1 Introduction

The importance of contexts in classifying and retrieving objects is often addressed in many literatures in the research fields of information retrieval and conceptual classification. A weighting scheme for attributes of objects or classes, which reflects a user's demand as a kind of context, is a standard approach to this issue and is widely used ([5], for instance). However, in this paper, we investigate a possibility of realizing context-dependent classification and retrieval from a standpoint of goal-dependent abstraction [3, 4] under a domain theory written in order-sorted Horn logic [10, 11].

As both the classification and retrieval depend on the similarities between objects we observe, we have to capture a notion of similarities between them depending on a given context. In a word, we suppose a goal and a present situation, simply called a goal and an initial situation, respectively. The initial situation is supposed to satisfy the goal.

Such a situation is represented by concept graphs [8], that are similar to Ψ-terms [6] in logic programming, and can be regarded as a restricted form of concept definitions in description logic [12, 9, 7]. More precisely, a concept graph consists of typed variables connected by relational predicates, and just corresponds to a body part of Horn rule in logic programming. It has just one external variable called a root. All the other internal variables are existentially quantified. One of the reasons why we consider concept graphs is to present a framework to handle a structural similarity between complex structured objects such as Ψ-terms or concept definitions.

As goals should be satisfied by initial situations, a concept graph representing a goal should be more general than a concept graph representing an initial situation. Such an ordering over a class of concept graphs is formally defined as a concept graph subsumption under a domain theory, and is actually a proof or an explanation of goal under the assumption of situational facts in concept graphs

Then our classification task is reduced to determining whether a concept graph in inquiry is similar to the initial concept graph under the presence of goal. The notion of similarities we consider in this paper is based on abstraction of first order theory [1]. In a word, two concept graphs are analogous whenever their explanations of a goal are both instances of some abstract explanation of a goal. Hence, there may exist more than two analogous concept graphs, given a goal and an initial concept graph. From this nature of abstraction, a concept graph in inquiry is judged similar to the initial situation from the viewpoint of goal, if it is subsumed by some analogue of initial concept graph.

One may understand that the system performs an operation like explanation-based generalization [15]. This is in some sense true, because the present system is completely deductive under a domain theory except the following points:

EBG is used to deductively learn a universally quantified macro rule from a domain theory. For this purpose, some constant symbols are replaced with variables. On the other hand, according to our abstraction approach, originally introduced by the work [1], rule corresponding to universal macro rule is restricted to some disjunction of primitive types, called sorts. In fact, the disjunction of sorts defines a subset of universe of discourse. It is not the whole interpretation domain. Thus a family of predicate symbols is replaced with more abstract predicate whose extension is wider than any predicate in that family.

Secondly, the system performs generalization to obtain more general

abstractable situation, only when some adequate similarity cannot be found. As an initial situation is more specific, the similarities with the situation may be fewer. As a result, less number of situations become non-similar to the initial situation. In such a case, we apply the generalization operator to obtain a more general situation from which more number of actual situations becomes similar.

This paper is organized as follows. In Section 2, we describe our underlying language, an order-sorted Horn logic, according to [10, 11]. In Section 3, we present a notion of goal-dependent abstraction (GDA) for order-sorted theory, which is a refined version of GDA stated in [3, 4]. A notion of concept graphs and their subsumptions is introduced in Section 4. Based on these preparation, we present in Section 5 our classification problem and a method to classify conceptual situations under a domain theory. In Section 6, we show how our system behaves, using two examples. Some experimental results are also presented. In Section 7, we conclude this paper and discuss some future works.

2 Preliminaries

Our domain theory is described in terms of *order-sorted logic*, so we first describe it briefly. Secondly, we introduce a notion of *sorted abstraction* for grouping several primitive concepts appeared in the domain theory and for producing an abstract level theory.

2.1 Domain Theory

Our *domain theory* $\mathcal{KB}$ consists of the following two components, a *sort hierarchy* $\mathcal{H}$ and a *rule set* $\mathcal{T}$, and is denoted by $\langle \mathcal{H}, \mathcal{T} \rangle$.

Sort Hierarchy:

A *sort hierarchy* is given as a partially ordered set of sort symbols, $\mathcal{H} = (S, \preceq)$. Each sort symbol $s \in S$ denotes a concept and is interpreted as a non-emtpy set A_s of individual objects belonging to the concept, given an interpretation* . An ordering $s \preceq s'$ means that s is a subconcept of s', and is understood as a subset relationship $A_s \subseteq A_{s'}$ under the interpretation. We also say that a concept *subsumes* its subconcepts or that the subconcepts are subsumed by their superconcept. Furthermore, we use no constant symbol denoting an individual, although Skolem constants are introduced for technical reasons. Instead of constant symbols, some minimal sorts $s \in S$ may be understood as *individuals*. Formally, a sort s is said minimal if no $s' \in S$ except s satisfies $s' \preceq s$.

Thus, compared with standard formalism of sort hierarchy, we do not suppose the least sort $\bot$ denoting contradiction or empty set. When we do like to put $\bot$ in S, then the minimal sorts corresponding to individuals should be direct successors of $\bot$.

* We suppose in this paper that a sort denotes a non-emtpy set of individuals. An alternative assumption[10] is that each sort has at least one ground term of it. These assumptions are necessary to guarantee the equivalence of order-sorted logic and the corresponding standard first order logic, as is shown in Example 1.

Rule Set:

A *rule set* $\mathcal{T}$ is a set of function-free typed Horn clauses of the form

$$\forall X_1 : s_1 \cdots \forall X_n : s_n \quad A :\!- B_1, \cdots, B_k,$$

where A and B_i are positive literals and $X_j : s_j$ is a typed variable of sort $s_j \in S$, and $X_1, \cdots, X_n$ are all the variables appearing in it. X_j is supposed to take an instance of s_j as its possible value. When we omit the universal quantifiers for notational convenience, the first occurrence of X_j appears as its typed form $X_j : s_j$. Other occurrences are simply written in its non-typed form, as it is clear from the syntax. For instance,

```
p(X:s1):- q(X,Y:s2),r(X,Y)
```

is an abbreviation of

```
∀X:s1∀Y:s2 p(X):- q(X,Y),r(X,Y).
```

Order-Sorted Inference:

The inference under $\mathcal{KB} = \langle \mathcal{H}, \mathcal{T} \rangle$ is the same as standard first-order deduction except the point that all the substitutions used in our inference are assumed to be *order-sorted*.

More precisely, every substitution $\theta = \{X_i : s_i / Y_i : u_i\}$, called an *order-sorted substitution*, should satisfy the type constraint $u_i \preceq s_i$. That is, the variable X_i of s_i must be instantiated to a variable Y_i of more specific sort u_i. By the definition, two terms $X_i : s_i$ and $Y_i : u_i$ are unifiable only if one of the following three conditions hold:

1. $s_i \preceq u_i$,
2. $u_i \preceq s_i$, or
3. s_i and u_i has a common subconcept v_i such that $v_i \preceq s_i$ and $v_i \preceq u_i$.

In any case, the variables are instantiated to their more specific sorts. Particularly in the last case, we introduce a new variable Z_i of common subconcept v_i, and then make a unifier $\{X_i : s_i / Z_i : v_i, Y_i : u_i, / Z_i : v_i\}$.

When s_i and u_i have no common lower bound, the unification fails. Even when they have some common lower bound, if there does not exist the greatest lower bound in $\mathcal{H}$, then we should consider a set of "maximal unifiers", each of which is tried in some order when we execute our inference. First order deduction equipped with the order-sorted unification is written as $\mathcal{T} \vdash_{\mathcal{H}} F$. In what follows, the term "order-sorted" may be dropped, for we assume it throughout this paper.

Translation to first order inference:

Inference under the domain theory $\mathcal{KB} = \langle \mathcal{H}, \mathcal{T} \rangle$ can easily be translated to a standard first order logic. We here show the translation axioms for our order-sorted language with no constant symbols, according to the literature [10]. Particularly, an axiom of non-emptyness of sorts is added to those in [10].

Sortal predicates: For each sort $s \in S$, we introduce a unary predicate symbol p_s called a *sortal predicate* of s. p_s is also simply written as s without any confusion.

Non-emptyness of sort: For each minimal sort $s \in S$, we assert an axiom $\exists X p_s(X)$ meaning that each sort denotes a non-empty set of individuals. Actually, it is equivalent to introduce a Skolem constant a_s of sort s by an axiom $p_s(a_s)$.

Sort subsumption rules: For each $s_1 \preceq s_2$ in $\mathcal{H}$, we assert a sort subsumption rule $\forall X\ p_{s_2}(X) : -p_{s_1}(X)$.

Rules in general: For each Horn rule

$$C = \forall X_1 : s_1 \cdots \forall X_n : s_n \quad A : -Body$$

in $\mathcal{T}$, we assert a rule

$$C^* = \forall X_1 \cdots \forall X_n \quad A : -p_{s_1}(X_1), ..., p_{s_n}(X_n), Body$$

using the sortal predicates.

Target formula to be proved: If the formula F has the form $\forall X : s\ A(X)$, then translate it to $F^* = \forall X\ A(X) : -s(X)$. If the formula F has the form $\exists X : s\ A(X)$, then translate it to $F^* = \exists X\ s(X) \wedge A(X)$. When $A(X)$ still has typed variables, then repeat the same translation to those variables.

Then we have a set $\mathcal{T}^*$ of first order Horn rules as in the above such that

$$\mathcal{T} \vdash_{\mathcal{H}} F \quad \text{if and only if} \quad \mathcal{T}^* \vdash F^*$$

Figure 1 illustrates all the constructs introduced in this subsection.

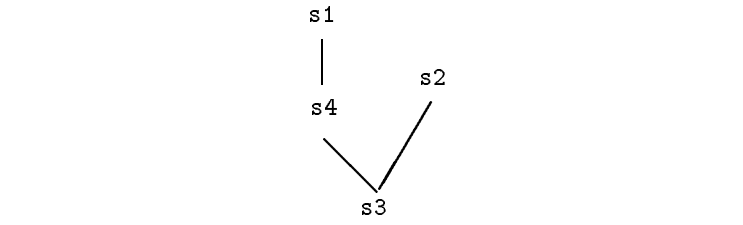
3 Sorted Abstraction

We introduce here a notion of *sorted abstraction*. A non-sorted version of abstraction has been originally proposed by J.D.Tenenberg [1] for constructing a grouping of relational predicates. The sorted abstraction, on the other hand, restricts predicates to be grouped to sortal predicates in the translated theory $\mathcal{T}^*$. Thus our abstraction provides a method for grouping sorts in $\mathcal{H}$.

