On a Complex Object Model over Specialization Constraints Supporting Selective Inheritance

Nobutaka Suzuki Minoru Ito
nobuta-s@is.aist-nara.ac.jp ito@is.aist-nara.ac.jp

Graduate School of Information Science, Nara Instutute of Science and Technology
Ikoma, Nara 630-01, Japan

Abstract: For a complex object model, a form of range restriction, called *specialization constraint* (SC), has been studied. On the other hand, very few models have been proposed that support selective inheritance. In this paper, the following consideration is taken into a complex object model which supports selective inheritance. A polynomial-time algorithm is presented for deciding if a given database schema is well-formed. Sound and complete axiomatization for SCs are given. A polynomial-time algorithm is given that decides if an SC is a logical consequence of a set of SCs. Finally, another polynomial-time algorithm is given, which decides if there exists a database that contains a given path.

Keywords: complex object, selective inheritance, object-oriented database

1 Introduction

For a complex object model, a form of range restriction, called *specialization constraint* (SC), has been studied [1, 2, 4]. On the other hand, very few models have been proposed that support the feature of both selective inheritance and complex object, though the object-oriented model proposed in [3] supports non-monotonic inheritance.

In this paper, we consider a complex object model based on SC supporting *selective inheritance*, which allows a subclass not to inherit some of the properties from its superclasses. To illustrate our model, let us consider the example shown in Figure 1. This example represents a schema that specifies relationship among several kinds of vehicles and other primitive classes. Each class is represented by a labeled rectangular box of solid line, where the label denotes a class name with sets of immediate properties and of no-inherited properties. No-inherited properties are those specified not to be inherited from the superclasses. A shaded rectangular box denotes a virtual class, which does not have its own objects. In Figure 1, only class Vehicle is virtual, which implies that every object in class Vehicle must be in one of its subclasses, WaterVehicle, AirVehicle, and so on. (This is usually called a cover constraint.) A set of classes is represented by a unlabeled rectangular box of dotted line. We represent SCs as labeled directed arcs of solid line, where the label is either `Id` or a property. If `Id` is labeled, then the arc means that each object in the 'from' classes must be in 'to' classes. The arc between classes Real and Int means that an object in Int must be in Real. If a property P is labeled, then it means that every object in the 'from' classes must have an object in the 'to' classes as

the property P value. For example, an object in class Car or Bicycle must have an object in class Real as the "wheelbase" property. A thick gray arc represents an ISA relationship; that is, the 'from' class is a subclass of the 'to' class. Class SeaPlane is an example of multiple inheritance. SCs with respect to a class are inherited to its subclasses. Hence the weight of a WaterVehicle object, as well as that of Vehicle object, must be in Real. By "NoInherit" phrase, we represent no-inherited properties. Since class Bicycle does not inherit "maxPower" from class LandVehicle, class Bicycle has four properties, "weight", "cost", "capacity", and "wheelbase". Note that if a class C has a no-inherited property P, then SCs with respect to the superclasses of C and property P are not inherited to C. For example, the SC stating that an object in class LandVehicle must have an object in class Int as the "maxPower" property is no longer valid for class Bicycle that is a subclass of LandVehicle.

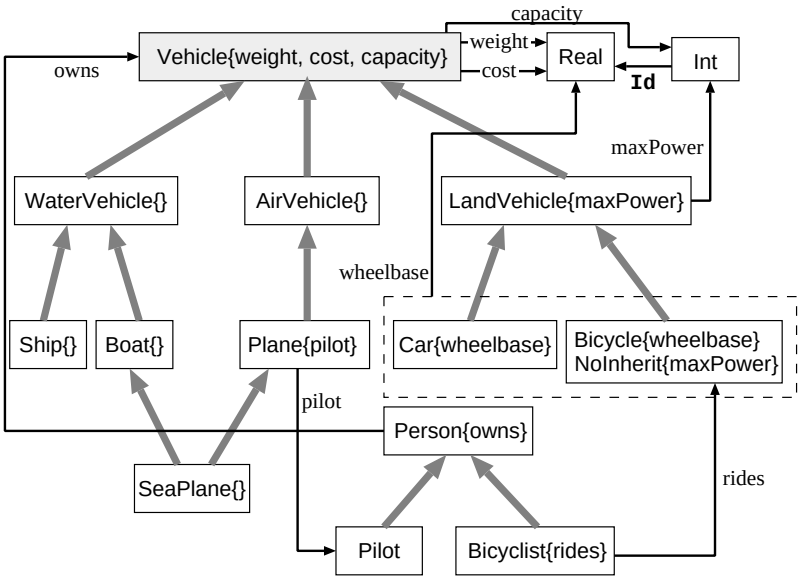

Figure 1: VEHICLE Schema Diagram

In this paper, we introduce a concept of *well-formedness* of a database schema, which guarantees that for each class C, there exists a database satisfying the following two conditions:

1. If C is actual, then the database contains an object in C.
2. If C is virtual, then the database contains an object in an actual subclass of C.

Intuitively, if a database schema is not well-formed, then it contains a redundant class in the sense that (1) no database has an object in the class or (2) for any database, no object in the class has an object as its property. For example, assume that there is a database schema that

contains a virtual class C having no subclass. Note that C cannot contain its own object by definition — there exists no subclass of C containing an object. Thus the database schema is not well-formed. Let us consider another database schema, which has two SCs with respect to a class C and its property P. The one and the other SCs assert constraints that C must have an object in classes C_1 and C_2 as property P value, respectively. Furthermore, assume that C_1 and C_2 are different and that there exists no common subclass of C_1 and C_2 in the class hierarchy of the schema. Then C cannot have any object. Assume, contrary, that there is an object in C. The object must have an object as property P value, say $\mathbf{o}$, which must be in both C_1 and C_2. But since (1) C_1 and C_2 are different and (2) there exists no common subclass of C_1 and C_2, there exists no class that $\mathbf{o}$ can be in, which is a contradiction. Thus the database schema is not well-formed, either. Clearly, a database schema should be well-formed. In Section 3, we show a polynomial-time algorithm for deciding whether or not a given database schema is well-formed. On the assumption that a database schema is well-formed, we consider two problems: *implication problem* and *the possibility of a path function.*

Implication problem is one deciding whether or not a given SC is a logical consequence of a set of SCs. The problem relates to a type check of a predicate in a query. For example, in the query $\sigma_{\text{owns.cost} = 99.8}(\mathsf{Person})$, we must have a meaningful answer only if $\mathsf{Person}(\text{owns.cost:}\mathsf{Real})$ is a logical consequence of the set of SCs given in the VEHICLE schema. In Section 4, we give a polynomial-time algorithm for deciding this problem.

Finally, we consider a problem of the possibility of a path function. A path function is either `Id` (short for *identity*), which denotes a path of zero length, or a sequence of properties. A path function pf is *possible* with respect to a class if there exists a database that contains a path described by pf from an object in the class. The problem relates to finding a "meaningless " path in a query. For example, in the query, $\sigma_{\text{rides.maxPower} = 30}(\mathsf{Bicyclist})$, the path function "rides.maxPower" is meaningless, since class Bicyclist does not have property "maxPower", and thus there cannot exist any object that can be reachable through the path function from an object in class Bicyclist. In Section 5, we give a polynomial-time algorithm for deciding the problem.

2 Definitions

A *database schema* is an ordered pair $\langle \mathbf{S}, \Sigma \rangle$, where $\mathbf{S}$ is a *class schema* and Σ is a set of SCs. $\mathbf{S}$ is a set of declarations of the form

$$[\textbf{virtual}]\ \textbf{class}\ C\{P_1, \ldots, P_l\}\ \textbf{isa}\ \{C_1, \ldots, C_m\}\ \textbf{NoInherit}\ \{P'_1, \ldots, P'_n\}$$

where C is a *class*, each P_i is an *immediate property* of C, each C_j is an *immediate superclass* of C and is different from C, and each P'_k is a *no-inherited property* for C; that is, C does not inherit property P'_k from its superclasses. If "**virtual**" is declared for C, then C is *virtual*; otherwise it is *actual*. The set of classes in $\mathbf{S}$ is denoted $Classes(\mathbf{S})$, the set of immediate properties of C is denoted $Props(C)$, and the set of no-inherited properties for C is denoted $NoInherit(C)$. By $C \Rightarrow C'$, we mean that C' is an immediate superclass of C. The reflexive transitive closure of "$\Rightarrow$" is denoted "$\stackrel{*}{\Rightarrow}$". Then C' is a *superclass* of C (or C is a *subclass* of C') *iff* $C \stackrel{*}{\Rightarrow} C'$. If C has no proper subclass, that is, if $C' \not\Rightarrow C$ for any C', then C is a *leaf class*. The set of properties of C, denoted $Props^*(C)$, is the union of its immediate properties and the properties inherited from its superclasses. Note that C does not inherit any no-inherited property. $Props^*(C)$ is recursively defined as follows:

1. If C has no immediate superclass, then $Props^*(C) = Props(C)$.
2. Assume that $Props^*(C')$ is defined for every immediate superclass C' of C. Then $P \in Props^*(C)$ *iff* (1) $P \in Props(C)$ or (2) $P \notin NoInherit(C)$ and $P \in Props^*(C')$ for some immediate superclass C' of C.

That is,

$$Props^*(C) = \begin{cases} Props(C) & \text{if } C \text{ has no immediate superclass,} \\ Props(C) \cup \{ \bigcup_{C \Rightarrow C'} Props^*(C') - NoInherit(C) \} & \text{otherwise;} \end{cases}$$

The generalization taxonomy of $\mathbf{S}$ is assumed to be *acyclic*; that is, for all distinct classes $C_1, C_2 \in Classes(\mathbf{S})$, either $C_1 \stackrel{*}{\not\Rightarrow} C_2$ or $C_2 \stackrel{*}{\not\Rightarrow} C_1$.

