathcal{K}_i \vdash (L \leftarrow \{L_1, \dots, L_n\})\theta$ (by the sorted substitution rule). By the cut rule, $\mathcal{K}_i \vdash L\theta$ is derivable. Hence, by Definition 22, $\mathcal{I}_{\mathcal{S}}, w_{\mathcal{K}_i} \models L\theta$. Therefore, $\mathcal{I}_{\mathcal{S}}, w_{\mathcal{K}_i} \models (L \leftarrow G)\theta$. Case 2: let $(L \leftarrow G)\theta^u\theta \in ground^+(L \leftarrow G)$ where $\theta^u = \{x_1\colon s_1/t_1, \dots, x_n\colon s_n/t_n\}$ is a ground unsorted substitution such that $\mathcal{K}_i \vdash p_{s_k}(t_k)$ for $1 \leq k \leq n$ and $\{x_1\colon s_1, \dots, x_n\colon s_n\} \subseteq EVar(L \leftarrow G)$, and θ is a ground sorted substitution for $(L \leftarrow G)\theta^u$. Suppose $\mathcal{I}_{\mathcal{S}}, w_{\mathcal{K}_i} \models \{L_1, \dots, L_n\}\theta^u\theta$ where $G = \{L_1, \dots, L_n\}$. By definition 22, $\mathcal{K}_i \vdash L_1\theta^u\theta, \dots, \mathcal{K}_i \vdash L_n\theta^u\theta$. Then, $\mathcal{K}_i \vdash (L \leftarrow \{L_1, \dots, L_n\})\theta^u\theta$ (by the unsorted substitution rule and $\mathcal{K}_i \vdash p_{s_k}(t_k)$ for $1 \leq k \leq n$). By the cut rule, $\mathcal{K}_i \vdash L\theta^u\theta$. Hence, by Definition 22, $\mathcal{I}_{\mathcal{S}}, w_{\mathcal{K}_i} \models L\theta^u\theta$. Therefore, $\mathcal{I}_{\mathcal{S}}, w_{\mathcal{K}_i} \models (L \leftarrow G)\theta^u\theta$. By the first statement in Lemma 2, we obtain $\mathcal{I}_{\mathcal{S}}, w_{\mathcal{K}_i} \models L \leftarrow G$. ∎

Theorem 2 (Completeness of the Horn-clause Calculus) *Let $\mathcal{S} = \{\mathcal{K}_1, \dots, \mathcal{K}_n\}$ be a finite set of knowledge bases* *and L be a ground sorted or ill-sorted atom. If $\mathcal{K}_i \models^+ L$, then $\mathcal{K}_i \vdash L$.*

Proof. Suppose $\mathcal{K}_i \models^+ L$. By Lemma 2, $\mathcal{I}_\mathcal{S}, w_{\mathcal{K}_i} \models \mathcal{K}_i$. Hence, we have $\mathcal{I}_\mathcal{S}, w_{\mathcal{K}_i} \models L$. Therefore, $\mathcal{K}_i \vdash L$ by Definition 22. ▮

Lemma 3 *Let $\mathcal{S} = \{\mathcal{K}_1, \dots, \mathcal{K}_n\}$ be a finite set of knowledge bases and $\mathcal{I}$ be a rigid KB Σ^+-interpretation for $\mathcal{S}$. If $\mathcal{I}, w_{\mathcal{K}_1} \models \mathcal{K}_1^m, \dots, \mathcal{I}, w_{\mathcal{K}_n} \models \mathcal{K}_n^m$, then $\mathcal{I}, w_{\mathcal{K}_1} \models \mathcal{K}_1^{m+1}, \dots, \mathcal{I}, w_{\mathcal{K}_n} \models \mathcal{K}_n^{m+1}$.*