Definition 1 (Grouping of Sorts)
Let $\mathcal{H} = (S, \preceq)$ be a sort hierarchy. A grouping ψ of sort set S is simply a partition $\{S_1, \cdots, S_k\}$ of S. That is, $S_i \cap S_j = \phi\ (i \neq j)$ and $S = \cup_{j=1}^{k} S_j$. Regarding each S_j as an abstract sort symbol α_{S_j}, we represent ψ as a mapping from S to an abstract sort set $\tilde{S} = \{\alpha_{S_j} | 1 \leq j \leq k\}$ such that $\psi(s) = \alpha_{S_j}$ if $s \in S_j$. We do not distinguish sort mappings and groupings of sorts. ■

A grouping $\psi = \{S_1, \cdots, S_k\}$ can be easily extended to a mapping over a set of any expressions. For an expression E, $\psi(E)$ is defined as an expression obtained by rewriting any sort s in E to its abstract sort $\psi(s) = \alpha_{S_j}$, where $s \in S_j$. $\psi(E)$ is simply called a *ψ-image* of E. Similarly, ψ-image of a sort subsumption $s_1 \preceq s_2$ is defined as $\psi(s_1) \preceq \psi(s_2)$.

As a grouping ψ classifies sorts in a sort set into several groups S_j, we can consider that ψ represents a hypothesis that s_1 and s_2 are similar



Sort Hierarchy

```
:-p(X:s1)
    | p(X1:s2):-q(X1).
    | θ1 ={X:s1/Y:s3, X1:s2/Y:s3}
:-p(Y:s3)
    | q(X2:s4).
    | θ2={X2:s4/Y:s3}
    □
```

Refutation of ¬∃X:s1 p(X)

```
p(X:s2):-q(X).
q(X:s4).
```

Domain Theory

```
s1(X):-s4(X),  s4(X):-s3(X),  s2(X):-s3(X).,
p(X1):-s2(X),q(X1).   q(X2):-s4(X),
∃X s3(X) (or equivalently s3(a3))
```

Translated Theory

```
:-s1(X),p(X)
```

Translated formula to be proved

Figure 1: An Example of Order-Sorted Inference and its Translation to First-Order Logic

whenever they belong to the same group S_j. In this sense, we say that s_1 is an *analogue* of s_2 or *similar* to s_2 w.r.t. ψ. Under this interpretation, the ψ-image $\psi(E)$ can be regarded as an expression at abstract level obtained by neglecting the differences between similar sorts in the same groups. More precisely speaking, the act of neglecting the differences can be captured by defining the meaning of abstract sort symbols.

Definition 2 (Meaning of Abstract Sorts)
For a grouping $\psi = \{S_1, \cdots, S_k\}$ of sort set S, an abstract sort α_{S_j} is defined by the disjunction $\alpha_{S_j} = \bigvee_{s \in S_j} s$. Particularly, when we consider the translated non-sorted first order theory $\mathcal{T}^*$, the introduction of disjunction is equivalent to asserting a formula $p_{\alpha_{S_j}}(X) \leftrightarrow \bigvee_{s \in S_j} p_s(X)$ to $\mathcal{T}^*$. $\mathcal{T}^*$ equipped with these disjunction rules is denoted as $\mathcal{T}^*(\psi)$. ■

For instance, suppose we have a rule $E =$ (`p(X:s1):-q(X)`) in $\mathcal{T}$ and a sort `s1`$\in S_j$ in a grouping ψ. Then we have its ψ-image (`p(X:`α_{S_j}`):-q(X)`) asserting that, for any $u \in S_j$, `p(X:`u`):-q(X)` holds, as α_{S_j} is defined by the disjunction of $u \in S_j$. `p(X:`u`):-q(X)` is called an *instance* of $\psi(E)$ or an ψ-*analogue* of E. If every ψ-analogue is provable from $\mathcal{T}$ with $\mathcal{H}$, then the ψ-image $\psi(E)$ is logically valid, and we can therefore actually identify sorts in the same group of ψ in the rule E. In such a case, the original rule E is said to be *abstractable w.r.t.* ψ. Otherwise, we cannot neglect the differences between sorts even in the same group, as the neglection produces a ψ-image which may be false.

Thus, for a given grouping ψ, rules in $\mathcal{T}$ and sort subsumptions in $\mathcal{H}$ can be divided into two kinds: one is abstractable w.r.t. ψ and the other is not. Suppose for instance the following domain theory consisting of three unit clauses and a sort hierarchy $\mathcal{H}$:

$\mathcal{T} =$ { `(p(X:s1).), (p(X:s2).),  (q(X:s1).)` }.
$\mathcal{H} =$ {`s1`$\preceq$`s0`, `s2`$\preceq$`s0`}

For a grouping {{`s0`},{`s1,s2`}}, the first two facts are both abstractable to `p(X:`$\alpha_{S_{12}}$`)` , where S_{12}={`s1,s2`}. On the other hand, `q(X:s1)` is not abstractable, since its analogue `q(X:s2)` is not derived from $\mathcal{T}$ with $\mathcal{H}$. Similarly, both the sort subsumptions `s1`$\preceq$`s0` and `s2`$\preceq$`s0` are abstractable to $\alpha_{S_{12}} \preceq \alpha_{S_0}$, where $S_0 =$ {`s0`}.

Now we are ready to present the definition of ψ-abstractableness. For both rules and sort subsumptions, it requires that any ψ-analogue, which is ψ-instance of ψ-image, is provable in the original domain theory.

Definition 3 (Abstractable Rule and Subsumption)
Let $\langle \mathcal{H} = (S, \preceq), \mathcal{T} \rangle$ be a domain theory and $\psi = \{S_1, \cdots, S_n\}$ be a grouping of S.

1. A clause $C = (A :- B_1, \cdots, B_k)$ in $\mathcal{T}$ is said to be *abstractable w.r.t.* ψ to the corresponding ψ-image $\psi(C)$, if $\mathcal{T} \vdash_{\mathcal{H}} D$ holds for any $D \in \psi^{-1}(\psi(C))$, a ψ-analogue of C, where $\psi^{-1}(E_a) = \{E_c | \psi(E_c) = E_a\}$ (that is, the set of ψ-instances of E_a).

2. A sort subsumption $s_1 \preceq s_2$ in $\mathcal{H}$ is said to be *abstractable w.r.t.* ψ to its ψ-image $\psi(s_1) \preceq \psi(s_2)$, if, for any ψ-analogue u_1 of s_1, there exists a ψ-analogue u_2 of s_2 such that $u_1 \preceq u_2$ holds in $\mathcal{H}$.

■

The proposition below is a direct consequence of the definition.

Proposition 1

1. *A rule E in $\mathcal{T}$ is abstractable w.r.t. ψ if and only if $\mathcal{T}^*(\psi) \vdash (\psi(E))^*$.*
2. *A sort subsumption $s_1 \preceq s_2$ is abstractable w.r.t. ψ if and only if $\mathcal{T}^*(\psi) \vdash \forall X(p_{\psi(s_2)}(X) :- p_{\psi(s_1)}(X))$.*

A notion of *Goal-Dependent Abstraction*, simply denoted as GDA, has been defined based on a simple intuition that

> Only an abstraction that keeps every necessary information to perform an inference under some context should be taken into account. Then, the power of proving what we concern will not be decreased.

The latter property is called *Upward Solution Property* (USP for short), and has been originally introduced in the pioneering work by Plaisted [2], although the notion of context is not involved in it. In the framework of GDA, on the other hand, the context is defined by means of a goal formula to be proved. Every rules used to prove the goal are relevant, and any other rules are not. Then, USP is satisfied, for every relevant rule is preserved even after the abstraction.

A similar property, the *safety condition* of abstraction, is also used in the study of abstract interpretation of programs. However, it allows an inference at abstract level to which no inference at concrete level corresponds. In contrast to USP, the property that any abstract inference corresponds to one at concrete level is called *Downward Solution Property* (DSP for short). GDA satisfies DSP as well as USP for a given goal to be proved.

Definition 4 (Goal Dependent Abstraction)
Given a goal $G = (:- A_1, ..., A_n)$, a grouping ψ is said to be a *Goal-Dependent Abstraction*(GDA) for G, if every rule in $\mathcal{T}$ and sort subsumption $s_1 \preceq s_2$ used to prove G is abstractable w.r.t. ψ, where a rule $R \in \mathcal{T}$ is said to be used to prove G, if it appears as a side clause of the SLD-refutation of G, and a sort subsumption $s_1 \preceq s_2$ is said to be used to prove G, if $X_2 : s_2 / X_1 : s_1$ is contained in some unifier in the SLD-refutation of G. ■

The following proposition about USP is easily proved from Proposition 1.

Proposition 2 *Suppose a grouping ψ is a GDA for a goal $G = (:- A_1, ..., A_n)$. If G has a SLD-refutation, then a goal $\psi(G) = (:- \psi(A_1), ..., \psi(A_n))$ also has a SLD-refutation under $\langle \psi(Rules(G)), \psi(Subsumptions(G)) \rangle$, where $Rule(G)$ and $Subsumptions(G)$ are the sets of used rules and subsumptions, respectively.*

4 Concept Graphs and Their Subsumption Relationships

In order to describe a situation in inquiry, a notion of *concept graph* is introduced in terms of order-sorted Horn logic. A concept graph consists of conceptual nodes labeled by sorts and relational predicates connecting them.

The concept graph can be considered as a variation of Ψ-terms [6] for representing complex record structures in logic programming. In fact, the syntax of Ψ-terms and the concept graphs are the same in case we treat only binary predicates. The differences are found in their semantics even in that case. In the Ψ-terms, links connecting conceptual nodes are regarded as attribute roles that are interpreted as functions, while the corresponding links in the concept graph in this paper are role relations interpreted as n-ary relations over interpretation domain.

Similar notions are also used in the research area on *description logic* to depict their concept definitions in the form of graphs [9, 12]. Although the description logic can deal with various kinds of constraints for role relations, we are concerned with only existentially quantified constraints to perform our inference equipped with the concept graph formalism within the framework of order-sorted Horn logic.

Definition 5 (Concept Graph)
A *concept graph* $G(X : s)$ is simply defined as a conjunction of atomic formulas, where just one variable $X : s$ is designated as a root variable, where s is a sort in our sort hierarchy $\mathcal{H}$. Every variable except $X : s$ is assumed to be existentially quantified. When every predicate in $G(X : s)$ is unary or binary, then we depict it as a finite directed graph (E, R) with a set E of typed variables in $G(X : s)$ and a set R of predicate instances connecting them. ■

For instance, Figure 2 shows a concept graph

CG_1`(X:person)= { has_a_child(X:person, Y:person), likes(Y, Z:dog), bite(Z,Y) }`

As a formula, it represents

$\exists$ `Y:person` $\exists$ `Z:dog has_a_child(X:person, Y:person)`$\wedge$
`likes(Y,Z:dog)`$\wedge$`bite(Z,Y)`

meaning "a person `X` who has a child bitten by a dog he/she likes".