For convention, we use C (with possible subscript) for a class in $Classes(\mathbf{S})$, and $\ldots, X, Y, Z$ (with possible subscripts) for subsets of $Classes(\mathbf{S})$. Furthermore, we write C for a singleton set $\{C\}$, if no confusion occurs. By $X\downarrow$, we mean the set of all subclasses of classes in X; that is, $X\downarrow = \{C \mid C \stackrel{*}{\Rightarrow} C' \text{ for some } C' \in X\}$. Then it is straightforward that

$$(X\downarrow \cap Y\downarrow)\downarrow = X\downarrow \cap Y\downarrow. \tag{2.1}$$

For a class C and its property P, all of the classes in $C\downarrow$ do not always inherit P from C. The *scope* of a property P with respect to a class C, denoted $C\downarrow_P$, is defined as the set of subclasses of C that can inherit P from C. That is, $C\downarrow_P$ is recursively defined as follows:

$$C\downarrow_P = \begin{cases} \{C\} & \text{if } C \text{ is a leaf class,} \\ \{C\} \cup \bigcup_{C' \Rightarrow C \wedge P \notin NoInherit(C')} C'\downarrow_P & \text{otherwise;} \end{cases}$$

Furthermore, for a set of classes X and a property P, We define $X{\downarrow_P} = \bigcup_{C \in X} C{\downarrow_P}$. Since $X \subseteq X{\downarrow_P} \subseteq X{\downarrow}$ by definition and $(X{\downarrow}){\downarrow} = X{\downarrow}$ by (2.1), it holds that $X{\downarrow} = (X{\downarrow}){\downarrow_P} = (X{\downarrow}){\downarrow}$; that is,

$$(X{\downarrow}){\downarrow_P} = X{\downarrow}. \tag{2.2}$$

A *database* over $\mathbf{S}$ is a (possibly infinite) labeled directed graph $G(V, A)$. The label for a vertex $u \in V$ is denoted $Cl(u)$, and corresponds to an actual class. The label for an arc in A corresponds to a property. G must also satisfy the following three conditions.

1. (*property value integrity*) If $u \xrightarrow{P} v \in A$, then $P \in Props^*(Cl(u))$.
2. (*property functionality*) If $u \xrightarrow{P} v, u \xrightarrow{P} w \in A$, then $v = w$.
3. (*property value completeness*) If $P \in Props^*(Cl(u))$, then there is an arc $u \xrightarrow{P} v \in A$.

The second and the third conditions mean that any property should be single-valued and total on its domain classes.

A *path function* pf is either `Id` (short for *identity*), or a sequence of properties separated by *dots*. Here, `Id` does not correspond to any property in $\mathbf{S}$. The identity path function is our means of referring to property value paths of zero length. Path composition and length are defined as follows:

$$pf_1 \circ pf_2 = \begin{cases} pf_1 & \text{if } pf_2 \text{ is } \texttt{Id}, \\ pf_2 & \text{if } pf_1 \text{ is } \texttt{Id}, \\ pf_1 \cdot pf_2 & \text{otherwise;} \end{cases}$$

$$len(pf) = \begin{cases} 0 & \text{if } pf = \texttt{Id}, \\ 1 + len(pf_1) & \text{if } pf = P \circ pf_1, \text{ where } P \text{ is a property.} \end{cases}$$

Note that the composition is associative; that is, $pf_1 \circ (pf_2 \circ pf_3) = (pf_1 \circ pf_2) \circ pf_3$. It is also straightforward that $len(pf_1 \circ pf_2) = len(pf_1) + len(pf_2)$. A path in a database $G(V, A)$ is *described* by a path function pf *iff* (1) the path is of zero length (that is, consists of a single vertex) and pf is `Id`, or (2) pf corresponds to the sequence of property labels on the path. By $u \rightsquigarrow_{pf} v \in G$, we mean that there is a path in G from a vertex u to a vertex v described by pf. Note that $u \rightsquigarrow_{\texttt{Id}} u \in G$ for any $u \in V$ and that $u \rightsquigarrow_P v \in G$ *iff* $u \xrightarrow{P} v \in A$, where P is a property.

An SC is used to assert a *type* condition on a path function, and has the form $X(pf : Y)$. A database $G(V, A)$ satisfies an SC $X(\texttt{Id} : Y)$ *iff* $Cl(u) \in X{\downarrow}$ implies $Cl(u) \in Y{\downarrow}$ for every vertex $u \in V$. Furthermore, G satisfies an SC $X(P \circ pf : Y)$ *iff* whenever there are two vertices $u, v \in V$ such that $u \rightsquigarrow_{P \circ pf} v$, then $Cl(u) \in X{\downarrow_P}$ implies $Cl(v) \in Y{\downarrow}$. The reason why

$Cl(u) \in X\downarrow_P$ is used instead of $Cl(u) \in X\downarrow$ is that for each class in X, the SC $X(P \circ pf : Y)$ asserts a kind of constraint on the property P with respect to the class, so that the constraint should be inherited to the scope of P with respect to the class. Here, if $X = \emptyset$, then $X\downarrow = \emptyset$ and $X\downarrow_P = \emptyset$ for any P, thereby the antecedents $Cl(u) \in X\downarrow$ and $Cl(u) \in X\downarrow_P$ cannot be satisfied; that is, any database must satisfy $X(pf : Y)$ trivially. Hence, if $X = \emptyset$, then $X(pf : Y)$ is said to be *trivial*. A database $G(V, A)$ over $\mathbf{S}$ *satisfies* Σ *iff* for any SC in Σ, $G(V, A)$ satisfies the SC. An SC σ is a *logical consequence* of a set of SCs Σ, written $\Sigma \models \sigma$, *iff* any database satisfying Σ must also satisfy σ.

A path function pf is *possible* with respect to a class C under a database schema $\langle \mathbf{S}, \Sigma \rangle$ *iff* there is a database $G(V, A)$ that satisfies Σ and that contains a path from a vertex u described by pf, where u satisfies the following condition.

$$\begin{cases} Cl(u) \in C\downarrow & \text{if } pf = \mathtt{Id}, \\ Cl(u) \in C\downarrow_P & \text{if } pf = P \circ pf'; \end{cases}$$

By $PossPaths(C)$, we denote the set of possible path functions with respect to C under $\langle \mathbf{S}, \Sigma \rangle$.

A database schema $\langle \mathbf{S}, \Sigma \rangle$ is *well-formed iff* for every C, there is a database $G(V, A)$ over $\mathbf{S}$ that satisfies Σ and that contains a vertex u such that

$$\begin{cases} Cl(u) = C & \text{if } C \text{ is actual;} \\ Cl(u) \in C\downarrow & \text{if } C \text{ is virtual.} \end{cases}$$

Through this paper, we assume that $len(pf) \leq 1$ for every $X(pf : Y) \in \Sigma$; that is, every SC in Σ has the form either $X(\mathtt{Id} : Y)$ or $X(P : Y)$. Note that the former and the latter can be applicable to represent an ISA relationship and a type constraint on a property, respectively. For example, an ISA relationship $C \Rightarrow C'$ is represented by an SC $C(\mathtt{Id} : C')$, and a type constraint on a property P of a class C is represented by an SC $C(P : C')$ for some class C'.

3 Testing Well-Formedness of a Database Schema

In this section, we will present a necessary and sufficient condition for $\langle \mathbf{S}, \Sigma \rangle$ to be well-formed. Intuitively, the condition states that every C must satisfy the following three conditions.

Condition 1: $\Sigma \not\models C(\mathtt{Id} : \emptyset)$.

Condition 2: If C is actual, then $\Sigma \not\models C(P : \emptyset)$ for any $P \in Props^*(C)$.

Condition 3: If C is actual and $\Sigma \models C(\mathtt{Id} : X)$ for some X, then $C \in X\downarrow$.

The following lemma gives necessity of the conditions.

SC1:	(reflexivity)	If $X \subseteq Y$, then $X(\texttt{Id} : Y)$.
SC2:	(cover)	If C is virtual, then $C(\texttt{Id} : C{\downarrow} - C)$.
SC3:	(transitivity)	If $X(pf_1 : Y)$ and $Y{\downarrow}(pf_2 : Z)$, then $X(pf_1 \circ pf_2 : Z)$.
SC4:	(inheritance)	If $X(\texttt{Id} : Y)$ and $C \in X{\downarrow}$, then $C(\texttt{Id} : Y)$.
SC5:	(selective inheritance)	If $X(P \circ pf : Y)$ and $C \in X{\downarrow}_P$, then $C(P \circ pf : Y)$.
SC6:	(attribution)	If $P \in \mathit{Props}^*(C)$, then $C(P : \mathit{Classes}(\mathbf{S}))$.
SC7:	(union)	If $X(pf : Z)$ and $Y(pf : Z)$, then $(X \cup Y)(pf : Z)$.
SC8:	(intersection)	If $X(pf : Y)$ and $X(pf : Z)$, then $X(pf : Y{\downarrow} \cap Z{\downarrow})$.
SC9:	(triviality)	If $X = \emptyset$, then $X(pf : Y)$ for any pf and any $Y \subseteq \mathit{Classes}(\mathbf{S})$.
SC10:	(generalization-1)	If $(C{\downarrow}_P - C)(P \circ pf : X)$ and $P \notin \mathit{Props}^*(C)$, then $C(P \circ pf : X)$.
SC11:	(generalization-2)	If $(C{\downarrow}_P - C)(P \circ pf : X)$ and C is virtual, then $C(P \circ pf : X)$.

Table 1: Axioms for SCs.

Lemma 1: If $\langle \mathbf{S}, \Sigma \rangle$ is well-formed, then every C satisfies Conditions 1 to 3.

Proof: Assume that $\langle \mathbf{S}, \Sigma \rangle$ is well-formed. Let us first consider Condition 2. Assume, contrary, that an actual class C does not satisfy Condition 2; that is, that $\Sigma \models C(P : \emptyset)$ for some actual class C and some $P \in \mathit{Props}^*(C)$. Since $\langle \mathbf{S}, \Sigma \rangle$ is well-formed, there is a database $G(V, A)$ over $\mathbf{S}$ that satisfies Σ and that contains a vertex u such that $Cl(u) = C$. Since $P \in \mathit{Props}^*(C)$, there is an arc $u \xrightarrow{P} v \in A$ by property value completeness for P. $Cl(u) = C$ implies $Cl(u) \in C{\downarrow}_P$. Then since $Cl(u) \in C{\downarrow}_P$ but $Cl(v) \notin \emptyset{\downarrow} = \emptyset$, G does not satisfy the SC $C(P : \emptyset)$. Hence $\Sigma \not\models C(P : \emptyset)$ by definition, which is a contradiction. That is, Condition 2 holds. The similar argument applies also to Condition 1.