Proof. Suppose $\mathcal{I}, w_{\mathcal{K}_1} \models \mathcal{K}_1^m, \dots, \mathcal{I}, w_{\mathcal{K}_n} \models \mathcal{K}_n^m$. Let $x \in \{1, \dots, n\}$. By Definition 18, $\mathcal{K}_x^{m+1} = \mathcal{K}_x^m \cup \Delta(Th(\mathcal{K}_1^m) \cup \dots \cup Th(\mathcal{K}_n^m))$. By Theorem 1 and $Th(\mathcal{K}_x^m) = \{L \mid \mathcal{K}_x^m \vdash L\}$, $\mathcal{I}, w_{\mathcal{K}_1} \models Th(\mathcal{K}_1^m), \dots, \mathcal{I}, w_{\mathcal{K}_n} \models Th(\mathcal{K}_n^m)$. Let $p_\tau(t) \in \Delta(Th(\mathcal{K}_1^m) \cup \dots \cup Th(\mathcal{K}_n^m))$. Then, there exists $\mathcal{K}_l^m$ such that $p_\tau(t) \in Th(\mathcal{K}_l^m)$. By assumption and the soundness of the Horn-clause calculus, $\mathcal{I}, w_{\mathcal{K}_l} \models p_\tau(t)$. So, $[\![t]\!]_{w_{\mathcal{K}_l}, \alpha} \in I_{w_{\mathcal{K}_l}}(\tau)$ because $\mathcal{I}$ is a Σ^+-interpretation. Then, by Definition 10 (saying that for all $w_i, w_j \in W$, $I_{w_i}(\tau) = I_{w_j}(\tau)$, $I_{w_i}(c) = I_{w_j}(c)$ and $I_{w_i}(f) = I_{w_j}(f)$), $[\![t]\!]_{w_{\mathcal{K}_x}, \alpha} \in I_{w_{\mathcal{K}_x}}(\tau)$. Hence, $\mathcal{I}, w_{\mathcal{K}_x} \models p_\tau(t)$. Therefore, we have $\mathcal{I}, w_{\mathcal{K}_x} \models \mathcal{K}_x^{m+1}$. ▮

Theorem 3 (Soundness of Rigid Property Derivation) *Let $\mathcal{S} = \{\mathcal{K}_1, \dots, \mathcal{K}_n\}$ be a finite set of knowledge bases and L be a ground sorted or ill-sorted atom. If $\mathcal{K}_i \vdash_\mathcal{S} L$, then $\mathcal{K}_i \models_\mathcal{S}^+ L$.*

Proof. Suppose $\mathcal{K}_i \vdash_\mathcal{S} L$. By Definition 19, there exists an expanded knowledge base $\mathcal{K}_i^m$ of $\mathcal{K}_i$ such that $\mathcal{K}_i^m \vdash L$. So we have $\mathcal{K}_i^m \models^+ L$ (by Theorem 1). Let $\mathcal{I}$ be a rigid KB Σ^+-interpretation for $\mathcal{S}$. Assume $\mathcal{I}, w_{\mathcal{K}_1} \models \mathcal{K}_1 (= \mathcal{K}_1^0), \dots, \mathcal{I}, w_{\mathcal{K}_n} \models \mathcal{K}_n (= \mathcal{K}_n^0)$. By Lemma 3, $\mathcal{I}, w_{\mathcal{K}_1} \models \mathcal{K}_1^m, \dots, \mathcal{I}, w_{\mathcal{K}_n} \models \mathcal{K}_n^m$. Hence, $\mathcal{I}, w_{\mathcal{K}_i} \models L$. Therefore, we obtain the conclusion. ▮

We now define the set $rigid\text{-}ground^+(L \leftarrow G) = ground(L \leftarrow G) \cup \bigcup_{\theta^u} ground((L \leftarrow G)\theta^u)$ with $\theta^u = \{x_1 \colon s_1/t_1, \dots, x_n \colon s_n/t_n\}$ where $\mathcal{K}_i \vdash_\mathcal{S} p_{s_k}(t_k)$ for $1 \leq k \leq n$ and $\{x_1 \colon s_1, \dots, x_n \colon s_n\} \subseteq EVar(L \leftarrow G)$.