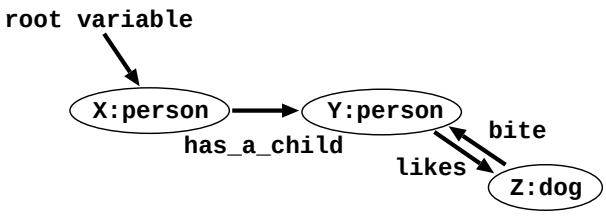


Figure 2: A Concept Graph

The denotation of a concept graph $CG(Y_0 : s_0) = CG(Y_0, Y_1 : s_1, ..., Y_k : s_k)$, given an interpretation $\mathcal{I} = \langle < A_s >_{s \in S}, \{p_{\mathcal{I}}\} \rangle$, is defined as the set of individual objects $b_0 \in A_{s_0}$ such that there exist individual objects $b_j \in A_{s_j}$ satisfying the property that every atom in $CG(Y_0 : s_0)$ becomes true under the variable assignment $\mathcal{I}(Y_j) = b_j$ $(0 \leq j \leq k)$.

Using our domain theory $\mathcal{T}$ as well as sort hierarchy $\mathcal{H}$, we can easily introduce an ordering between concept graphs. For instance, suppose we have a rule in $\mathcal{T}$

```
has_a_child(X:person,Y:male_person):-has_a_son(X,Y).
```

Then,

```
CG2(X:person)= { has_a_son(X:person, Y:male_person),
                 likes(Y,Z:dog), bite(Z,Y) }
```

turns out to be a special case of CG_1. This is simply because

1. if `X:person` has a son `Y:male_person`, it follows from the rule that `X` has `Y` as his child and
2. `male_person` $\preceq$ `person`.

In such a case a more general case CG_1 is said to subsume a more special case CG_2. Formally, we have the following definition.

Definition 6 (Subsumption between Concept Graphs) Let $CG_i(X_i : s_i)$ and $CG_j(X_j : s_j)$ be two concept graphs. $CG_j(X_j : s_j)$ is said to subsume $CG_i(X_i : s_i)$ w.r.t. a knowledge base $\mathcal{B} = \langle \mathcal{H}, \mathcal{T} \rangle$, denoted by $CG_i(X_i : s_i) \preceq_{\mathcal{B}} CG_j(X_j : s_j)$, if (1) $s_i \preceq s_j$ and (2) $\mathcal{T} \vdash_{\mathcal{H}} \forall X : s_i(\exists CG_i(X) \rightarrow \exists CG_j(X))$, where $\exists CG(X)$ stands for $\exists Y_1 ... \exists Y_m CG(X, Y_1, ..., Y_m)$. ■

The following theorem is a refinement of one in [8].

Theorem 1 *$CG_2(Z_2 : u_2) \leq_{\mathcal{T}} CG_1(Z_1 : u_1) \Leftrightarrow \mathcal{I}(CG_2) \subseteq \mathcal{I}(CG_1)$ holds for any model $\mathcal{I}$ of $\mathcal{T}$, where $\mathcal{I}(CG)$ is the denotation of CG, given an interpretation $\mathcal{I}$*

Proof. .

First, from the definition of subsumption, the left hand side directly implies the right hand side. Conversely, suppose an Herbrand interpretation $M(T)$ and let the variables in CG_2 be $Z_2 : u_2 = X_0 : s_0, X_1 : s_1, ..., X_m : s_m$. As we assume non-empty domain for each sort, we can introduce new constants a_j for each $X_j : s_j$ of the same sort, and make a ground atom $p(a_{j_1}, ..., a_{j_k})$ for each atom $p(X_{j_1}, ..., X_{j_k}) \in CG_2$ in addition to clauses in $\mathcal{T}$. This forms a logic program $P = \mathcal{T} \cup \{p(a_{j_1}, ..., a_{j_k}) | p(X_{j_1}, ..., X_{j_k}) \in CG_2\}$. From the definition of denotation of concept graph, we have $a_0 \in M(P)(CG_2)$, where $M(P)(CG_2)$ is the least Herbrand model of P. By the assumption $M(P)(CG_2) \subseteq M(P)(CG_1)$, a_0 is also a member of $M(P)(CG_1)$. As the root variable is typed by u_1, this occurs only when $u_2 \preceq u_1$. This establishes the first requirement of the definition of subsumption. Again by $M(P)(CG_2) \subseteq M(P)(CG_1)$, $\exists CG_1(a_0)$ is true under $M(P)$. This implies that there exists an SLD refutation of goal clause $:- CG_1(a_0)$ from P. This is equivalent to $T \vdash \exists CG_2(a) \rightarrow \exists CG_1(a)$, as is explained in Subsection 4.1. Hence, by the theorem on constant, we have the conclusion.

4.1 Subsumption Check in terms of order-sorted SLD-refutation

Now, from the theorem, we have an actual subsumption checking algorithm. In order to prove a subsumption

$$\mathcal{T} \vdash_{\mathcal{H}} \forall X [\exists Y_1 \cdots \exists Y_m CG_1(X : s, Y_1 : s_1, \ldots, Y_m : s_m) \rightarrow \exists Z_1 \cdots \exists Z_n CG_2(X, Z_1 : u_1, \ldots, Z_n : u_n)],$$

we first take the negation of conclusion:

$$\begin{aligned}
&\neg\forall X[\exists Y_1 \cdots \exists Y_m CG_1(X : s, Y_1 : s_1, \ldots, Y_m : s_m) \\
&\qquad \rightarrow \exists Z_1 \cdots \exists Z_n CG_2(X, Z_1 : u_1, \ldots, Z_n : u_n)] \equiv \\
&\exists X[\exists Y_1 \cdots \exists Y_m CG_1(X : s, Y_1 : s_1, \ldots, Y_m : s_m) \wedge \\
&\qquad \forall Z_1 \cdots \exists Z_n \neg CG_2(X, Z_1 : u_1, \ldots, Z_n : u_n)] \equiv \\
&\exists X \exists Y_1 \cdots \exists Y_m \forall Z_1 \cdots \exists Z_n [CG_1(X : s, Y_1 : s_1, \ldots, Y_m : s_m) \wedge \\
&\qquad \neg CG_2(X, Z_1 : u_1, \ldots, Z_n : u_n)].
\end{aligned}$$

Then by skolemizing the existentially quantified variables X and $Y_i : s_i$ to new constant symbols a and b_i of the corresponding sorts, we have a set of clauses

$$\forall Z_1, \ldots, \forall Z_n [CG_1(a, b_1, \ldots, b_m) \wedge \neg CG_2(a, Z_1 : u_1, \ldots, Z_n : u_n)],$$

and add it to $\mathcal{T}$. Note that an empty clause (denoting a contradiction) would be derived under the resultant rule set with $\mathcal{H}$, if the original subsumption can be proved under $\mathcal{T}$ with $\mathcal{H}$. Then the original problem of proving the subsumption is now reduced to the problem of whether we can refute a goal clause

$$:-CG_2(a, Z_1 : u_1, \ldots, Z_n : u_n)$$

under $\mathcal{T} \wedge CG_1(a, b_1, \ldots, b_m)$ by our order-sorted SLD-refutation. An example is presented in the next subsection.

4.2 Chunking the Proof Block

As we have already mentioned, the inferential power of our order-sorted Horn logic is the same as one of its translation to first-order Horn logic. In the order-sorted logic, taxonomic information as a form of sort subsumptions in $\mathcal{H}$ is separated from a rule set $\mathcal{T}$, and is used to make the rules applicable to a goal clause. As in the definition of GDA, used rules as well as used unifications are required to be abstractable. If we consider this requirement in the framework of non-sorted first order translation, then we are forced to check the abstractableness for each clause corresponding to either a rule in $\mathcal{T}$ or a sort subsumption in $\mathcal{H}$ independently, although the subsumption is always coupled with some rule in $\mathcal{T}$ whenever the rule is used in the proof. In contrast to the first order case, therefore, we consider in this paper that

> the information of how a rule in $\mathcal{T}$ is used to show a given goal clause should be abstractable as a whole. In other words, a used rule and corresponding sort subsumptions must be abstractable at the same time (that is, they are considered as a chunk).

In this subsection, we redefine the proof structure in terms of ground atoms, based on Subsection 4.1 and the above statement. This enables us to treat generalization processes used in the latter sections in a simpler way. First we make a general remark for the case that rules in $\mathcal{T}$ are not range-restricted.

Completely similar to a standard theory of logic programming, [13] for instance, if there exists an order-sorted SLD-refutation of a goal clause $G = (:-A_1, ..., A_n)$, then we have $\mathcal{T} \vdash_{\mathcal{H}} A_1\theta, ..., A_n\theta$, where θ is a computed answer substitution. When we still find free variables $Z_1 : s_1, ..., Z_k : s_k$ in $A_1\theta, ..., A_n\theta$, it follows that

$$\mathcal{T} \vdash_{\mathcal{H}} \forall Z_1 \cdots \forall Z_k (A_1\theta, ..., A_n\theta)$$

Even in such a case, we introduce a new constant symbol for each free variable Z_j of the same sort to denote an indefinite object in the sort. Then, without loss of generality, we can assume in what follows that we have a set

of ground atoms $A_1\theta, ..., A_n\theta$ each of which is a logical consequence of our domain theory $\mathcal{KB}$.

Now, we are ready to define a proof structure in terms of ground atoms as follows.

Definition 7 (Proof Structure)
Given two concept graphs $CG_0(X : s, Y_1 : s_1, ..., Y_k : s_m)$ and $CG_1(X' : s', Z_1 : u_1, ..., Z_k : u_n)$ such that $CG_0 \preceq_{\mathcal{KB}} CG_1$, the proof structure of the subsumption is a directed acyclic graph (V, E) of set of vertices V and set of edges E, and is constructed from an order-sorted SLD-refutation $(R_1, \theta_1), ..., (R_d, \theta_d), \square$ of a goal clause $(:-CG_1(a : s, Z_1 : u_1, ..., Z_k : u_m))$, where R_j is the j-th side clause in

$$\mathcal{T} \cup \{A\tau | A \in CG_0, \tau \text{ is a Skolemization of variables in } CG_0 \text{ with } X\tau = a\},$$

and θ_j is the unifier used to resolve j-th goal clause with R_j. The corresponding computed answer substitution is denoted as $\theta = \theta_1 \cdots \theta_d$, and is supposed as a grounding substitution, as we have just made a remark.