Let us next consider Condition 3 and assume $\Sigma \models C(\texttt{Id} : X)$. We must show $C \in X{\downarrow}$. Since $\langle \mathbf{S}, \Sigma \rangle$ is well-formed and C is actual, there is a database $G(V, A)$ over $\mathbf{S}$ that satisfies Σ and that contains a vertex u such that $Cl(u) = C$. Since $\Sigma \models C(\texttt{Id} : X)$, G must also satisfy $C(\texttt{Id} : X)$. Thus $Cl(u) = C$ implies $Cl(u) \in C{\downarrow}$ implies $Cl(u) \in X{\downarrow}$. That is, Condition 3 holds. □

The condition above, however, is still a *semantical* characterization of a well-formed database schema, and is therefore not amenable to testing well-formedness of $\langle \mathbf{S}, \Sigma \rangle$. A related *syntactical* characterization will be based on the eleven axioms for SCs listed in Table 1. Let $\mathcal{A}$ denote the set of SC1 to SC11. If an SC σ is derivable from Σ using the axioms in $\mathcal{A}$, then we denote $\Sigma \vdash \sigma$. It will be shown in Section 4.1 that if $\langle \mathbf{S}, \Sigma \rangle$ is well-formed, then $\mathcal{A}$ is sound and complete; that is, $\Sigma \models X(pf : Y)$ *iff* $\Sigma \vdash X(pf : Y)$.

Lemma 2: The eleven axioms in Table 1 are sound; that is, $\Sigma \vdash \sigma$ implies $\Sigma \models \sigma$.

Proof: Let $G(V, A)$ be a database over $\mathbf{S}$ satisfying Σ and let $u \in V$. We will prove soundness for each axiom.

SC1: If $X \subseteq Y$, then $X\!\downarrow \subseteq Y\!\downarrow$, thereby $Cl(u) \in X\!\downarrow$ implies $Cl(u) \in Y\!\downarrow$. That is, if $X \subseteq Y$, then G satisfies $X(\mathtt{Id} : Y)$.

SC2: Assume that $Cl(u) \in C\!\downarrow$ for a virtual class C. Since $Cl(u)$ is actual by definition, C must be different from $Cl(u)$. Thus $Cl(u) \in C\!\downarrow - C$. Hence G satisfies $C(\mathtt{Id} : C\!\downarrow - C)$.

SC3: Assume that G satisfies both $X(pf_1 : Y)$ and $Y\!\downarrow(pf_2 : Z)$. There are two cases to be considered: $pf_1 \neq \mathtt{Id}$ and $pf_1 = \mathtt{Id}$. In the case of $pf_1 \neq \mathtt{Id}$, pf_1 is denoted $P \circ pf'$ for a property P and a path function pf'. Assume that $Cl(u) \in X\!\downarrow_P$ and that $u \rightsquigarrow_{pf_1 \circ pf_2} v \in G$ for some $v \in V$. Clearly, $u \rightsquigarrow_{pf_1 \circ pf_2} v \in G$ implies $u \rightsquigarrow_{pf_1} w \in G$ and $w \rightsquigarrow_{pf_2} v \in G$ for some $w \in V$. Then $Cl(u) \in X\!\downarrow_P$ implies $Cl(w) \in Y\!\downarrow$ by virtue of $X(pf_1 : Y)$. Since $Y\!\downarrow = (Y\!\downarrow)\!\downarrow$ by (2.1), $Cl(u) \in Y\!\downarrow$ implies $Cl(w) \in (Y\!\downarrow)\!\downarrow$, which implies $Cl(v) \in Z\!\downarrow$ by virtue of $Y\!\downarrow(pf_2 : Z)$. Thus G satisfies $X(pf_1 \circ pf_2 : Z)$. The proof in case of $pf_1 = \mathtt{Id}$ is similar to above.

SC4: Assume that G satisfies $X(\mathtt{Id} : Y)$ and that $C \in X\!\downarrow$. Further, assume that $Cl(u) \in C\!\downarrow$ for some $u \in V$. Then $Cl(u) \in C\!\downarrow$ and $C \in X\!\downarrow$ imply $Cl(u) \in X\!\downarrow$, which implies $Cl(u) \in Y\!\downarrow$ by virtue of $X(\mathtt{Id} : Y)$. Thus G satisfies $C(\mathtt{Id} : Y)$.

SC5: Assume that G satisfies $X(P \circ pf : Y)$ and that $C \in X\!\downarrow_P$. Further, assume that $Cl(u) \in C\!\downarrow_P$ and $u \rightsquigarrow_{P \circ pf} v \in G$ for some $v \in V$. $Cl(u) \in C\!\downarrow_P$ and $C \in X\!\downarrow_P$ imply $Cl(u) \in X\!\downarrow_P$, which implies $Cl(v) \in Y\!\downarrow$ by virtue of $X(P \circ pf : Y)$. Thus G satisfies $C(P \circ pf : Y)$.

SC6: Assume that $P \in \mathit{Props}^*(C)$, $Cl(u) \in C\!\downarrow_P$, and $u \rightsquigarrow_P v \in G$ (that is, u $\xrightarrow{P}$ $v \in A$) for some $v \in V$. Clearly, $Cl(v) \in \mathit{Classes}(\mathbf{S})$. Thus G satisfies $C(P : \mathit{Classes}(\mathbf{S}))$.

SC7: Assume that G satisfies $X(pf : Z)$ and $Y(pf : Z)$. There are two cases to be considered: $pf \neq \mathtt{Id}$ and $pf = \mathtt{Id}$. In the case of $pf \neq \mathtt{Id}$, pf is denoted $P \circ pf'$ for a property P and a path function pf'. Assume that $Cl(u) \in (X \cup Y)\!\downarrow_P$ and that $u \rightsquigarrow_{P \circ pf'} v \in G$. Since $(X \cup Y)\!\downarrow_P = X\!\downarrow_P \cup Y\!\downarrow_P$ by definition, $Cl(u) \in (X \cup Y)\!\downarrow_P$ implies either $Cl(u) \in X\!\downarrow_P$ or $Cl(u) \in Y\!\downarrow_P$. In either case, $Cl(v) \in Z\!\downarrow$ by virtue of $X(pf : Z)$ and $Y(pf : Z)$. Thus G satisfies $(X \cup Y)(pf : Z)$. The proof in case of $pf = \mathtt{Id}$ is similar to above.

SC8: Assume that G satisfies $X(pf : Y)$ and $X(pf : Z)$. Further, assume that (1) $Cl(u) \in X\!\downarrow$ if $pf = \mathtt{Id}$, $Cl(u) \in X\!\downarrow_P$ if $pf = P \circ pf'$, and (2) $u \rightsquigarrow_{pf} v \in G$. Then $Cl(v) \in Y\!\downarrow$ and $Cl(v) \in Z\!\downarrow$ by virtue of $X(pf : Z)$ and $Y(pf : Z)$, respectively. That is, $Cl(v) \in Y\!\downarrow \cap Z\!\downarrow$. Hence G satisfies $X(pf : Y\!\downarrow \cap Z\!\downarrow)$.

SC9: if $X = \emptyset$, then any database for $\mathbf{S}$ satisfies $X(pf : Y)$ *trivially* by definition.

SC10: Assume that G satisfies $(C{\downarrow_P} - C)(P \circ pf : X)$ and $P \notin \mathit{Props}^*(C)$. Further, assume that $\mathit{Cl}(u) \in C{\downarrow_P}$ and $u \rightsquigarrow_{P \circ pf} v \in G$. Since $u \rightsquigarrow_{P \circ pf} v \in G$, there is an arc $u \xrightarrow{P} w \in A$ for some $w \in V$. Since G is a database over $\mathbf{S}$, it must satisfy property value integrity for the arc; that is, $P \in \mathit{Props}^*(\mathit{Cl}(u))$. On the other hand, since $P \notin \mathit{Props}^*(C)$, $\mathit{Cl}(u)$ must be different from C. Thus $\mathit{Cl}(u) \in C{\downarrow_P}$ implies $\mathit{Cl}(u) \in C{\downarrow_P} - C$. Furthermore, $C{\downarrow_P} - C \subseteq (C{\downarrow_P} - C){\downarrow_P}$ by definition. Then $\mathit{Cl}(u) \in C{\downarrow_P} - C$ and $C{\downarrow_P} - C \subseteq (C{\downarrow_P} - C){\downarrow_P}$ imply $\mathit{Cl}(u) \in (C{\downarrow_P} - C){\downarrow_P}$. Since $u \rightsquigarrow_{P \circ pf} v \in G$ and G satisfies $(C{\downarrow_P} - C)(P \circ pf : X)$ by assumption, $\mathit{Cl}(u) \in (C{\downarrow_P} - C){\downarrow_P}$ implies $\mathit{Cl}(v) \in X{\downarrow}$. Consequently, G satisfies $C(P \circ pf : X)$.