Lemma 4 *Let $\mathcal{S} = \{\mathcal{K}_1, \dots, \mathcal{K}_n\}$ be a finite set of knowledge bases. Let $\mathcal{I}_\mathcal{S}$ be a rigid canonical interpretation for $\mathcal{S}$ and $L \leftarrow G$ be a clause. Then, the following statements hold:*

1. *$\mathcal{I}_\mathcal{S}, w_{\mathcal{K}_i} \models L \leftarrow G$ if and only if $\mathcal{I}_\mathcal{S}, w_{\mathcal{K}_i} \models rigid\text{-}ground^+(L \leftarrow G)$.*
2. *$\mathcal{I}_\mathcal{S}$ is a rigid KB Σ^+-interpretation of $\mathcal{K}_i$.*

Proof. 1. By Definition 21 and the definition of $rigid\text{-}ground^+(L \leftarrow G)$, this can be shown in the similar way to the proof of Lemma 2.

2. First of all, we have to show that the derivability $\vdash$ in the Horn-clause calculus can be applied to the rigid property derivability $\vdash_\mathcal{S}$. Namely, if $\mathcal{K}_i \vdash A_1, \dots, \mathcal{K}_i \vdash A_n$ derives $\mathcal{K}_i \vdash B$ in an inference rule, then $\mathcal{K}_i \vdash_\mathcal{S} A_1, \dots, \mathcal{K}_i \vdash_\mathcal{S} A_n$ derives $\mathcal{K}_i \vdash_\mathcal{S} B$. Suppose $\mathcal{K}_i \vdash_\mathcal{S} A_1, \dots, \mathcal{K}_i \vdash_\mathcal{S} A_n$. For each A_l, there exists an expanded knowledge base $\mathcal{K}_i^{m_l}$ of $\mathcal{K}_i$ such that $\mathcal{K}_i^{m_l} \vdash A_l$ (by Definition 19). Thus, if $m_k \geq m_j$ for $1 \leq j \leq n$, then $\mathcal{K}_i^{m_k} \vdash A_1, \dots, \mathcal{K}_i^{m_k} \vdash A_n$. Hence, $\mathcal{K}_i^{m_k} \vdash B$. Therefore, $\mathcal{K}_i \vdash_\mathcal{S} B$.

By the above claim and the same way of the proof of Lemma 2, we can show that $\mathcal{I}_\mathcal{S}$ is a Σ^+-interpretation and satisfies $\mathcal{K}_i$. Furthermore, we have to prove that $\mathcal{I}_\mathcal{S}$ is a *rigid* Σ-interpretation, i.e., (1) $I_{w_i}(\tau) = I_{w_j}(\tau)$, (2) $I_{w_i}(c) = I_{w_j}(c)$ and (3) $I_{w_i}(f) = I_{w_j}(f)$. (1) let $t \in I_{w_i}(\tau)$. By the definition of $\mathcal{RT}_{0,\tau}^{\mathcal{K}_i}$, $\mathcal{K}_i \vdash_\mathcal{S} p_\tau(t)$. So, by Definition 19, $\mathcal{K}_i^m \vdash p_\tau(t)$ where $\mathcal{K}_i^m$ is an expanded knowledge base of $\mathcal{K}_i$. This derives $p_\tau(t) \in Th(\mathcal{K}_i^m)$. By Definition 18, $p_\tau(t) \in \mathcal{K}_j^{m+1}$. Then, $\mathcal{K}_j^{m+1} \vdash p_\tau(t)$. Therefore, $t \in I_{w_j}(\tau) (= \mathcal{RT}_{0,\tau}^{\mathcal{K}_j})$. (2) let $c \colon \ \rightarrow \tau \in \Omega$. Then, $I_{w_i}(c) = c \colon \tau = I_{w_j}(c)$ (by Definition 21). (3) let $f \colon \tau_1 \times \dots \times \tau_n \rightarrow \tau \in \Omega$. Because $I_{w_i}(\tau) = I_{w_j}(\tau)$ for any $\tau \in T$, we have $I_{w_{\mathcal{K}_i}}(f)(t_1, \dots, t_n) = f(t_1, \dots, t_n) \colon \tau = I_{w_{\mathcal{K}_j}}(f)(t_1, \dots, t_n)$ where $t_k \in I_{w_{\mathcal{K}_i}}(\tau_k)$ for $1 \leq k \leq n$ (by Definition 21). ▮