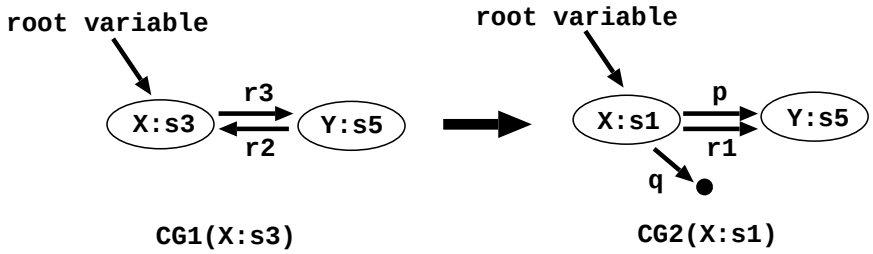
1. V is defined as $\{A\theta | A$ appears as an atom in some $R_j\}$.
2. $(B_{in_body} \longrightarrow B_{head}) \in E$ if there exists $R_j = (A :- As) \in \mathcal{T}$ such that $A\theta = B_{head}$ and $B_{in_body} \in As\theta$. Moreover, the pair $(R_j, subs(R_j))$ of used rule and sort subsumptions is labeled on the edge, and it is denoted as $(B_{in_body} \xrightarrow{(R_j, subs(R_j))} B_{head})$. As $subs(R_j)$ is defined as the set of sort subsumptions $s \preceq s'$ such that a variable in R_j of sort s' is instantiated into a constant of sort s by the answer substitution θ, $(R_j, subs(R_j))$ represents how the rule R_j is used to prove CG_1 under CG_0. If $As = \phi$, it is denoted as $\texttt{true} \xrightarrow{(R_j, subs(R_j))} B_{head}$. For simplicity, in our actual graph representation, such a vertex B_{head} is considered as a leaf node with which $(R_j, subs(R_j))$ is directly associated. As a result, we have two kinds of leaf nodes. One is an instance of unit clause in $\mathcal{T}$ and the other is a Skolemized atom in CG_0.
3. If the (V, E) thus defined contains a cycle, we collect only the shortest paths from vertices in CG_0 or instances of unit clauses in $\mathcal{T}$. Then we consider only a set of vertices on such paths. These paths and vertices form the desired graph.

■

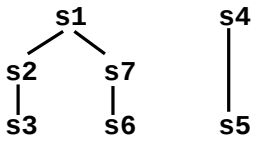
For instance, suppose we have two concept graphs shown in Figure3, where the left one, `CG1(X:s3)`, is subsumed by the right one, `CG2(X:s1)`, under the domain theory presented in the same figure. According to the argument in Subsection 4.1, suppose the root variable `X` and all the intermediate variables in `CG1(X:s3)` are Skolemized. Then the subsumption checking succeeds by order-sorted SLD-refutation as illustrated in Figure 4. The proof of the subsumption consists of instantiated atoms appearing in the clauses `(p(a,b):-q(a),r1(a,b))`, `r1(a,b):-r3(a,b)` and `(q(a)`. Thus, the vertex set V is

{`p(a,b), q(a), r1(a,b), r3(a,b)`}.

Moreover, we have in this case the set E of edges



The dark circle ● following an arrow $X \xrightarrow{q}$ means that q is a unary predicate denoting a property and that the variable X satisfies q.



Sort Hierarchy $\mathcal{H}$

```
C1: p(X:s2,Y:s4):-q(X),r1(X,Y).
C2: q(X:s2).
C3: r1(X:s2,Y:s4):-r3(X,Y).
C4: p(X:s7,Y:s4):-q(X),r1(X,Y).
C5: q(X:s7).
```

Rule Set $\mathcal{T}$

Figure 3: An Example of Concept Graph Subsumption

$$\{\ \mathtt{q(a)} \xrightarrow{(\mathtt{C1},\{\mathtt{s3}\preceq\mathtt{s2},\mathtt{s5}\preceq\mathtt{s4}\})} \mathtt{p(a,b)},\quad \mathtt{r1(a,b)} \xrightarrow{(\mathtt{C1},\{\mathtt{s3}\preceq\mathtt{s2},\mathtt{s5}\preceq\mathtt{s4}\})} \mathtt{p(a,b)},\quad \mathtt{true} \xrightarrow{(\mathtt{C2},\{\mathtt{s3}\preceq\mathtt{s2}\})} \mathtt{q(a)},\quad \mathtt{r3(a,b)} \xrightarrow{(\mathtt{C3},\{\mathtt{s3}\preceq\mathtt{s2},\mathtt{s5}\preceq\mathtt{s4}\})} \mathtt{r1(a,b)}\ \}.$$

For example, the sort subsumptions associated with an edge

$$\mathtt{r3(a,b)} \xrightarrow{(\mathtt{C3},\{\mathtt{s3}\preceq\mathtt{s2},\mathtt{s5}\preceq\mathtt{s4}\})} \mathtt{r1(a,b)}$$

implies that $\mathtt{C3} = (\mathtt{r1(X:s2,Y:s4)} :- \mathtt{r3(X,Y)})$ is instantiated to

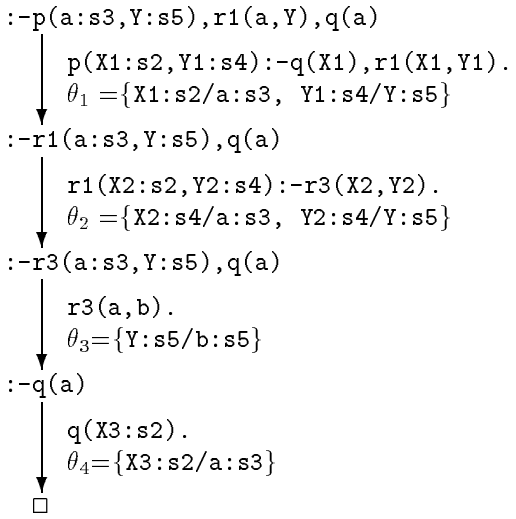
$$\mathtt{r1(a:s3,b:s5)} :- \mathtt{r3(a,b)})$$

by using the sort subsumptions.

For notational convenience, we here introduce some notations used in the next section.

Definition 8 (Notation)
For a label $(R = R(X_1 : s_1, ..., X_n : s_n), subs(R))$ in a proof structure, its ground instance $R' = R(a_1 : t_1, ..., a_n : t_n)$ of R by an substitution $\tau = \{X_1/a_1, ..., X_n/a_n\}$ corresponding to $subs(R)$ is written as $apply(R, subs(R))$. Moreover, for such a grounding rule $R' = R\theta$, a rule $R(Z_1 : t_1, ..., Z_n : t_n)$ obtained by replacing each constant $a_j : t_j$ by a new variable $Z_j : t_j$ of the same sort is denoted by $var_abs(R')$. ■

In the case of the present example,



Remarks on the refutation:

1. As is explained in Subsection 4.1 generally, the subsumption $\texttt{CG1(X:s3)} \preceq \texttt{CG2(X:s1)}$ is verified as in the refutation left. We use new constant symbols from outside of our language, a:s3 and b:s5 with the corresponding sorts s3 and s5, respectively.
2. In the order-sorted substitution, any variable X of type s_X can be instantiated to a constant a of type s_a only when $s_a \preceq s_X$ holds in $\mathcal{H}$.

Figure 4: Proof of $CG1(X) \preceq CG2(X)$ in the form of order-sorted SLD-refutation

$$var_abs(\texttt{r1(a : s3, b : s5)} : -\texttt{r3(a, b)}) = \texttt{r1(Z1 : s3, Z2 : s5)} : -\texttt{r3(Z1, Z2)}.$$

The final proof structure is illustrated in Figure 5 [†] .

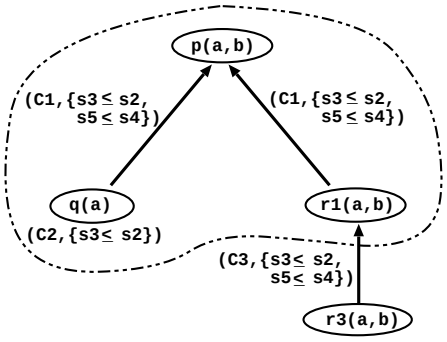


Figure 5: Proof Structure of $\texttt{CG}_1 \preceq_{\mathcal{KB}} \texttt{CG}_2$

5 Abstractable Concept Graph based on Generalization

In this section, we describe our classification problem and a method to classify conceptual situation under a domain theory, based on the subsumption relationships between concept graphs. The basic strategy here for classifying

[†] In the next section, the substructure circled by dotted line is refered as the proof structure of $\texttt{CG_12} \preceq_{\mathcal{KB}} \texttt{CG}_2$, where CG_12 is defined as $\{\texttt{r1(X : s3, Y : s5)}\}$.

things represented by concept graphs is to use a similarity notion between concept graphs.

As an example, suppose we have a person looking for "a dangerous thing in a park just like a car". Normally, the property of dangerousness will be specified by using some attributes or predicates. In addition, it should be similar to car from the viewpoint of dangerousness. As we know various dangerous things because of their own particular reasons, we cannot consider that things are similar to car simply because they share the property of dangerousness. Similar things must be dangerous in the same reason why car is dangerous in a park. In other words, we regard the similarity is situationally valid only when we can explain dangerous aspect of things just as we can do for car in a park. In the definition below, two concept graphs CG_{goal} and CG_{init} are used to specify such a similarity situation. The former is called a goal and is used to specify what aspect or property of things we are concerned with. The latter shows a particular situation from which we can draw an explanation, the proof structure of goal from the situation. Then we try to find a goal-dependent abstraction, a grouping of sorts by which any used rule $(R, subs(R))$ in the proof structure of a goal becomes abstractable.

Definition 9 (GDA in terms of Proof Structure)
A grouping ψ of sorts is said to be a goal-dependent abstraction for a goal CG_{goal} from an initial concept graph $CG_{init} = CG_{init}(X_0 : s_0, ..., X_k : s_k)$, if the following condition holds:

> For any label $(R, subs(R))$ appearing on some edge of proof structure of $CG_{init} \preceq_{\mathcal{KB}} CG_{goal}$, both R and $subs(R)$ are abstractable w.r.t ψ,

where we always assume $CG_{init} \preceq_{\mathcal{KB}} CG_{goal}$, whenever we talk about GDA for concept graphs. In this case, we also say that CG_{init} is an *abstractable concept graph for* CG_{goal} *w.r.t.* ψ. Moreover, an analogue of CG_{init} is defined as a concept graph $CG_{init}(Y_0 : u_0, ..., Y_k : u_k)$, where u_j is an analogue of s_j w.r.t. ψ for each j. ■

From the definition, the analogous concept graph $CG_{init}(Y_0 : u_0, ..., Y_k : u_k)$ is obtained by replacing every variable $X_j : s_j$ in the original CG_{init} with another variable $Y_j : u_j$ of analogous sort u_j, where $\psi(s_j) = \psi(u_j)$. Hence, if s_j belongs to a group of size $size(s_j)$, then the number of all possible analogous concept graphs is the product of $size(s_j)$ such that s_j appears in CG_{init}

The following proposition is a basis in transforming an initial concept graph CG_{init} to its analogue, CG_{sim}.