SC11: Assume that G satisfies $(C{\downarrow_P} - C)(P \circ pf : X)$ and that C is virtual. Further, assume that $\mathit{Cl}(u) \in C{\downarrow_P}$ and $u \rightsquigarrow_{P \circ pf} v \in G$. Since C is virtual, $\mathit{Cl}(u)$ must be different from C. Then along the same line as proving SC10, we can show that G satisfies $C(P \circ pf : X)$. □

Let us consider how to obtain a *syntactical* characterization of a well-formed database schema. By axiom SC8 and (2.1), for a subset X of $\mathit{Classes}(\mathbf{S})$, there must exist a unique smallest subset S of $\mathit{Classes}(\mathbf{S})$ such that $S{\downarrow} = S$ and that $\Sigma \models X(pf : S)$. For a class C, if we have the smallest subset S of $\mathit{Classes}(\mathbf{S})$ such that $S{\downarrow} = S$ and that $\Sigma \models C(\mathtt{Id} : S)$, then we can examine that C satisfies Condition 1 by testing if $S \neq \emptyset$. If C is actual, then Condition 3 can also be examined by testing if $C \in S$. Similarly, for $P \in \mathit{Props}^*(C)$, if we have the smallest subset S' of $\mathit{Classes}(\mathbf{S})$ such that $S'{\downarrow} = S'$ and that $\Sigma \models C(P : S')$, then we can examine that C satisfies Condition 2 by testing if $S' \neq \emptyset$. The intuitive idea is to obtain such S and S' for each C and each $P \in \mathit{Props}^*(C)$. For each C and each $P \in \mathit{Props}^*(C)$, let

$$\Sigma_{\langle C,\mathtt{Id}\rangle} = \{X(\mathtt{Id} : Y) \in \Sigma \mid C \in X{\downarrow}\} \tag{3.1}$$

$$\Sigma_{\langle C,P\rangle} = \{X(P : Y) \in \Sigma \mid C \in X{\downarrow_P}\}. \tag{3.2}$$

Furthermore, let

$$S_{\langle C,\mathtt{Id}\rangle} = \begin{cases} C{\downarrow} \cap (\bigcap_{X(\mathtt{Id}:Y) \in \Sigma_{\langle C,\mathtt{Id}\rangle}} Y{\downarrow}) & \text{if } C \text{ is actual,} \quad (3.3) \\ (C{\downarrow} - C){\downarrow} \cap (\bigcap_{X(\mathtt{Id}:Y) \in \Sigma_{\langle C,\mathtt{Id}\rangle}} Y{\downarrow}) & \text{if } C \text{ is virtual;} \quad (3.4) \end{cases}$$

$$S_{\langle C,P\rangle} = \mathit{Classes}(\mathbf{S}) \cap (\bigcap_{X(P:Y) \in \Sigma_{\langle C,P\rangle}} Y{\downarrow}). \tag{3.5}$$

By (2.1) and the fact that $\mathit{Classes}(\mathbf{S}){\downarrow} = \mathit{Classes}(\mathbf{S})$, we have:

$$\begin{cases} S_{\langle C,\mathtt{Id}\rangle}{\downarrow} &= S_{\langle C,\mathtt{Id}\rangle} \\ S_{\langle C,P\rangle}{\downarrow} &= S_{\langle C,P\rangle} \end{cases} \tag{3.6}$$

By (2.2) and (3.6), we have:

$$\begin{cases} S_{\langle C,\mathtt{Id}\rangle}{\downarrow_{P'}} &= S_{\langle C,\mathtt{Id}\rangle} \\ S_{\langle C,P\rangle}{\downarrow_{P'}} &= S_{\langle C,P\rangle} \end{cases} \quad \text{for any property } P' \tag{3.7}$$

Furthermore, the following lemma holds.

Lemma 3: $\Sigma \vdash C(\mathtt{Id} : S_{\langle C,\mathtt{Id}\rangle})$ and $\Sigma \vdash C(P : S_{\langle C,P\rangle})$.

Proof: We claim in advance that if $X(\mathtt{Id} : Y) \in \Sigma$ and $C \in X{\downarrow}$, then $\Sigma \vdash C(\mathtt{Id} : Y{\downarrow})$.

Assume that $X(\mathtt{Id} : Y) \in \Sigma$ and $C \in X{\downarrow}$. By axiom SC4, $X(\mathtt{Id} : Y)$ and $C \in X{\downarrow}$ imply $C(\mathtt{Id} : Y)$. By axiom SC8, $C(\mathtt{Id} : Y)$ and $C(\mathtt{Id} : Y)$ imply $C(\mathtt{Id} : Y{\downarrow} \cap Y{\downarrow})$, where $Y{\downarrow} \cap Y{\downarrow} = Y{\downarrow}$. Hence the claim follows.

Let us now prove $\Sigma \vdash C(\mathtt{Id} : S_{\langle C,\mathtt{Id}\rangle})$. From the claim above, it follows that $\Sigma \vdash C(\mathtt{Id} : Y{\downarrow})$ for every $X(\mathtt{Id} : Y) \in \Sigma_{\langle C,\mathtt{Id}\rangle}$. If C is virtual, then $\Sigma \vdash C(\mathtt{Id} : C{\downarrow} - C)$ by axiom SC2. That is, we have $C(\mathtt{Id} : C{\downarrow} - C)$ and $C(\mathtt{Id} : Y{\downarrow})$ for every $X(\mathtt{Id} : Y) \in \Sigma_{\langle C,\mathtt{Id}\rangle}$, thereby $\Sigma \vdash C(\mathtt{Id} : S_{\langle C,\mathtt{Id}\rangle})$ by axiom SC8. On the other hand, if C is actual, then $\Sigma \vdash C(\mathtt{Id} : C{\downarrow})$ by axiom SC1, since $C \in C{\downarrow}$ by definition. Hence by axiom SC8, $\Sigma \vdash C(\mathtt{Id} : S_{\langle C,\mathtt{Id}\rangle})$. Along the same line as proving $\Sigma \vdash C(\mathtt{Id} : S_{\langle C,\mathtt{Id}\rangle})$, we can show $\Sigma \vdash C(P : S_{\langle C,P\rangle})$. □

An actual class C is said to be *lowest* with a property P if (1) $P \in \mathit{Props}^*(C)$ and (2) there is no actual class C' such that $C' \in C{\downarrow_P} - C$. We are now ready to present a necessary and sufficient condition for **S** to be well-formed, as follows:

WF1: Every leaf class is actual.

WF2: $S_{\langle C,P\rangle} \neq \emptyset$ for every lowest actual class C with every property $P \in \mathit{Props}^*(C)$.

WF3: $C \in S_{\langle C,\mathtt{Id}\rangle}$ for every actual class C.

Proof of Necessity.

Note that $\Sigma \models C(\mathtt{Id} : S_{\langle C,\mathtt{Id}\rangle})$ and $\Sigma \models C(P : S_{\langle C,P\rangle})$ for every C and every $P \in \mathit{Props}^*(C)$ by Lemmas 2 and 3. Assume that condition WF1 does not hold; that is, a leaf class C is virtual. Since C is a leaf class, $C' \not\Rightarrow C$ for any C'; that is, $C{\downarrow} - C = \emptyset$. Thus by (3.4), it must follow that $S_{\langle C,\mathtt{Id}\rangle} = \emptyset$. That is, $\Sigma \models C(\mathtt{Id} : \emptyset)$, thereby C does not satisfy Condition 1. Assume that condition WF2 does not hold; that is, $S_{\langle C,P\rangle} = \emptyset$ for a lowest actual class C with a property P. Then $\Sigma \models C(P : \emptyset)$, that is C does not satisfy Condition 2. Assume that condition WF3 does not hold; that is, $C \notin S_{\langle C,\mathtt{Id}\rangle}$ for an actual class C. Then C does not satisfy Condition 3. Consequently, if one of conditions WF1 to WF3 does not hold, then $\langle \mathbf{S}, \Sigma \rangle$ is not well-formed by Lemma 1.

Proof of Sufficiency.

Assume that $\langle \mathbf{S}, \Sigma \rangle$ satisfies conditions WF1 to WF3. We will construct a database $G(V, A)$ over $\mathbf{S}$ satisfying (1) Σ and that (2) for every actual class C, there is a vertex $u \in V$ such that $Cl(u) = C$. Then $\langle \mathbf{S}, \Sigma \rangle$ is well-formed, as explained below:

Let C be a class. It suffices to show that there is a vertex $u \in V$ such that $Cl(u) = C$ if C is actual, and that $Cl(u) \in C{\downarrow}$ if C is virtual. By construction, if C is actual, then there is a vertex $u \in V$ such that $Cl(u) = C$. Assume that C is virtual. Clearly, there is a leaf class C' such that $C' \stackrel{*}{\Rightarrow} C$. Since C' is a leaf class, C' is actual by condition WF1. By construction, there is a vertex $v \in V$ such that $Cl(v) = C'$. Hence $Cl(v) = C'$ and $C' \stackrel{*}{\Rightarrow} C$ imply $Cl(v) \in C{\downarrow}$.

In order to construct such a database $G(V, A)$, we will use the following lemma.

Lemma 4: Assume that $\langle \mathbf{S}, \Sigma \rangle$ satisfies WF1 to WF3. For every actual class C and every $P \in Props^*(C)$, $S_{\langle C,P \rangle}$ contains an actual class.

Proof: We first claim $S_{\langle C,P \rangle} \neq \emptyset$. If C is a lowest actual class with a property P, then the claim follows from condition WF2. Assume that C is actual and $P \in Props^*(C)$ but C is not lowest with a property P. Then there is a lowest actual class C' with a property P such that $C' \in C{\downarrow}_P$. Since $P \in Props^*(C)$ and $C' \in C{\downarrow}_P$ imply $P \in Props^*(C')$, $S_{\langle C',P \rangle}$ should be defined. Since C' is a lowest actual class with a property P, $S_{\langle C',P \rangle} \neq \emptyset$ by condition WF2. By $C' \in C{\downarrow}_P$, for any subset X of $Classes(\mathbf{S})$, $C \in X{\downarrow}_P$ implies $C' \in X{\downarrow}_P$. Thus $\Sigma_{\langle C,P \rangle} \subseteq \Sigma_{\langle C',P \rangle}$ by (3.2), which implies $S_{\langle C,P \rangle} \supseteq S_{\langle C',P \rangle}$ by definition. Hence $S_{\langle C',P \rangle} \neq \emptyset$ and $S_{\langle C,P \rangle} \supseteq S_{\langle C',P \rangle}$ imply $S_{\langle C,P \rangle} \neq \emptyset$. The claim follows.

Since $S_{\langle C,P \rangle}{\downarrow} = S_{\langle C,P \rangle}$ by (3.6), the claim implies that $S_{\langle C,P \rangle}$ must contain a leaf class. The class is actual by condition WF1. Hence Lemma 4 holds. □

Now $G(V, A)$ is defined as follows: For each actual class C, V contains a vertex u such that $Cl(u) = C$. For each $u \in V$ and each $P \in Props^*(Cl(u))$, A contains an arc $u \stackrel{P}{\longrightarrow} v$ such that $Cl(v) \in S_{\langle Cl(u),P \rangle}$. Note that A is well-defined, since $S_{\langle Cl(u),P \rangle}$ contains an actual class by Lemma 4.