Theorem 4 (Completeness of Rigid Property Derivation) *Let $\mathcal{S} = \{\mathcal{K}_1, \dots, \mathcal{K}_n\}$ be a finite set of knowledge bases and L be a ground sorted or ill-sorted atom. If $\mathcal{K}_i \models_\mathcal{S}^+ L$, then $\mathcal{K}_i \vdash_\mathcal{S} L$.*

Proof. Suppose that we have $\mathcal{K}_i \models_\mathcal{S}^+ L$. By Lemma 4, $\mathcal{I}_\mathcal{S}, w_{\mathcal{K}_1} \models \mathcal{K}_1, \dots, \mathcal{I}_\mathcal{S}, w_{\mathcal{K}_n} \models \mathcal{K}_n$. Hence, by assumption, $\mathcal{I}_\mathcal{S}, w_{\mathcal{K}_i} \models L$. Therefore, $\mathcal{K}_i \vdash_\mathcal{S} L$ (by Definition 22). ▮

Conclusion

We have extended an order-sorted logic and its Horn-clause calculus by capturing the notion of the ontological property classification. In particular, our logic contains three kinds of expressions of properties (types, non-rigid sorts and unary predicates) and adheres to the rigidity of the different properties within instantiation and subsumption of properties (excluding $\tau \leq \sigma$). Using these facilities, we have developed a reasoning algorithm for many separated knowledge bases. This knowledge base reasoning is provided by the sorted Horn-clause calculus with extracting rigid property information from other knowledge bases. The suitability of this reasoning is guaranteed by the fact that instantiation of rigid properties and subsumption between sorts (as subsort relations) eternally hold as common knowledge, i.e., the truth is independent of the situation of each knowledge base.

Our future work concerns an ontological consideration of the relationships *part_of* (Winston, Chaffin, & Hermann 1987; Simons 1987) and *member_of* which can be embedded in various extensions of sorted logical systems (Kaneiwa 2004b; Kaneiwa & Tojo 2001). Although these have been often used to build ontologies, the meanings are ambiguously defined for some ontologies. For example, there may be two definitions for the concepts *professor* and *laboratory* such that (i) professors are parts of laboratories and (ii) Prof. Smith is a member of the laboratory (but not an instance of the laboratory). Combining these relationships with this work can be expected to clearly formalize enriched terminological hierarchies for ontology development.

References

Aït-Kaci, H., and Nasr, R. 1986. LOGIN: A logic programming language with built-in inheritance. *Journal of Logic Programming* 3(3):185–215.

Beierle, C.; Hedtsück, U.; Pletat, U.; Schmitt, P.; and Siekmann, J. 1992. An order-sorted logic for knowledge representation systems. *Artificial Intelligence* 55:149–191.

Bürckert, H.-J. 1994. A resolution principle for constraint logics. *Artificial Intelligence* 66:235–271.

Carrara, M., and Giaretta, P. 2001. Identity criteria and sortal concepts. In *Proceedings of the international conference on Formal Ontology in Information Systems*, 234–243. ACM Press.

Cohn, A. G. 1987. A more expressive formulation of many sorted logic. *Journal of Automated Reasoning* 3:113–200.

Cohn, A. G. 1989. Taxonomic reasoning with many sorted logics. *Artificial Intelligence Review* 3:89–128.

Doets, K. 1994. *From Logic to Logic Programming*. The MIT Press.

Frisch, A. M. 1989. A general framework for sorted deduction: Fundamental results on hybrid reasoning. In *In Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*.

Gangemi, A.; Guarino, N.; and Oltramari, A. 2001. Conceptual analysis of lexical taxonomies: the case of wordnet top-level. In *Proceedings of the international conference on Formal Ontology in Information Systems*, 285–296. ACM Press.