Proposition 3 *Suppose we have a label* $(R = R(X_1 : s_1, ..., X_n : s_n), subs(R))$ *in a proof structure of* $CG_{init} \preceq_{\mathcal{KB}} CG_{goal}$. *Then, for any analogue* R' *of* $var_abs(apply(R, subs(R)))$, *we have* $\mathcal{T} \vdash_{\mathcal{H}} R'$, *whenever* R *and* $subs(R)$ *are abstractable w.r.t. a grouping* ψ,

Proof. Suppose that $R = R(X_1 : s_1, ..., X_n : s_n)$ is instantiated to $R_1 = apply(R, subs(R)) = R(a_1 : t_1, ..., a_n : t_n)$ and that $\psi(t_j) = \psi(u_j)$ for each j. By the variable abstraction var_abs, we obtain $R_2 = var_abs(R_1) = R(Z_1 : t_1, ..., Z_n : t_n)$, and consider its arbitrary analogue $R' = R(Y_1 : u_1, ..., Y_n : u_n)$ of $var_abs(R_1)$. As $t_j \preceq s_j \in subs(R)$, $u_j \preceq v_j$ holds for some v_j such that $\psi(s_j) = \psi(v_j)$. Moreover the abstractableness of R implies that

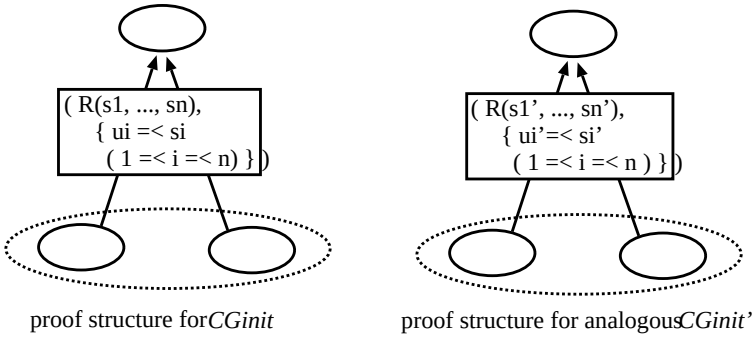


Figure 6: Relationship between Proof Structures

$\mathcal{T} \vdash_{\mathcal{H}} R(W_1 : v_1, ..., W_n : v_n)$. Hence, $\mathcal{T} \vdash_{\mathcal{H}} R(Y_1 : u_1, ..., Y_n : u_n)$ holds by the substitution $\{W_1/Y_1, ..., W_n/Y_n\}$. E.Q.D.

Remarks on Proposition 3:

The proposition provides us a basis of *similarity* between concept graphs as discussed later. Let us consider an initial concept graph CG_{init} and a goal CG_{goal}. Intuitively speaking, from the proposition, it is implied that if all labels in a proof structure of $CG_{init} \preceq_{\mathcal{KB}} CG_{goal}$ are abstractable under a grouping, then for any analogue CG'_{init} of CG_{init}, we can obtain the same proof structure of $CG'_{init} \preceq_{\mathcal{KB}} CG_{goal}$. It follows the observation below.

Let us consider a rule $R(X_1 : s_1, \ldots, X_n : s_n)$ containing variables of sorts s_i. For a concept graph CG_{init}, suppose we have a simple proof structure shown in Figure 6. Furthermore, suppose the label is abstractable under a grouping ψ. From the definition of abstractableness, any analogue of $R(X_1 : s_1, \ldots, X_n : s_n)$, $R(X_1 : s'_1, \ldots, X_n : s'_n)$, is derivable from our domain theory $\mathcal{KB}$, where $\psi(s_i) = \psi(s'_i)$. Furthermore, since each sort subsumption $u_i \preceq s_i$ in the label is abstractable, for any analogue u'_i of u_i, there exists an analogue s'_i of s_i such that $u'_i \preceq s'_i$ is valid in $\mathcal{KB}$. Therefore, for any analogue CG'_{init} of CG_{init}, we can construct the same proof structure by using an analogous rule of $R(X_1 : s_1, \ldots, X_n : s_n)$ and analogous subsumptions $u'_i \preceq s'_i$, as shown in Figure 6. It should be noted here that, precisely speaking, since such an analogous rule is derived from $\mathcal{KB}$ in general, it cannot be used directly in the construction of proof structure. This means that the actual proof structure would involve some additional steps to derive the analogous rule.

To illustrate the above definition and proposition, let us examine the last example shown in the preceding section, where we use a new concept graph

$$\texttt{CG12(X : s3)} = \{\texttt{r1(X : s3, Y : s5)}, \texttt{r2(Y, X)}\}$$

as an initial concept graph, for a while. It should be remarked that `CG12` is a generalization of `CG1(X:s3)` because of the rule `C3` and that the corresponding proof structure of $\texttt{CG12} \preceq_{\mathcal{KB}} \texttt{CG2}$ is obtained by cutting the leaf node `r3(a:s3,b:s5)` and one edge

$$\texttt{r3(a:s3,b:s5)} \xrightarrow{(\texttt{C3}, subs(\texttt{C3}))} \texttt{r1(a,b)}.$$

The substructure is circled by dotted line in Figure 5. When we consider a grouping

$$\psi_1=\{\{\texttt{s3},\texttt{s6}\},\{\texttt{s2},\texttt{s7}\},\{\texttt{s1}\},\{\texttt{s4}\},\{\texttt{s5}\}\},$$

that ignores the differences between `s3` and `s6`, also `s2` and `s7`, both the rule and sort subsumption in the label

$$\begin{array}{ll}(\texttt{C1}: & \texttt{p(X:s2,Y:s4):-q(X),r1(X,Y)},\\ & \{\ \texttt{s3}\preceq\texttt{s2},\ \texttt{s5}\preceq\texttt{s4}\})\end{array}$$

are abstractable. This is because, an analogue of `C1`,

$$\texttt{p(X:s7,Y:s4):-q(X),r1(X,Y)},$$

is in our rule set, and in addition, for an analogue `s6` of `s3` in $\texttt{s3}\preceq\texttt{s2}$, there exists an analogue `s7` of `s2` such that $\texttt{s6}\preceq\texttt{s7}$ holds in $\mathcal{H}$. Note that checking abstractableness for $\texttt{s5}\preceq\texttt{s4}$ is trivial, since `s5` is just one possible analogue of itself. From these observations and Proposition 3, we can assert that an analogue of $var_abs(apply(\texttt{C1}, subs(\texttt{C1})))$,

$$\texttt{p(Y1}:\texttt{s6},\texttt{Y2}:\texttt{s5}):-\texttt{q(Y1)},\texttt{r1(Y1,Y2)}$$

is a valid rule under our domain theory $\mathcal{KB}$. Similarly, with respect to the same ψ_1, `C2` and $subs(\texttt{C2})$ are abstractable, and it turns out that an analogue $var_abs(apply(\texttt{C2}, subs(\texttt{C2})))$, `q(Z1:s6)`, is a logical consequence of $\mathcal{KB}$. Hence, a new concept graph defined as

```
CG12(X:s6)= { r1(X:s6,Y:s5), r2(Y,X) }
```

is subsumed by the goal `CG2(X:s1)`, using the analogue rule of `C1`, `C1(s7,s4)=p(X:s7,Y:s4):-q(X),r1(X,Y)`, and the analogue of `C2`, `C2(s7)=q(X:s7)`, and their sort subsumptions $\texttt{s6}\preceq\texttt{s7}$, $\texttt{s5}\preceq\texttt{s4}$. Strictly speaking, the proof structure of the initial `CG12(X:s3)` and its analogue `CG12(X:s6)` for the goal `CG2(X:s1)` becomes the same one, provided we allow the derived rules `C2(s7)` and `C1(s7,s4)` as possible edges of proof structures. In this sense, we say that the subsumptions $\texttt{CG12(X:s3)}\preceq_{\mathcal{KB}}\texttt{CG2(X:s3)}$ and $\texttt{CG12(X:s7)}\preceq_{\mathcal{KB}}\texttt{CG2(X:s1)}$ are almost same. Note that we require the sort of root variable `s6` is still a subsort of root variable `s1` of goal, `CG2(X:s1)`.

Now we are ready to define a notion of *similarity between concept graphs* CG_{init} and CG_{sim} from the viewpoint of goal CG_{goal}

Definition 10 (Similarity between Concept Graphs)
A concept graph $CG_{sim}(Z:u, Z_1:u_1, ..., Z_k:u_k)$ is said to be similar to $CG_{init}(X:s)$ w.r.t.a goal CG_{goal}, if there exists a grouping ψ satisfying the following two conditions:

1. ψ is a GDA for CG_{goal} from CG_{init}. That is, CG_{init} is an abstractable concept graph for CG_{goal} w.r.t. ψ.
2. CG_{sim} is subsumed by some analogue of CG_{init} w.r.t. ψ.

■

In a word, an analogous concept graph keeps almost similar proof, and the similar concept graph may be a specialization of some analogous graph. So the similar graphs can involve their own particular information. Some concrete examples are shown in Section 6.

Moreover, from the viewpoint of information retrieval, every concept graph similar to the initial graph CG_{init} w.r.t. some GDA ψ for the goal CG_{goal}, is understood as a relevant concept graph to the query specified by the pair $\langle CG_{goal}, CG_{init}\rangle$.

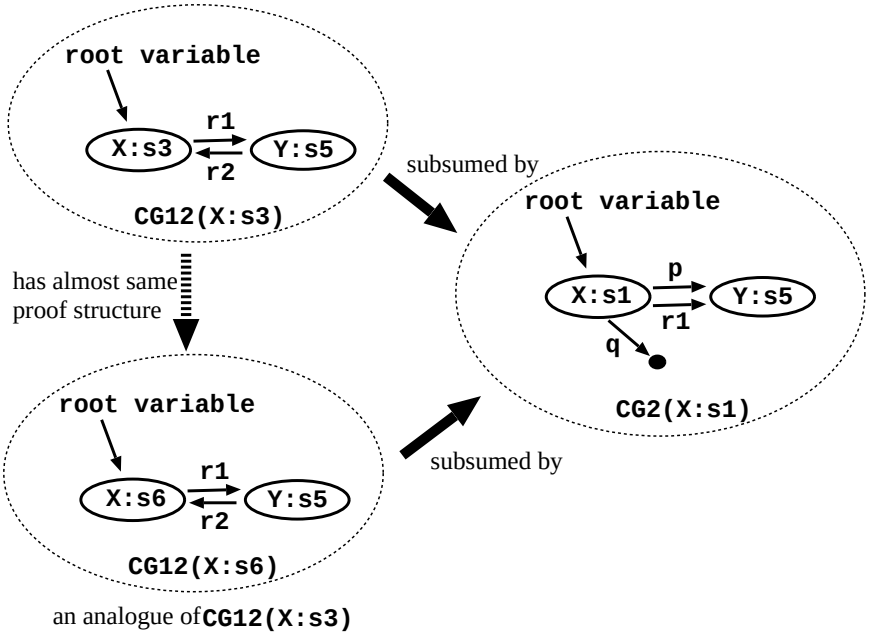


Figure 7: An analogous concept graph subsumed by the same goal

5.1 Needs of Generalization

As is easily expected, as the number of used rules and their corresponding sort subsumptions increases, it is often the case that the number of all possible analogous concept graphs decreases. As an extreme case, that number is just 1. This case happens when there exists just one GDA which is a trivial grouping

$$\psi = \{\{s\} | s \in S\}.$$

In this case, nothing except concept graph subsumed by CG_{init} is judged relevant to our query $\langle CG_{goal}, CG_{init} \rangle$.

To cope with such a problem, we suppose a control mechanism for adjusting the level of generalization by which we can look for an adequate level. We present in the next section an algorithm for performing such a generalization. This subsection simply motivates it.