In order to prove sufficiency, we must show that G is a database over $\mathbf{S}$ satisfying Σ. Since $u \stackrel{P}{\longrightarrow} v \in A$ implies $P \in Props^*(Cl(u))$ by construction, G satisfies property value integrity. Since by construction, A contains a single arc $u \stackrel{P}{\longrightarrow} v$ for every $u \in V$ and every $P \in Props^*(Cl(u))$, G satisfies both property functionality and property value completeness for every property. Hence G is a database over $\mathbf{S}$. It remains to show that G satisfies Σ. This follows from the following lemma.

Lemma 5: Assume that $\langle \mathbf{S}, \Sigma \rangle$ satisfies conditions WF1 to WF3. A database $G(V, A)$ over $\mathbf{S}$ satisfies Σ *iff* $Cl(v) \in S_{\langle Cl(u),P \rangle}$ for every $u \xrightarrow{P} v \in A$.

Proof: *If part.* Assume that $Cl(v) \in S_{\langle Cl(u),P \rangle}$ for every $u \xrightarrow{P} v \in A$. Let $X(pf : Y) \in \Sigma$. There are two cases to be considered: $pf = \mathtt{Id}$ and $pf = P$ for a property P.

Case 1: where $pf = \mathtt{Id}$. Assume that $Cl(u) \in X\!\downarrow$ for a vertex $u \in V$. It suffices to show that $Cl(u) \in Y\!\downarrow$. Since G is a database over $\mathbf{S}$, $Cl(u)$ is actual by definition. Thus $Cl(u) \in S_{\langle Cl(u),\mathtt{Id} \rangle}$ by condition WF3. Furthermore, since $Cl(u) \in X\!\downarrow$ implies $X(\mathtt{Id} : Y) \in \Sigma_{\langle Cl(u),\mathtt{Id} \rangle}$ by (3.1), it follows from (3.3) that $S_{\langle Cl(u),\mathtt{Id} \rangle} \subseteq Y\!\downarrow$. Hence $Cl(u) \in Y\!\downarrow$.

Case 2: where $pf = P$ *for a property* P. Assume that $Cl(u) \in X\!\downarrow_P$ and $u \rightsquigarrow_{pf} v \in G$ for vertices $u, v \in G$. It suffices to show that $Cl(v) \in Y\!\downarrow$. Then $u \rightsquigarrow_{pf} v \in G$ implies $u \xrightarrow{P} v \in A$ by definition. Note that $P \in \mathit{Props}^*(Cl(u))$ by property value integrity for G. Since $Cl(u) \in X\!\downarrow_P$, we can show that $S_{\langle Cl(u),P \rangle} \subseteq Y\!\downarrow$ along the same line as in Case 1. Furthermore, since $Cl(v) \in S_{\langle Cl(u),P \rangle}$ by assumption of G, it holds that $Cl(v) \in Y\!\downarrow$.

Only if part. Assume that $Cl(v) \notin S_{\langle Cl(u),P \rangle}$ for some $u \xrightarrow{P} v \in A$. Note that $P \in \mathit{Props}^*(Cl(u))$ by property value integrity for G. Thus by (3.2) and (3.5), $\Sigma_{\langle Cl(u),P \rangle}$ must contain an SC $X(P : Y)$ such that $Cl(u) \in X\!\downarrow_P$ and $Cl(v) \notin Y\!\downarrow$. Furthermore, $u \xrightarrow{P} v \in A$ implies $u \rightsquigarrow_P v \in G$ by definition. Hence, G does not satisfy $X(P : Y) \in \Sigma_{\langle Cl(u),P \rangle} \subseteq \Sigma$. □

Consequently, we have the following theorem.

Theorem 1: $\langle \mathbf{S}, \Sigma \rangle$ is well-formed *iff* it satisfies conditions WF1 to WF3. □

Let us now consider the running time for testing conditions WF1 to WF3. To test the conditions, we need a number of set operations for subsets of $\mathit{Classes}(\mathbf{S})$. Let K denote the number of classes in $\mathit{Classes}(\mathbf{S})$. To represent a subset of $\mathit{Classes}(\mathbf{S})$, we use a bit array of size K. Then for a class C and a subset S of $\mathit{Classes}(\mathbf{S})$, membership "$C \in S$" can be tested in constant time. For subsets S and T of $\mathit{Classes}(\mathbf{S})$, set containment "$S \subseteq T$" can be tested in $O(|S|)$ time, where $|S|$ denotes the cardinality of S. Furthermore, union "$S \cup T$", intersection "$S \cap T$", and set difference "$S - T$" can also be computed in $O(|S| + |T|)$ time. Note that $O(|S| + |T|) \leq O(K)$.

The generalization taxonomy of $\mathbf{S}$ can be represented by a directed graph $G_{\mathtt{S}}(V_{\mathtt{S}}, A_{\mathtt{S}})$, called a *GT-graph* of $\mathbf{S}$, such that $V_{\mathtt{S}} = \mathit{Classes}(\mathbf{S})$ and $A_{\mathtt{S}} = \{C \rightarrow C' \mid C \Rightarrow C' \text{ for classes } C, C' \in \mathit{Classes}(\mathbf{S})\}$. By assumption, $G_{\mathtt{S}}$ should be acyclic. Let M denote the size of $A_{\mathtt{S}}$; that is, the number of edges of $G_{\mathtt{S}}$. Then $G_{\mathtt{S}}$ can be constructed in $O(M)$ time. Furthermore, we define another kind of directed graph $G^P_{\mathtt{S}}(V^P_{\mathtt{S}}, A^P_{\mathtt{S}})$, called *GT-graph with*

respect to P of $\mathbf{S}$, such that $V_{\mathtt{S}}^P = \mathit{Classes}(\mathbf{S})$ and $A_{\mathtt{S}}^P = \{C \to C' \mid C \Rightarrow C'$ for classes $C, C' \in \mathit{Classes}(\mathbf{S})$ and $P \notin \mathit{NoInherit}(C)\}$. Let L denote the number of properties in $\mathbf{S}$. To represent $\mathit{NoInherit}(C)$ for each class by bit arrays, we need $O(K \cdot L)$ time. Then membership "$P \in \mathit{NoInherit}(C)$" can be tested in constant time. Thus $G_{\mathtt{S}}^P$ can be constructed in $O(M)$ time. Then $G_{\mathtt{S}}^P$ for all P can be constructed in $O(L \cdot M)$ time.

To test condition WF1, we must compute the set of leaf classes. Since the set is proportional to the size of $G_{\mathtt{S}}$ which can be constructed in $O(M)$ time, we can compute the set of leaf classes in that time. Thus condition WF1 can be tested in $O(M)$ time.

To test condition WF2, we must compute $\Sigma_{\langle C,P\rangle}$ and $S_{\langle C,P\rangle}$ for each class C and each $P \in \mathit{Props}^*(C)$. Here, to get $\mathit{Props}^*(C)$, we must compute set operation "$\bigcup_{C \Rightarrow C'} \mathit{Props}^*(C')$", in which each union can be computed in $O(L)$ time. Then since the total number of such union operations is the number of ISA relationship "$\Rightarrow$" of $\mathbf{S}$ (which is equivalent to M), $\mathit{Props}^*(C)$ for all C can be computed in $O(L \cdot M)$ time. To compute $\Sigma_{\langle C,P\rangle}$ and $S_{\langle C,P\rangle}$, we must compute $Z{\downarrow_P}$ for a subset Z of $\mathit{Classes}(\mathbf{S})$. Let $\overline{G}_{\mathtt{S}}^P$ be the graph that is constructed from $G_{\mathtt{S}}^P$ by reversing the direction of every arc in $A_{\mathtt{S}}^P$. Since $G_{\mathtt{S}}^P$ for all property P can be constructed in $O(L \cdot M)$ time, $\overline{G}_{\mathtt{S}}^P$ for all property P can also be constructed in that time. Then $Z{\downarrow_P}$ coincides with the set of vertices reachable from a vertex in Z in $\overline{G}_{\mathtt{S}}^P$. Thus $Z{\downarrow_P}$ can be computed in $O(M)$ time, which implies that $X{\downarrow_P}$ for all $X(P:Y) \in \Sigma$ can be computed in $O(M \cdot |\Sigma|)$ time. Similarly, using $\overline{G}_{\mathtt{S}}$ graph that is constructed from $G_{\mathtt{S}}$ by reversing the direction of every arc in $A_{\mathtt{S}}$, $Y{\downarrow}$ for all $X(P:Y) \in \Sigma$ can be computed in $O(M \cdot |\Sigma|)$ time. To represent such $X{\downarrow_P}$ and $Y{\downarrow}$ by bit arrays, we need $O(K \cdot |\Sigma|)$ time. Then set containment "$C \in X{\downarrow_P}$" can be tested in constant time, thereby it can also be decided in constant time whether or not an SC is in $\Sigma_{\langle C,P\rangle}$. Furthermore, the number of intersections for computing $S_{\langle C,P\rangle}$ is $O(|\Sigma|)$, and each intersection requires $O(K)$ time. Thus $S_{\langle C,P\rangle}$ can be computed in $O(K \cdot |\Sigma|)$ time. Then the time for computing $S_{\langle C,P\rangle}$ for all C and all $P \in \mathit{Props}^*(C)$ is $O(K \cdot |\Sigma| \times K \cdot L) = O(K^2 \cdot L \cdot |\Sigma|)$. To represent them by bit arrays, we need $O(K^2 \cdot L)$ time. After that, condition WF2 can also be tested in that time. Consequently, to test condition WF2, we need $O(L \cdot M + M \cdot |\Sigma| + K^2 \cdot L \cdot |\Sigma|)$ time. Note that $M \leq K^2$. Hence $K^2 \cdot L \cdot |\Sigma|$ is the dominant factor; that is, $O(L \cdot M + M \cdot |\Sigma| + K^2 \cdot L \cdot |\Sigma|) = O(K^2 \cdot L \cdot |\Sigma|)$.