Guarino, N., and Welty, C. 2000a. A formal ontology of properties. In Dieng, R., and Corby, O., eds., *Proceedings of EKAW-2000: The 12th International Conference on Knowledge Engineering and Knowledge Management*, 97–112.

Guarino, N., and Welty, C. 2000b. Ontological analysis of taxonomic relationships. In Laender, A., and Storey, V., eds., *Proceedings of ER-2000: The Conference on Conceptual Modeling.* Springer-Verlag LNCS.

Guarino, N.; Carrara, M.; and Giaretta, P. 1994. An ontology of meta-level categories. In P. Torasso, J. Doyle, E. S., ed., *Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasoning*, 270–280. Morgan Kaufmann.

Hanus, M. 1992. Logic programming with type specifications. In Pfenning, F., ed., *Types in Logic Programming*. The MIT Press.

Kaneiwa, K., and Tojo, S. 1999. Event, property, and hierarchy in order-sorted logic. In *Proceedings of the 1999 Int. Conf. on Logic Programming*, 94–108. The MIT Press.

Kaneiwa, K., and Tojo, S. 2001. An order-sorted resolution with implicitly negative sorts. In *Proceedings of the 2001 Int. Conf. on Logic Programming*, 300–314. Springer-Verlag. LNCS 2237.

Kaneiwa, K. 2004a. The completeness of logic programming with sort predicates. *Systems and Computers in Japan* 35(1):37–46.

Kaneiwa, K. 2004b. Resolution for label-based formulas in hierarchical representation. *New Generation Computing* 23(1).

Kaplan, A. N. 2001. Towards a consistent logical framework for ontological analysis. In *Proceedings of the international conference on Formal Ontology in Information Systems*, 244–255. ACM Press.

Kifer, M.; Lausen, G.; and Wu, J. 1995. Logical foundations of object-oriented and frame-based languages. *J. ACM* 42(4):741–843.

Lloyd, J. W. 1987. *Foundations of Logic Programming*. Springer-Verlag.

Lowe, E. J. 1989. *Kinds of Being. A Study of Individuation, Identity and the Logic of Sortal Terms.* Basil Blackwell, Oxford.

Manzano, M. 1993. Introduction to many-sorted logic. In *Many-sorted Logic and its Applications*. John Wiley and Sons. 3–86.

Oberschelp, A. 1962. Untersuchungen zur mehrsortigen quantorelogik. *Mathematische Annalen 145* 297–333.

Oberschelp, A. 1989. Order sorted predicate logic. In *Workshop on Sorts and Types in Artificial Intelligence*.

Schmidt-Schauss, M. 1989. *Computational Aspects of an Order-Sorted Logic with Term Declarations*. Springer-Verlag.

Simons, P. 1987. *Parts: A Study in Ontology*. Clarendon Press, Oxford.

Smolka, G. 1989. *Logic Programming over Polymorphically Order-Sorted Types.* Ph.D. Dissertation, Universitat Kaiserslautern.

Socher-Ambrosius, R., and Johann, P. 1996. *Deduction Systems*. Springer-Verlag.

Strawson, P. F. 1959. *Individuals: An Essay in Descriptive Metaphysics*. London: Methuen.

Walther, C. 1985. A mechanical solution of Schuber's steamroller by many-sorted resolution. *Artificial Intelligence* 26(2):217–224.

Walther, C. 1987. *A Many-Sorted Calculus Based on Resolution and Paramodulation*. Pitman and Kaufman Publishers.

Walther, C. 1988. Many-sorted unification. *Journal of the Association for Computing Machinery* 35:1.

Weibel, T. 1997. An order-sorted resolution in theory and practice. *Theoretical Computer Science* 185(2):393–410.

Welty, C., and Guarino, N. 2001. Supporting ontological analysis of taxonomic relationships. *Data and Knowledge Engineering* 39(1):51–74.

Winston, M. E.; Chaffin, R.; and Hermann, D. 1987. A taxonomy of part-whole relations. *Cognitive Science* 11(4):417–444.