Now let us return to the previous example of concept graphs in Figure 4. We already know two subsumptions: $\texttt{CG1(X:s3)} \preceq_{\mathcal{KB}} \texttt{CG12(X:s3)}$ and $\texttt{CG12(X:s3)} \preceq_{\mathcal{KB}} \texttt{CG2(X:s1)}$. As is shown in Figure 5, we are forced to use three labels of the proof structure of $\texttt{CG1(X:s3)} \preceq_{\mathcal{KB}} \texttt{CG2(X:s1)}$. Two of them are shared by the proof structure of $\texttt{CG12(X:s3)} \preceq_{\mathcal{KB}} \texttt{CG2(X:s1)}$. The remaining one is

$$\texttt{r3(a:s3,b:s5)} \xrightarrow{(\texttt{C3}, subs(\texttt{C3}))} \texttt{r1(a,b)}.$$

It has been already shown that CG12(X:s3) is abstractable for CG2(X:s1) w.r.t. ψ_1:

$$\psi_1 = \{\{\texttt{s3,s6}\}, \{\texttt{s2,s7}\}, \{\texttt{s1}\}, \{\texttt{s4}\}, \{\texttt{s5}\}\}.$$

When our similarity situation is given by a pair $\langle CG_{goal}, CG_{init} \rangle = \langle \texttt{CG1,CG2} \rangle$, C3 and $subs$(C3) must be abstractable. However, C3 is not abstractable w.r.t.

ψ_1, as a ψ-analogue of `C3`,

$$r1(X : s7, Y : s4) : -r3(X, Y),$$

is not provable under our domain theory. Thus, as long as we like to treat `C3` as a candidate of abstractable rule, two sorts `s2` and `s7` should not be grouped together. Another candidate grouping is to put `s3` and `s6` into the same group, classifying `s2` and `s7` into distinct groups at the same time. However, in this case, the used subsumption `s3`$\preceq$`s2` becomes non-abstractable. As a result, GDA ψ becomes a trivial grouping when we are thus particular about `C3`. In such a case, based on the generalization, we consider `CG12` instead of `CG1` for which non-trivial ψ_1 is regarded as a right GDA for $\langle$`CG12,CG2`$\rangle$,

5.2 Generalization

The generalization of concept graphs, given $\langle CG_{goal}, CG_{init}\rangle$, is simply defined as a "contraction process" from leaves to root in our proof structure. Another possible method is to go down the proof structure from its root to leaves. However, we prefer a bottom up approach by some reason shown in the next section.

Suppose we have a proof structure of $CG_1 \preceq_{\mathcal{KB}} CG_2$.

1. If there exists just one vertex in the proof structure, there is no way to generalize it.
2. Suppose there exists an edge from a leaf B_{in_body}

   $$(B_{in_body} \xrightarrow{(R, subs(R))} B_{head})$$

   such that B_{in_body} is a Skolemized atom in CG_1.

   (a) Collect all vertex $B_1, ..., B_k$ with the same label $(R, subs(R))$ to B_{head}, including B_{in_body} as well as instances of unit clauses in $\mathcal{T}$. Then, by the definition of proof structure, $B_{head} : -B_1, ..., B_k$ is an instance of R.

   (b) If every B_j is a leaf node, then remove all the B_j and the edges from our proof structure. Then the set of leaves except instances of unit clauses in $\mathcal{T}$ forms a generalized concept graph, where we convert every Skolemized constant to a corresponding variable.

   (c) Otherwise, try another leaf node as a possible candidate for our generalization.

An example is already presented in Figure 5. For `r1(a:s3,b:s5)` is the unique Skolemized atom appearing as a leaf, `CG12 =` {`r1(X:s3,Y:s5)`} is the result of generalization. This `CG12` does not coincide with the `CG12` in Figure 7, as the latter involves the edge `r2(Y,X)` in it. However, `r2(Y,X)` has no effect on the proof structure of goal, so we neglect such an atom.

6 Experimental Results

In this section, we show our preliminary experimental results obtained by our current system [‡] .

[‡] The system has been implemented in SICStus Prolog and run on a UltraSPARC Workstation(UltraSPARC II-i 440MHz).

We have implemented the main module of our conceptual classification system, that is, the module for finding groupings among sort concepts. Let $\mathcal{KB}$ be a domain theory, and CG_{init} and CG_{goal} be an initial and goal concept graphs, respectively. Assume we have a proof structure of $CG_{init} \preceq_{\mathcal{KB}} CG_{goal}$. Given the proof structure, our system tries to find GDA ψ_i for CG_{goal} from CG_{init}, that is, CG_{init} is an abstractable concept graph for CG_{goal} w.r.t. ψ_i.

Before showing our experimental results, we give an overview of the computation process in our system.

Overview of Our System:

Given the proof structure as input, our system finds all *maximal* GDA ψ_i for CG_{goal} with a generate-and-test strategy, where the ordering on groupings is defined as follows:

Definition 11 (Ordering on Groupings)
Let $\psi = \{S_1, \ldots S_k\}$ and $\psi' = \{S'_1, \ldots S'_l\}$ be two groupings of a sort set S. Then ψ precedes ψ' (denoted by $\psi \preceq \psi'$) if for each S_i in ψ, there exists some S'_j in ψ' such that $S_i \subseteq S'_j$. ■

Given a set of sorts S, the possible groupings of S forms a lattice under the ordering: the minimum (that is, trivial) grouping is placed at the bottom and the maximum grouping at the top.

For each generated candidate ψ_i of grouping, we examine whether all labels $(R_i, subs(R_i))$ in the proof structure are abstractable under ψ_i or not. If all of them are abstractable under ψ_i, it is said to be *appropriate*.

Since we are interested in maximal appropriate groupings, one may consider that candidate generation should be performed in a top-down manner. However, our system generates candidate groupings in a bottom-up manner. There are two reasons why we do so:

- We expect that appropriate groupings would be found, if anything, at lower positions in the candidate lattice.
- Based on a property w.r.t. the appropriateness of candidates, we can prune candidate generation in the upper direction in the lattice.

Since the former is merely empirical, there might exist some cases in which top-down generations would be adequate for our purpose. However, we have not seen such a case yet. On the other hand, the soundness of the candidate pruning is theoretically ensured. It is based on the following observation:

> For a candidate grouping ψ, if the rule R of a label $(R, subs(R))$ in the proof structure is not abstractable under ψ, then any grouping ψ' such that $\psi \preceq \psi'$ is not appropriate.

Our candidate generation process is carried out with the pruning strategy based on this property.

Moreover, we can obtain further pruning from another viewpoint. The candidate lattice contains a large number of needless ones. We are interested only in groupings that are concerned with sorts occurring in the initial concept graph CG_{init}. Therefore, any candidates other than such interesting ones should be rejected. These needless candidates are not generated in our generation process. The number of candidates that our system would generate in the worst case is denoted by "Num. of Possible Cands." in the

experimental results we will show later. Moreover, the numbers of candidates that are *actually* generated and examined are denoted by "Num. of Generated Cands." and "Num. of Examined Cands.", respectively.

As discussed in the previous section, there often exists a case where the trivial (minimum) grouping is the only appropriate one. In such a case, we try to generalize the initial concept graph. Since we have some possible generalizations of the graph, a certain preferential order on possible ones would be necessary. In our system, we prefer a generalization which is obtained by removing less number of edges and for which a non-trivial grouping can be obtained as an appropriate one.

For our preliminary experimentations, two examples have been provided. We show the results for the examples and discuss them below.

Example1: Dangerous Event

Let us consider a domain theory $\mathcal{KB}_D$ and an initial concept graph $\mathtt{CG}_{\mathtt{init1}}$ shown in Figure 8. In the figure, any variable with which no sort is associated is assumed to be typed with a sort `top` subsuming any sort. Since it is clear, any sort subsumption of the form $\mathtt{s_i} =< \mathtt{top}$ is omitted.

$\mathcal{KB}_D$ is a domain theory concerning a dangerous event. For example, a rule

$$\mathtt{mv(X : vehicle)} : -\mathtt{has(X, Y : rot_part), maintained(Y)}.$$

states that if a vehicle has rotational part that is maintained, then the vehicle is movable.

Suppose we have a goal

$$CG_{goal1}(\mathtt{E : event}) = \{\mathtt{dangerous(E)}\}.$$

It is easily verified that CG_{init1} is subsumed by $CG_{goal1}(\mathtt{E : event})$ under the domain theory. The proof structure of the subsumption is shown in Figure 9.

Given the set of labels $(\mathtt{C_i}, subs(\mathtt{C_i}))$ in the proof structure as input, our system tries to find maximal appropriate groupings among sort symbols. The obtained results are shown in Figure 10.

Since there are 13 sort symbols in the example, the number of possible candidates (including needless ones) is 27,644,437 ($\approx 3 \times 10^7$). As mentioned above, our system would generate approximately 7×10^6 of them in the worst case by rejecting needless candidates.

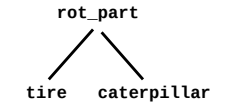
In order to find all maximal grouping, only 473 candidates have actually been generated and 191 of them examined their appropriateness. It should be emphasized here that 99.99% of possible candidates have been pruned with our pruning strategy for candidate generation. Therefore, it is considered that our strategy can work very well to improve efficiency of finding desired groupings.

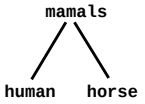
There are 3 groups of similar sorts in the appropriate grouping:

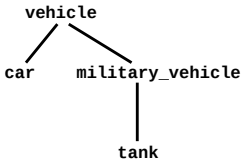
[`car`, `tank`, `military_vehicle`, `vehicle`],
[`tire`, `caterp`, `rot_part`] and [`human`, `pet`].