As for condition WF3, noting that $C{\downarrow}$ (or $(C{\downarrow} - C)$) can be computed in $O(M)$ time, we can compute $S_{\langle C,\mathtt{Id}\rangle}$ in $O(K \cdot |\Sigma| + M)$ time by a similar way as computing $S_{\langle C,P\rangle}$. Thus the time for computing $S_{\langle C,\mathtt{Id}\rangle}$ for all C is $O(K^2 \cdot |\Sigma| + K \cdot M)$. Condition WF3 can also be tested in that time. We have the following theorem.

Theorem 2: For a database schema $\langle \mathbf{S}, \Sigma \rangle$, it can be decided in $O(K \cdot M + K^2 \cdot L \cdot |\Sigma|)$ time whether or not $\langle \mathbf{S}, \Sigma \rangle$ is well-formed, where K, L, and M denote the number of classes, properties, and the total number of classes declared as immediate superclasses for each class in $\mathbf{S}$, respectively. □

4 Implication Problems for Specialization Constraints

4.1 Complete axiomatization

In this section, we will prove that if $\langle \mathbf{S}, \Sigma \rangle$ is well-formed, then $\mathcal{A}$ is complete for deciding whether or not $\Sigma \models \sigma$; that is, $\Sigma \models \sigma$ implies $\Sigma \vdash \sigma$. For a path function pf $(= P_n.P_{n-1}.\cdots.P_1)$, a pf-list over $\mathbf{S}$ is a directed graph of the form

$$v_n \xrightarrow{P_n} v_{n-1} \xrightarrow{P_{n-1}} \cdots \xrightarrow{P_1} v_0,$$

which satisfies the following two conditions:

1. $Cl(v_i) \in Classes(\mathbf{S})$, where $0 \leq i \leq n$.
2. $P_i \in Props^*(Cl(v_i))$, where $1 \leq i \leq n$.

In particular, if $pf = \mathtt{Id}$, then the pf-list consists of a single vertex v_0 with no arc. We give some lemmas on a pf-list, which are necessary to show the completeness. The following lemma holds by the similar argument to Lemma 5.

Lemma 6: Assume that $\langle \mathbf{S}, \Sigma \rangle$ is well-formed. A pf-list $v_n \xrightarrow{P_n} v_{n-1} \xrightarrow{P_{n-1}} \cdots \xrightarrow{P_1} v_0$ over $\mathbf{S}$ satisfies Σ *iff* $Cl(v_{i-1}) \in S_{\langle Cl(v_i),P \rangle}$ for every $v_i \xrightarrow{P_i} v_{i-1}$, where $1 \leq i \leq n$. □

Lemma 7: Assume that $\langle \mathbf{S}, \Sigma \rangle$ is well-formed. If there is a pf-list $v_n \xrightarrow{P_n} v_{n-1} \xrightarrow{P_{n-1}} \cdots \xrightarrow{P_1} v_0$ over $\mathbf{S}$ satisfying Σ, then there is a database $G(V, A)$ over $\mathbf{S}$ that (1) satisfies Σ and (2) contains the pf-list as a subgraph.

Proof: Let $G_0(V_0, A_0)$ denote the pf-list. Then $G(V, A)$ is defined as follows: V is constructed from V_0 by adding, for each actual class C, a vertex u such that $Cl(u) = C$. A is constructed from A_0 by adding, for each $u \in V$ and each $P \in Props^*(Cl(u))$, an arc $u \xrightarrow{P} v$ such that $Cl(v) \in S_{\langle Cl(u),P \rangle}$, unless $u = v_i$ and $P = P_i$ for some i with $1 \leq i \leq n$.

Since $\langle \mathbf{S}, \Sigma \rangle$ is well-formed, it satisfies conditions WF1 to WF3 by Theorem 1. Thus by Lemma 4, for each $u \in V$ and each $P \in Props^*(Cl(u))$, $S_{\langle Cl(u),P \rangle}$ contains an actual class. Hence A is well-defined. Clearly, G contains the pf-list as a subgraph. It remains to show that G is a database over $\mathbf{S}$ and satisfies Σ.

Let us first prove that G is a database over $\mathbf{S}$. By definition, G_0 satisfies property value integrity; that is, $P \in Props^*(Cl(u))$ for every $u \xrightarrow{P} v \in A_0$. Furthermore, by construction, $P \in Props^*(Cl(u))$ for each newly added arc $u \xrightarrow{P} v \in A$. That is, G satisfies property value integrity. Since by construction, A contains a single arc $u \xrightarrow{P} v$ for every $u \in V$ and every $P \in Props^*(Cl(u))$, G satisfies both property functionality and property value completeness for every property. Thereby G is a database over $\mathbf{S}$.

Let us finally prove that G satisfies Σ. Since (1) by construction, $Cl(v) \in S_{\langle Cl(u),P\rangle}$ for every $u \xrightarrow{P} v \in A$ and (2) $\langle \mathbf{S}, \Sigma\rangle$ satisfies conditions WF1 to WF3, it follows from Lemma 5 that G satisfies Σ. Thus G satisfies Σ. □

Lemma 8: Assume that $\langle \mathbf{S}, \Sigma\rangle$ is well-formed. If there is a pf-list $v_n \xrightarrow{P_n} v_{n-1} \xrightarrow{P_{n-1}} \cdots \xrightarrow{P_1} v_0$ over $\mathbf{S}$ that satisfies Σ but not $X(pf : Y)$, then $\Sigma \not\models X(pf : Y)$.

Proof: Assume that there is a pf-list $v_n \xrightarrow{P_n} v_{n-1} \xrightarrow{P_{n-1}} \cdots \xrightarrow{P_1} v_0$ over $\mathbf{S}$ that satisfies Σ but not $X(pf : Y)$. By Lemma 7, there is a database $G(V, A)$ over $\mathbf{S}$ that satisfies Σ and contains the pf-list as a subgraph. Since the pf-list violates $X(pf : Y)$ and is contained in G, the whole graph G also violates the SC. Hence $\Sigma \not\models X(pf : Y)$ by definition. □

Since $\Sigma \vdash \sigma$ implies $\Sigma \models \sigma$ by Lemma 2, the proof of completeness of $\mathcal{A}$ will be done by showing that if $\Sigma \not\vdash X(pf : Y)$, then there is a pf-list over $\mathbf{S}$ that satisfies Σ but not $X(pf : Y)$ (which implies $\Sigma \not\models X(pf : Y)$ by Lemma 8). In order to construct such a pf-list, we introduce the concept of *SC base*.

Let $pf = P_n.P_{n-1}.\cdots.P_1$. By pf_i, we mean $P_i.P_{i-1}.\cdots.P_1$, where $0 \leq i \leq n$. In particular, $pf_n = pf$ and $pf_0 = \mathtt{Id}$. The *SC base* of a subset Y of $Classes(\mathbf{S})$ and a path function pf under $\langle \mathbf{S}, \Sigma\rangle$, written $SCbase(Y, pf)$, is a sequence $\langle Z_n, Z_{n-1}, \cdots, Z_0\rangle$ such that $Z_i = \{C \mid \Sigma \vdash C(pf_i : Y)\}$, where $0 \leq i \leq n$. Using the *SC base*, we proceed the proof of completeness as follows:

Step 1: Show that $\Sigma \vdash X(pf_i : Y)$ *iff* $X \subseteq Z_i$ (Lemma 9).

Step 2: Give necessary conditions for a sequence $\langle Z_n, Z_{n-1}, \cdots, Z_0\rangle$ to coincide with $SCbase(Y, pf)$ (Lemma 10).

Step 3: Show that if a sequence $\langle Z_n, Z_{n-1}, \cdots, Z_0\rangle$ satisfies the above necessary conditions, then $X \not\subseteq Z_i$ implies that there is a pf_i-list over $\mathbf{S}$ that satisfies Σ but not $X(pf_i : Y)$ (Lemma 11).

Step 4: By Steps 1 to 3, it holds that $\Sigma \not\vdash X(pf : Y)$ implies $\Sigma \not\models X(pf : Y)$ (Theorem 3).

Lemma 9: Let $SCbase(Y, pf) = \langle Z_n, Z_{n-1}, \cdots, Z_0\rangle$. Then $\Sigma \vdash X(pf_i : Y)$ *iff* $X \subseteq Z_i$.

Proof: *If part.* Assume $X \subseteq Z_i$. If $X = \emptyset$, then $\Sigma \vdash X(pf_i : Y)$ by axiom SC9. If $X \neq \emptyset$, then $\{C(pf_i : Y) \mid C \in X\}$ implies $X(pf_i : Y)$ by axiom SC7. Note that $\Sigma \vdash C(pf_i : Y)$ for every $C \in X \subseteq Z_i$ by definition. Hence $\Sigma \vdash X(pf_i : Y)$.

Only if part. Assume $\Sigma \vdash X(pf_i : Y)$. We must show $X \subseteq Z_i$. There are two cases to be considered: $i = 0$ (that is, $pf_i = \mathtt{Id}$) and $1 \leq i \leq n$. In the case of $i = 0$, since $X \subseteq X{\downarrow}$ by definition, for each $C \in X{\downarrow}$, $X(\mathtt{Id} : Y)$ implies $C(\mathtt{Id} : Y)$ by axiom SC4. That is, $\Sigma \vdash C(\mathtt{Id} : Y)$. Thus $C \in Z_0$ by definition. Hence $X \subseteq Z_0$. In the case of $1 \leq i \leq n$, we can show that $X \subseteq Z_i$ along the same line as proving above. □

Lemma 10: $SCbase(Y, pf)(= \langle Z_n, Z_{n-1}, \cdots, Z_0 \rangle)$ satisfies the following five conditions.

SCB1: $Y{\downarrow} \subseteq Z_0$.

SCB2: For a class C, if $S_{\langle C,\mathtt{Id}\rangle} \subseteq Z_i$, then $C \in Z_i$, where $0 \leq i \leq n$.

SCB3: For a class C, if $P_i \in Props^*(C)$ and $S_{\langle C,P_i\rangle} \subseteq Z_{i-1}$, then $C \in Z_i$, where $1 \leq i \leq n$.

SCB4: For a class C, if $C{\downarrow_{P_i}} - C \subseteq Z_i$ and $P_i \notin Props^*(C)$, then $C \in Z_i$, where $1 \leq i \leq n$.