Note that the first two groups are implied from the sort subsumptions in the domain theory as well as from the result. Therefore, one might prefer more interesting groupings. In order to obtain such a grouping, we try to generalize the original concept graph CG_{init1} with the label

$$(\mathtt{C7 : mv(X : vehicle)} : -\mathtt{has(X, Y : rot_part), maintained(Y)},$$
$$\{\mathtt{car} =< \mathtt{vehicle}, \mathtt{tire} =< \mathtt{rot_part}\})$$



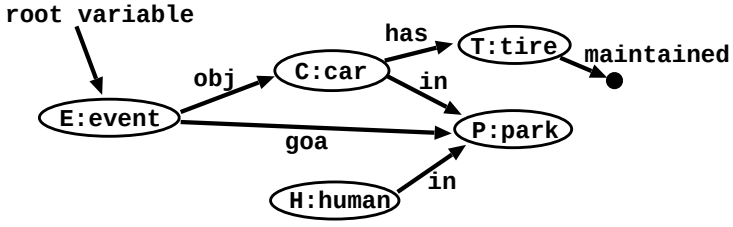




Sort Hierarchy

```
C1: receive_damage_mv_obj(X:pet,Y):-
       mv(Y), bigger_than(Y,X).

C2: receive_damage_mv_obj(X:human,Y):-
       mv(Y), bigger_than(Y,X).

C3: bigger_than(X:vehicle,Y:human).

C4: bigger_than(X:vehicle,Y:pet).

C5: bigger_than(X:horse,Y:human).

C6: bigger_than(X:horse,Y:pet).

C7: mv(X:vehicle):-
       has(X,Y:rot_part), maintained(Y).

C8: mv(X:horse):-
       has(X,Y:leg), maintained(Y).

C9: dangerous(E:event):-
       object(E,X), goa(E,Y:park),
       in(X,Y), bigger_than(X,Z), in(Z,Y),
       to_be_protected_against_mv_obj(Z,X)

C10: to_be_protected_against_mv_obj(X,Y):-
       mv(Y), receive_damage_mv_obj(X,Y).
```

Rule Set



Concept Graph

Figure 8: Domain Theory and Concept Graph Concerning Dangerous Event

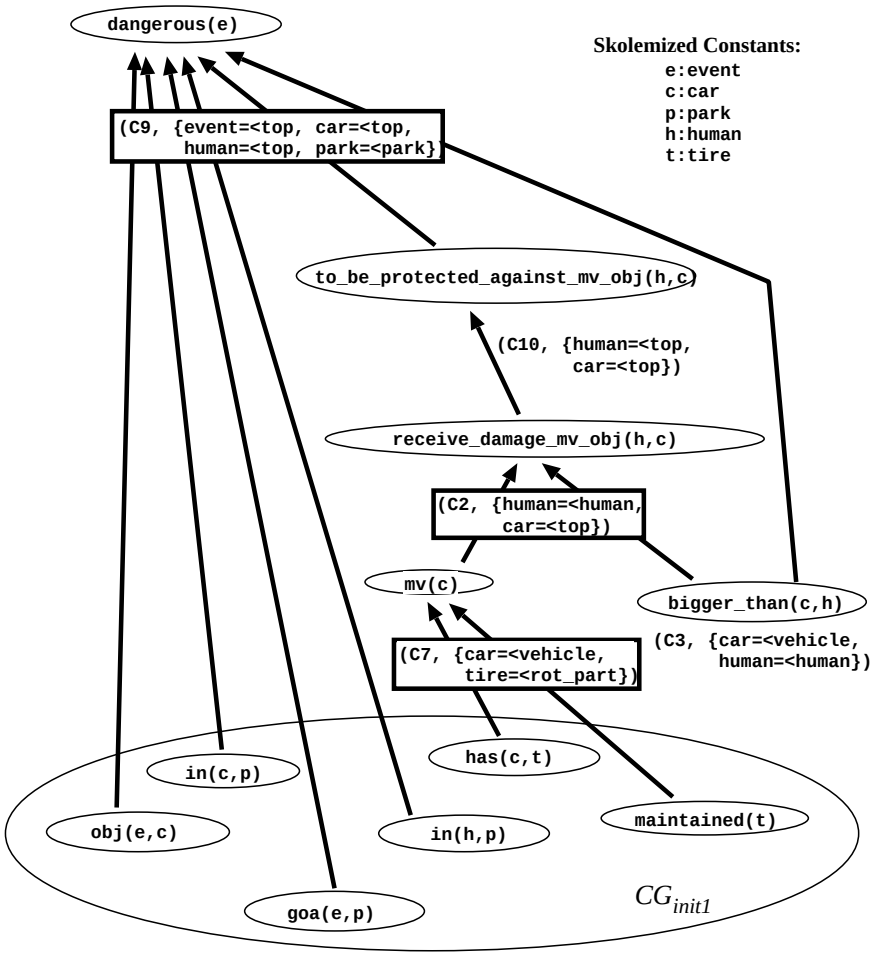


Figure 9: Proof Structure of $CG_{init1} \preceq CG_{goal}(\texttt{E:event})$

Num. of Possible Cands	7,322,937 ($\approx 7 \times 10^6$)
Num. of Generated Cands	473
Num. of Examined Cands	191
Execution Time	7,643 msec.
Maximal Appropriate Grouping	[[car, tank, military_vehicle, vehicle], [tire, caterp, rot_part], [human, pet], [park], [event]]

Figure 10: Experimental Results

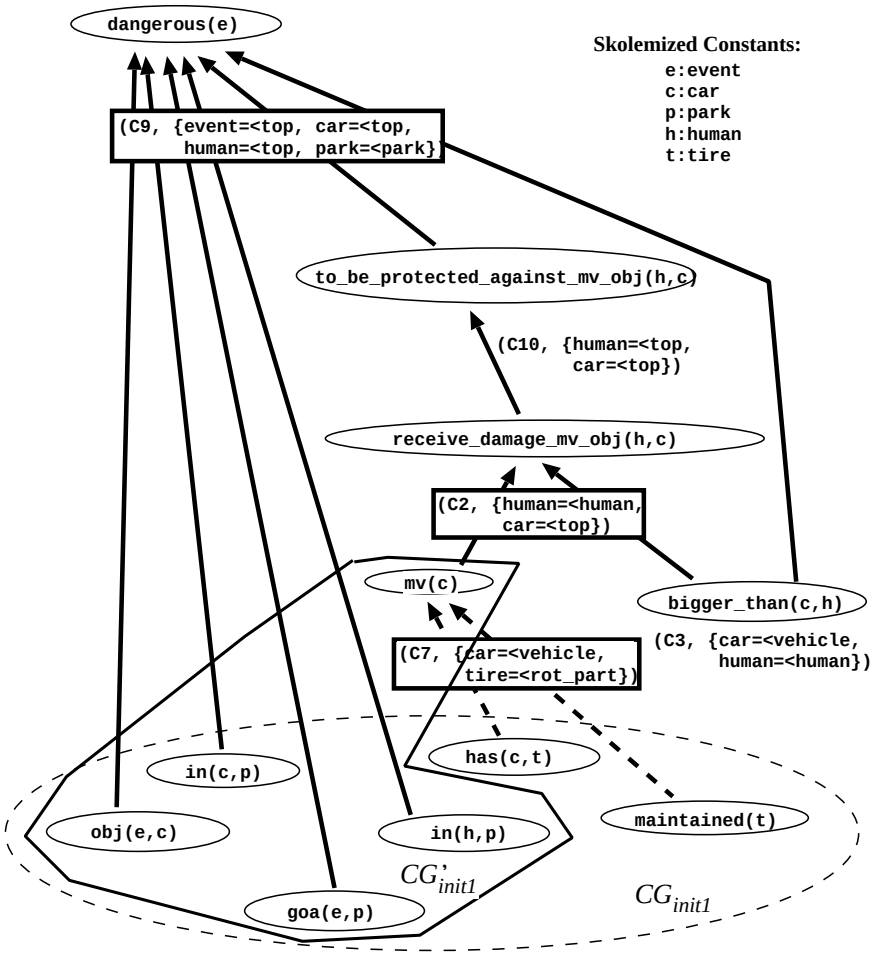


Figure 11: Proof Structure of $CG'_{init1} \preceq CG_{goal1}(\mathtt{E} : \mathtt{event})$

in the proof structure. Figure 11 shows the generalized concept graph CG'_{init1} and the proof structure of $CG'_{init1} \preceq CG_{goal1}(\mathtt{E} : \mathtt{event})$

After the generalization, by taking the labels in the proof structure as input, our system can find a more interesting appropriate grouping shown in Figure 12.

The obtained grouping shows that we can consider `car`, `tank`, `military_vehicle`, `vehicle` and `horse` to be analogous. It should be emphasized that it is not implied from the given domain theory. Thus, we can find a more interesting grouping by taking a generalization of CG_{init1} into account. Furthermore our pruning strategy for candidate generation has still worked very well to reduce the computational cost.

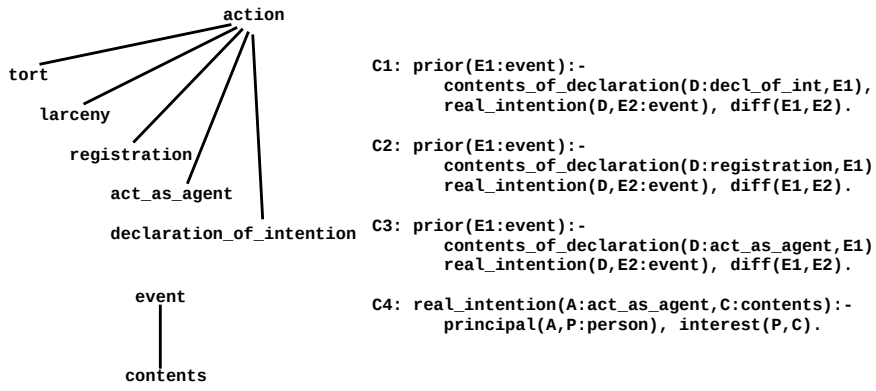
Example2: Priority on Legal Event

Let us consider a legal domain theory $\mathcal{KB}_P$ and a concept graph CG_{init2} shown in Figure 13.

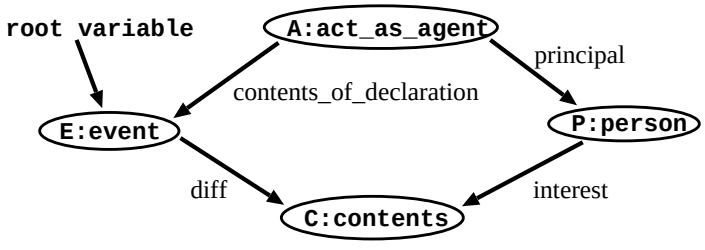
$\mathcal{KB}_P$ is a domain theory concerning priority on legal events. For example, a rule $C2$ states that if a registration declares an event and the real intention of the registration is contrarily to declare a different event, then the registration should be prior to the real intention.

Num. of Possible Cands	
Num. of Generated Cands	222
Num. of Examined Cands	73
Execution Time	2,193 msec.
Maximal Appropriate Grouping	`[[car, tank, military_vehicle, vehicle, horse], [human, pet], [tire], [park], [event]]`

Figure 12: Experimental Results after Generalization



Sort Hierarchy Rule Set



Concept Graph

Figure 13: Domain Theory and Concept Graph Concerning Priority on Legal Event

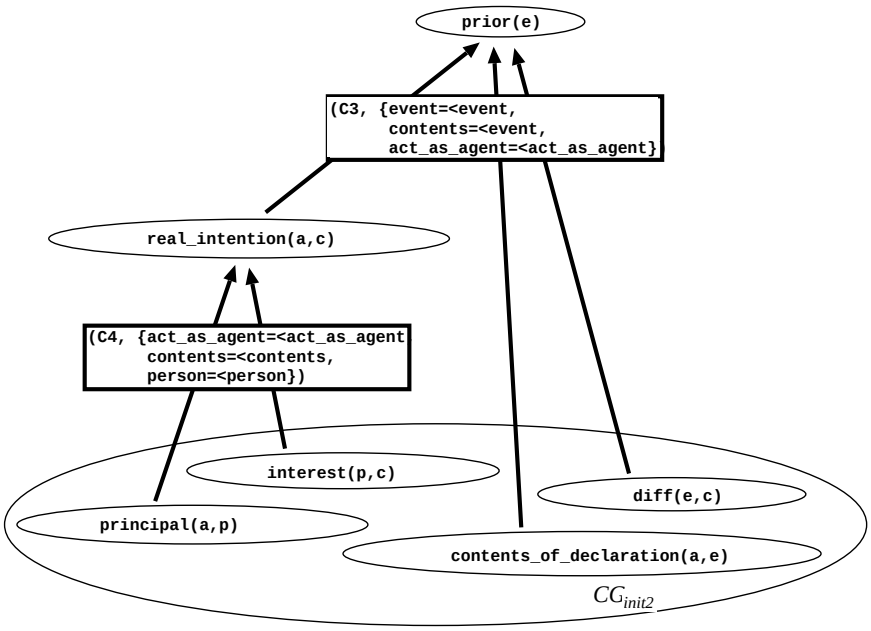


Figure 14: Proof Structure of $CG_{init2} \preceq CG_{goal2}(\mathtt{E} : \mathtt{event})$

Suppose we have a goal

$$CG_{goal2}(\mathtt{E} : \mathtt{event}) = \{\mathtt{prior}(\mathtt{E})\}.$$

It is easily verified that CG_{init2} is subsumed by $CG_{goal2}(\mathtt{E} : \mathtt{event})$ under the domain theory. The proof structure of the subsumption is shown in Figure 14.

Given the set of labels $(C_i, subs(C_i))$ in the proof structure as input, our system, our system can find only one maximal appropriate grouping. Unfortunately, it is a trivial grouping. We therefore try to generalize the original concept graph CG_{init2} with the label

$$(\mathtt{C4} : \mathtt{real_intention}(\mathtt{X} : \mathtt{act_as_agent}, \mathtt{Y} : \mathtt{contents}) : -$$
$$\mathtt{principal}(\mathtt{X}, \mathtt{Z} : \mathtt{person}), \mathtt{interest}(\mathtt{Z}, \mathtt{Y}),$$
$$\{\mathtt{act_as_agent} =< \mathtt{act_as_agent},$$
$$\mathtt{contents} =< \mathtt{contents}, \mathtt{person} =< \mathtt{person}\}$$

Figure 15 shows the generalized concept graph CG'_{init2} and the proof structure $CG'_{init2} \preceq CG_{goal2}(\mathtt{E} : \mathtt{event})$

After the generalization, our system can find another grouping as an appropriate one. The obtained grouping is shown in Figure 16.

As shown in the results, we can consider that `act_as_agent`, `registration` and `declaration_of_intention` to be similar, even though any two sorts are not similar without generalization. Thus an interesting grouping can be found with the help of generalization.

Furthermore, the number of actually generated candidates is only 0.38% of possible ones. It is, therefore, verified that our pruning strategy of candidate generation can still work very effectively in this example.

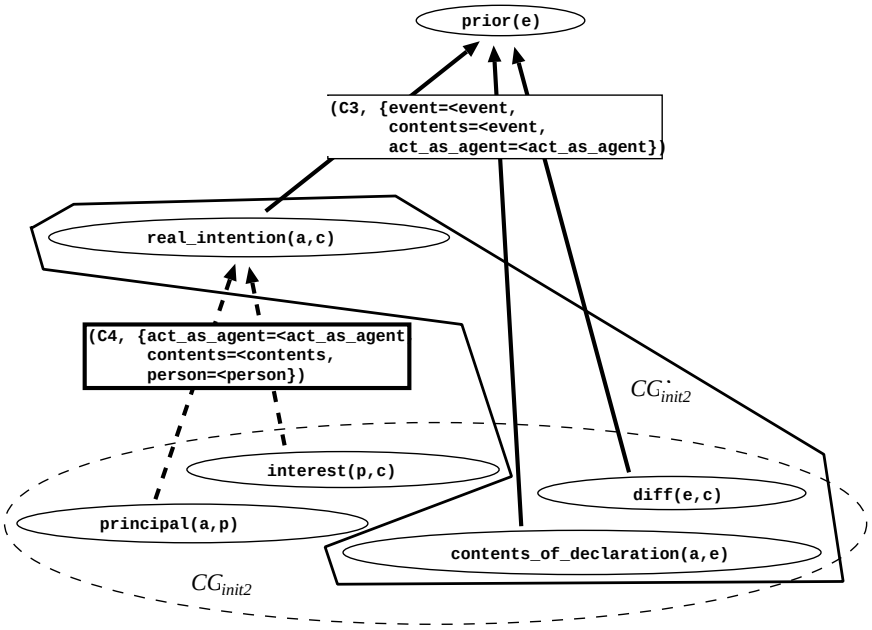


Figure 15: Proof Structure of $CG'_{init2} \preceq CG_{goal2}(\mathtt{E} : \mathtt{event})$

Num. of Possible Cands.	11,002
Num. of Generated Cands.	42
Num. of Examined Cands.	26
Execution Time	4,253 msec.
Max. Appropriate Grouping	`[[act_as_agent, registration, declaration_of_intention], [contents, event], [person]]`

Figure 16: Experimental Results after Generalization

7 Discussion

According to the present version of this research, we do not use any negative information, as the underlying logic is order-sorted Horn logic. However, the use of negative information such as disjoint constraints [14] is useful to obtain richer and meaningful class of goal-dependent abstractions. For instance, suppose the following rule is used in a concept graph subsumption $CG_{init} \preceq_{\mathcal{KB}} CG_{goal}$:

```
movable(X:car):-has(X,Y:tire), maintained(Y).
```

When we examine a similarity between car and horse, the present method has to check the abstractableness for a grouping such as

$$\psi = \{\{\texttt{car, horse, ...}\}, \{\texttt{tire, leg, ...}\}, \ldots\},$$

for instance. Then, an analogous rule

```
movable(X:horse):-has(X,Y:tire), maintained(Y)
```

is tested for its validness. However, it is not provable under a standard theory, so the grouping ψ is rejected. It is meaningless not to allow ψ because of such a reason. A possible way to resolve this kind of anomalies is to introduce some negative information or completion of constraint rules to prove that there does not exist any `X:horse` satisfying `has(X,tire)`. Then, the body part becomes false, so the analogous rule is evaluated as a logically valid rule, and is never used due to the constraint. This will enable us to take an adequate grouping ψ as a right GDA.

Another serious problem we have to consider is the incompleteness of our domain theory. The problem is troublesome, since it may be concerned with default knowledge we use. However, it seems that we can have at least two methods to cope with it. We suppose that a user presents an initial goal G, initial situation E_0 from which G is derived. In addition to the pair $\langle G, E_0 \rangle$ as a query to our system, suppose that a set of training instance situations some of which should be similar to E_0 w.r.t. G and some of which should not be. When our domain theory $\mathcal{T}$ is inadequate, then the system will return a false answer about the similarity opposed to user's intention. The methods to complete and to adjust $\mathcal{T}$ will depend on a reason why such an inadequateness is derived under the present $\mathcal{T}$.

Case 1: Some rule is missed in $\mathcal{T}$ and is never constructed from any rule in $\mathcal{T}$ by both specialization and generalization. In this case, we need to have a rule relating the goal with both E_0 and E_j. Abductive ILP might be successful for this purpose.

Case2: Some missing rule is constructed from a rule in $\mathcal{T}$ by generalizing or by specializing it. We have already tried to apply the specialization to adjust or to revise the existing rule so that wrong similarities are never concluded by the system under a new revised theory [8]. It will be interesting when we try the generalization so that it constructs a new revised theory under which only a right similarity between E_0 and E_j become observed by our system.

Finally, we must discuss how we provide a class of concept graphs. It is generally hard to answer this question. However, recent development of natural language processing for text corpus will be regarded. From a text, we can extract various kinds of relations between words in the form of case structure or of dependency triples. They can be pieces of our concept

graphs, where words correspond to our sorts. A concept graph would be extracted by gathering such information from a restricted small part of corpus. Moreover, some information theoretic measure is available to estimate an actual similarity between words appearing in the large text and to exclude non-relevant similarities. Using such a measure, it seems to complete the extracted concept graph and to design a heuristic control to find an appropriate grouping of sorts at the same time.

References

[1] J.D.Tenenberg: Abstracting first-order theories, in *Change of Representation and Inductive Bias*, D.P.Benjamin ed., Kluwer Academic Publishers, 67–79, 1989.

[2] D. A. Plaisted: Theorem Proving with Abstraction, *Artificial Intelligence*, 16, 47–108, 1981.

[3] T.Kakuta, M.Haraguchi & Y.okubo: A Goal-Dependent Abstraction for Legal Reasoning by Analogy *International Journal of Artificial Intelligence and Law*, 5, 1-2, 97–118, 1997.

[4] Y. OKUBO and M. HARAGUCHI, Constructing Predicate Mappings for Goal-Dependent Abstraction, *Annals of Mathematics and Artificial Intelligence*, Vol. 23, No. 1–2, pp. 169–197, 1998

[5] K.Kasahara, K.Matsuzawa, T.Ishikawa A method for judgment of semantic similarity between daily-used words by using machine readable dictionaries, Journal of JSIP, 38, 7, 1272–1283, 1997.

[6] H.Ait-Kaci, A.Podelski : Towards a meaning of LIFE, Proc. of 3rd PLILP (Programming Language Implementation and Logic Programming, 255–274, Springer-Verlag, 1991.

[7] A.Y.Levy & M.C.Rousset: Combining Horn rules and description logics in CARIN, Artificial Intelligence 104, 165–209, 1998.

[8] N.Morita, M.Haraguchi & Y.okubo: A method of similarity-driven knowledge revision for type specialization, Springer LNAI 1720, 194–205, 1998.

[9] W.W.Cohen & H.Hirsh: The Learnability of Description Logics with Equality Constraints, *Machine Learning*, Vol. 17, No. 2–3, pp 169–199, 1996.

[10] C.Beierle, U.Hedtstuck, U.Pletat, P.H.Schmitt, J.Siekmann: An order-sorted logic for knowledge representation systems, Artif. Intell. (Netherlands), vol.55, no.2–3, 149-91, 1992.

[11] Christph Walter, *Many-Sorted Unification*, JACM, 35, 1, 1–17, 1988.

[12] Brachman,R.J., Borgida,A., & McGuinness,D.L.: Living with CLASSIC, In J.Sowa ed. *Principles of Semantic Networks*, 401–456, Morgan Kaufmann, 1991.

[13] J. W. Lloyd: *Foundations of Logic Programming*, Springer, Berlin, 1984.

[14] B.Nebel: *Reasoning and Revision in Hybrid Representation Systems*, Springer LNAI Vol. 422, 270 pages , 1990.

[15] T. M. Mitchell, R. Keller and S. Kedar-Cabelli, *Explanation-Based Generalization : An Unifying View*, Machine Learning, vol.1, no.1, pp.47-80, 1986.