SCB5: For a class C, if $C{\downarrow_{P_i}} - C \subseteq Z_i$ and C is virtual, then $C \in Z_i$, where $1 \leq i \leq n$.

Proof: *SCB1:* Let $C \in Y{\downarrow}$. $Y \subseteq Y$ implies $Y(\mathtt{Id} : Y)$ by axiom SC1. Then by axiom SC4, $Y(\mathtt{Id} : Y)$ and $C \in Y{\downarrow}$ imply $C(\mathtt{Id} : Y)$. That is, $\Sigma \vdash C(\mathtt{Id} : Y)$. Thus $C \in Z_0$ by definition. Hence $Y{\downarrow} \subseteq Z_0$.

SCB2: Assume $S_{\langle C,\mathtt{Id}\rangle} \subseteq Z_i$. Then $\Sigma \vdash S_{\langle C,\mathtt{Id}\rangle}(pf_i : Y)$ by Lemma 9. Since $S_{\langle C,\mathtt{Id}\rangle}{\downarrow} = S_{\langle C,\mathtt{Id}\rangle}$ by (3.6), $S_{\langle C,\mathtt{Id}\rangle}(pf_i : Y)$ is equivalent to $S_{\langle C,\mathtt{Id}\rangle}{\downarrow}(pf_i : Y)$. Furthermore, $\Sigma \vdash C(\mathtt{Id} : S_{\langle C,\mathtt{Id}\rangle})$ by Lemma 3. Thus $C(\mathtt{Id} : S_{\langle C,\mathtt{Id}\rangle})$ and $S_{\langle C,\mathtt{Id}\rangle}{\downarrow}(pf_i : Y)$ imply $C(\mathtt{Id} \circ pf_i : Y)$ by axiom SC3, where $\mathtt{Id} \circ pf_i = pf_i$. That is, $\Sigma \vdash C(pf_i : Y)$. Hence $C \in Z_i$ by definition.

SCB3: Assume $P_i \in Props^*(C)$ and $S_{\langle C,P_i\rangle} \subseteq Z_{i-1}$. Then the latter implies $\Sigma \vdash S_{\langle C,P_i\rangle}(pf_{i-1} : Y)$ by Lemma 9. Since $S_{\langle C,P_i\rangle}{\downarrow} = S_{\langle C,P_i\rangle}$ by (3.6), $S_{\langle C,P_i\rangle}(pf_{i-1} : Y)$ is equivalent to $S_{\langle C,P_i\rangle}{\downarrow}(pf_{i-1} : Y)$. Furthermore, $\Sigma \vdash C(P_i : S_{\langle C,P_i\rangle})$ by Lemma 3. Thus $C(P_i : S_{\langle C,P_i\rangle})$ and $S_{\langle C,P_i\rangle}{\downarrow}(pf_{i-1} : Y)$ imply $C(P_i \circ pf_{i-1} : Y)$ by axiom SC3, where $P_i \circ pf_{i-1} = pf_i$. That is, $\Sigma \vdash C(pf_i : Y)$. Hence $C \in Z_i$ by definition.

SCB4: Assume $C{\downarrow_{P_i}} - C \subseteq Z_i$ and $P_i \notin Props^*(C)$. Then the former implies $\Sigma \vdash (C{\downarrow_{P_i}} - C)(pf_i : Y)$ by Lemma 9. By axiom SC10, $(C{\downarrow_{P_i}} - C)(pf_i : Y)$ and $P_i \notin Props^*(C)$ imply $C(pf_i : Y)$. That is, $\Sigma \vdash C(pf_i : Y)$. Hence $C \in Z_i$ by definition.

SCB5: Assume that $C{\downarrow_{P_i}} - C \subseteq Z_i$ and C is virtual. Then the former implies $\Sigma \vdash (C{\downarrow_{P_i}} - C)(pf_i : Y)$ by Lemma 9. $(C{\downarrow_{P_i}} - C)(pf_i : Y)$ and the fact that C is virtual imply $\Sigma \vdash C(pf_i : Y)$ by axiom SC11. Hence $C \in Z_i$ by definition. □

Lemma 11: Let $\langle Z_n, Z_{n-1}, \cdots, Z_0 \rangle$ denote a sequence satisfying conditions SCB1 to SCB5, which does not necessarily coincide with $SCbase(Y, pf)$. If $\langle \mathbf{S}, \Sigma \rangle$ is well-formed, then $X \not\subseteq Z_i$ implies $\Sigma \not\models X(pf_i : Y)$.

Proof: We first prove the following two conditions.
Condition 1: If $X \not\subseteq Z_0$, then there is an actual class $C \in X\downarrow - Z_0$ such that $C\downarrow - C \subseteq Z_0$.
Condition 2: If $X \not\subseteq Z_i$, then there is an actual class $C \in X\downarrow_{P_i} - Z_i$ such that $C\downarrow_{P_i} - C \subseteq Z_i$, where $1 \leq i \leq n$.

Assume $X \not\subseteq Z_0$. Since $X \subseteq X\downarrow$ by definition, the assumption implies $X\downarrow \not\subseteq Z_0$. Thus there is a class in $X\downarrow - Z_0$. By definition of $X\downarrow$, every subclass of a class in $X\downarrow$ is also in $X\downarrow$. Thus we can choose a class $C \in X\downarrow$ such that $C \notin Z_0$ and that every proper subclass of C is in Z_0. In other words, $C \in X\downarrow - Z_0$ and $C\downarrow - C \subseteq Z_0$. If C is actual, then Condition 1 holds. Assume that C is virtual. Then $S_{\langle C, \mathtt{Id} \rangle} \subseteq C\downarrow - C$ by (3.4), which implies $S_{\langle C, \mathtt{Id} \rangle} \subseteq Z_0$. Since the given sequence satisfies condition SCB2, $S_{\langle C, \mathtt{Id} \rangle} \subseteq Z_0$ implies $C \in Z_0$. This, however, contradicts that $C \in X\downarrow - Z_0$. Therefore C must be actual. Hence Condition 1 holds. The similar argument applies also to Condition 2.

We next prove that if $\langle \mathbf{S}, \Sigma \rangle$ is well-formed, then $X \not\subseteq Z_i$ implies $\Sigma \not\models X(pf_i : Y)$. Assume $X \not\subseteq Z_i$ for some i with $0 \leq i \leq n$. Since the sequence satisfies condition SCB2 and SCB5, (1) by Condition 1, there is an actual class $C \in X\downarrow - Z_0$ such that $C\downarrow - C \subseteq Z_0$ and (2) by Condition 2, there is an actual class $C \in X\downarrow_{P_i} - Z_i$ such that $C\downarrow_{P_i} - C \subseteq Z_i$, where $1 \leq i \leq n$. In the following we will prove that there is a pf_i-list $v_i \xrightarrow{P_i} v_{i-1} \xrightarrow{P_{i-1}} \cdots \xrightarrow{P_1} v_0$ over $\mathbf{S}$ that satisfies Σ but not $X(pf_i : Y)$, which implies $\Sigma \not\models X(pf_i : Y)$ by Lemma 8. Induction on i.

Basis. Let $i = 0$, that is $pf_i = \mathtt{Id}$. Let v_0 be a vertex such that $Cl(v_0) = C$. Clearly, v_0 itself is an $\mathtt{Id}$-list over $\mathbf{S}$. Since (1) $\langle \mathbf{S}, \Sigma \rangle$ is well-formed and (2) the $\mathtt{Id}$-list has no arc, the $\mathtt{Id}$-list satisfies Σ trivially by Lemma 6. On the other hand, since $Cl(v_0) = C \in X\downarrow - Z_0$ implies (1) $Cl(v_0) \in X\downarrow$ and (2) $Cl(v_0) \notin Y\downarrow$ by condition SCB1, the $\mathtt{Id}$-list does not satisfy $X(\mathtt{Id} : Y)$. Hence our claim holds in the basis case.

Induction. As an induction hypothesis, assume that if $i \leq j - 1$ and $X \not\subseteq Z_i$, then there is a pf_i-list $v_i \xrightarrow{P_i} v_{i-1} \xrightarrow{P_{i-1}} \cdots \xrightarrow{P_1} v_0$ over $\mathbf{S}$ that satisfies Σ but not $X(pf_i : Y)$, where $j \geq 1$. Consider the case of $i = j$. Since $C\downarrow_{P_i} - C \subseteq Z_i$ and $C \in X\downarrow_{P_i} - Z_i$ (that is, $C \notin Z_i$) and C is actual, it follows from condition SCB4 that $P_i \in Props^*(C)$. Then since $P_i \in Props^*(C)$ and $C \notin Z_i$, it follows from condition SCB3 that $S_{\langle C, P_i \rangle} \not\subseteq Z_{i-1}$. Hence (1) if $i > 1$, then there is an actual class $C' \in S_{\langle C, P_i \rangle}\downarrow_{P_{i-1}} - Z_{i-1}$ such that $C'\downarrow_{P_{i-1}} - C' \subseteq Z_{i-1}$ by Condition 2, and (2) if $i = 1$, then there is an actual class $C' \in S_{\langle C, P_i \rangle}\downarrow - Z_{i-1}$ such that $C'\downarrow - C' \subseteq Z_{i-1}$

by Condition 1. Since $C' \notin Z_{i-1}$, it follows from the induction hypothesis that there is a pf_{i-1}-list $v_{i-1} \stackrel{P_{i-1}}{\longrightarrow} v_{i-2} \stackrel{P_{i-2}}{\longrightarrow} \cdots \stackrel{P_1}{\longrightarrow} v_0$ over $\mathbf{S}$ that satisfies Σ but not $C'(pf_{i-1} : Y)$. By adding to the pf_{i-1}-list a vertex v_i with $Cl(v_i) = C$ and an arc $v_i \stackrel{P_i}{\longrightarrow} v_{i-1}$, we construct a sequence $v_i \stackrel{P_i}{\longrightarrow} v_{i-1} \stackrel{P_{i-1}}{\longrightarrow} \cdots \stackrel{P_1}{\longrightarrow} v_0$. We will prove that the sequence is a pf_i-list over $\mathbf{S}$ that satisfies Σ but not $X(pf_i : Y)$.

Let us first prove that the sequence is a pf_i-list over $\mathbf{S}$. Since that the subsequence $v_{i-1} \stackrel{P_{i-1}}{\longrightarrow} v_{i-2} \stackrel{P_{i-2}}{\longrightarrow} \cdots \stackrel{P_1}{\longrightarrow} v_0$ is a pf_{i-1}-list over $\mathbf{S}$, it suffice to show that $Cl(v_i)$ is actual and that $P_i \in Props^*(Cl(v_i))$. The former follows from the fact that C is actual. The latter follows form the fact that $P_i \in Props^*(C)$.

Let us next prove that the pf_i-list satisfies Σ. Since (1) $\langle \mathbf{S}, \Sigma \rangle$ is well-formed and (2) the pf_{i-1}-list $v_{i-1} \stackrel{P_{i-1}}{\longrightarrow} v_{i-2} \stackrel{P_{i-2}}{\longrightarrow} \cdots \stackrel{P_1}{\longrightarrow} v_0$ satisfies Σ, it follows from Lemma 6 that $Cl(v_{k-1}) \in S_{\langle Cl(v_k), P_k \rangle}$ for $1 \leq k \leq i-1$. Thus if $Cl(v_{i-1}) \in S_{\langle Cl(v_i), P_i \rangle}$, then the pf_i-list also satisfies Σ by Lemma 6. That is, it suffices to show that $Cl(v_{i-1}) \in S_{\langle Cl(v_i), P_i \rangle}$. Since the pf_{i-1}-list does not satisfy $C'(pf_{i-1} : Y)$, it holds that

$$\begin{cases} Cl(v_{i-1}) \in C'\downarrow & \text{if } i = 1, \\ Cl(v_{i-1}) \in C'\downarrow_{P_{i-1}} & \text{if } i > 1; \end{cases} \tag{4.1}$$

$$Cl(v_0) \notin Y\downarrow. \tag{4.2}$$

In the case of $i > 1$, since $C' \in S_{\langle C, P_i \rangle}\downarrow_{P_{i-1}}$ by definition of C' and $Cl(v_i) = C$, (4.1) implies $Cl(v_{i-1}) \in S_{\langle Cl(v_i), P_i \rangle}\downarrow_{P_{i-1}}$. Since $S_{\langle Cl(v_i), P_i \rangle}\downarrow_{P_{i-1}} = S_{\langle Cl(v_i), P_i \rangle}$ by (3.7), it holds that $Cl(v_{i-1}) \in S_{\langle Cl(v_i), P_i \rangle}$. Hence the pf_i-list satisfies Σ. The proof in case of $i = 1$ is similar to above.

Finally, since $Cl(v_i) = C \in X\downarrow_{P_i} - Z_i \subseteq X\downarrow_{P_i}$, (4.2) implies that the pf_i-list does not satisfy $X(pf_i : Y)$. Consequently, the induction proof has been completed. □

We are now ready to prove the following theorem.

Theorem 3: If $\langle \mathbf{S}, \Sigma \rangle$ is well-formed, then $\Sigma \models X(pf : Y)$ *iff* $\Sigma \vdash X(pf : Y)$. That is, $\mathcal{A}$ is sound and complete for deciding whether or not $\Sigma \models X(pf : Y)$.

Proof: By Lemma 2, $\Sigma \vdash X(pf : Y)$ implies $\Sigma \models X(pf : Y)$. Conversely, assume $\Sigma \not\vdash X(pf : Y)$. Let $SCbase(Y, pf) = \langle Z_n, Z_{n-1}, \cdots, Z_0 \rangle$. By Lemma 9, the assumption implies $X \not\subseteq Z_n$. Furthermore, by Lemma 10, $SCbase(Y, pf)$ satisfies conditions SCB1 to SCB5. Hence if $\langle \mathbf{S}, \Sigma \rangle$ is well-formed, then $\Sigma \not\models X(pf : Y)$ by Lemma 11. □

4.2 Computing an SC base

In this section, we will show that if $\langle \mathbf{S}, \Sigma \rangle$ is well-formed, then $SCbase(Y, pf)$ can be computed in polynomial time. Thus it can also be decided in polynomial time whether or not $\Sigma \models X(pf : Y)$. Computing $SCbase(Y, pf)$ is based on the following lemma.

Lemma 12: Assume that $\langle \mathbf{S}, \Sigma \rangle$ is well-formed. A sequence $\langle Z_n, Z_{n-1}, \cdots, Z_0 \rangle$ coincides with $SCbase(Y, pf)$ *iff* the sequence satisfies the following condition as well as conditions SCB1 to SCB5.
SCB6: $\Sigma \vdash Z_i(pf_i : Y)$ for $0 \leq i \leq n$.

Proof: $SCbase(Y, pf)$ satisfies condition SCB6 by Lemma 9, and conditions SCB1 to SCB5 by Lemma 10. Conversely, assume that a sequence $\langle Z_n, Z_{n-1}, \cdots, Z_0 \rangle$ satisfies conditions SCB1 to SCB6. Let $SCbase(Y, pf) = \langle W_n, W_{n-1}, \cdots, W_0 \rangle$. Since $\Sigma \vdash Z_i(pf_i : Y)$ by condition SCB6, it follows from Lemma 9 that $Z_i \subseteq W_i$. We have to show $W_i \subseteq Z_i$. Assume $C \notin Z_i$ for a class C. Since $\langle \mathbf{S}, \Sigma \rangle$ is well-formed and $\langle Z_n, Z_{n-1}, \cdots, Z_0 \rangle$ satisfies conditions SCB1 to SCB5, by Lemma 11 the assumption $C \notin Z_i$ implies $\Sigma \not\models C(pf_i : Y)$. Thus $\Sigma \not\vdash C(pf_i : Y)$ by Theorem 3. By Lemma 9, this implies $C \notin W_i$. Hence $W_i \subseteq Z_i$. □

The intuitive idea is that each of conditions SCB2 to SCB5 is treated as a *transformation rule* for a class C and a set Z_i by adding C to Z_i if (1) $C \notin Z_i$ and (2) the antecedent of the condition is satisfied with respect to C and Z_i. For convenience, conditions SCB2, SCB3, SCB4, and SCB5 are called *rules* SCB2, SCB3, SCB4 and SCB5, respectively, when they are treated as transformation rules.

Procedure 1: Computing $SCbase(Y, pf)$.
Input: a well-formed database schema $\langle \mathbf{S}, \Sigma \rangle$, a subset Y of $Classes(\mathbf{S})$, and a path function $pf (= P_n.P_{n-1}.\cdots.P_1)$.
Output: $SCbase(Y, pf) = \langle Z_n, Z_{n-1}, \cdots, Z_0 \rangle$.
Method:

1. Let $Z_0 \leftarrow Y\downarrow$ and $Z_i \leftarrow \emptyset$ for $1 \leq i \leq n$.
2. Apply rules SCB2 to SCB5 repeatedly to sequence $\langle Z_n, Z_{n-1}, \cdots, Z_0 \rangle$, until all of conditions SCB2 to SCB5 are satisfied with respect to every C and every Z_i. □

Note that for each of rules SCB2 to SCB5, once the antecedent of the rules is satisfied with respect to a class C and a set Z_i, then the same antecedent will continue to be satisfied with respect to C and Z_i, regardless of how the rules might be applied to the sequence. Hence

the result of Procedure 1 is independent of the order of applying the rules. We prove the correctness of Procedure 1. Let $\langle Z_n, Z_{n-1}, \cdots, Z_0 \rangle$ be the output of Procedure 1. By Theorem 3, it suffices to show that the output satisfies conditions SCB1 to SCB6. Clearly, the output satisfies conditions SCB1 to SCB5. As for condition SCB6, let us define

$$\begin{aligned}\Sigma' = &\{C(\mathtt{Id} : S_{\langle C,\mathtt{Id}\rangle}) \mid C \in \mathit{Classes}(\mathbf{S})\} \\ &\cup \{C(P : S_{\langle C,P\rangle}) \mid C \in \mathit{Classes}(\mathbf{S}) \text{ and } P \in \mathit{Props}^*(C)\}.\end{aligned}$$

Then it follows from Lemma 3 that $\Sigma \vdash \sigma$ for every $\sigma \in \Sigma'$. Thus for an SC σ', if $\Sigma' \vdash \sigma'$, then $\Sigma \vdash \sigma'$. Hence by the following lemma, the output satisfies condition SCB6.

Lemma 13: Let $\langle Z_n, Z_{n-1}, \cdots, Z_0 \rangle$ be the output of Procedure 1. Then $\Sigma' \vdash Z_i(pf_i : Y)$, where $0 \leq i \leq n$.

Proof: Induction on the number of applying rules SCB2 to SCB5 in step 2 of Procedure 1.

Basis. Initially, $Z_0 = Y\downarrow$ and $Z_i = \emptyset$ for $1 \leq i \leq n$. Thus for $1 \leq i \leq n$, $\Sigma' \vdash Z_i(pf_i : Y)$ by axiom SC9. As for Z_0, let $C \in Z_0$. $Z_0 = Y\downarrow$ and $C \in Z_0$ imply $C \in Y\downarrow$. $Y \subseteq Y$ implies $Y(\mathtt{Id} : Y)$ by axiom SC1. $Y(\mathtt{Id} : Y)$ and $C \in Y\downarrow$ imply $C(\mathtt{Id} : Y)$ by axiom SC4. That is, $\Sigma' \vdash C(\mathtt{Id} : Y)$ for every $C \in Z_0$. Then $\{C(\mathtt{Id} : Y) \mid C \in Z_0\}$ implies $Z_0(\mathtt{Id} : Y)$ by axiom SC7. Hence $\Sigma' \vdash Z_0(\mathtt{Id} : Y)$. Note that $\mathtt{Id} = pf_0$. Thus our claim holds in the basis case.

Induction. As an induction hypothesis, assume during an execution of step2 that $\Sigma' \vdash Z_i(pf_i : Y)$ for $0 \leq i \leq n$. Assume that a class C should be added to Z_i by applying one of rules SCB2 to SCB5. We have to prove that $\Sigma' \vdash (Z_i \cup C)(pf_i : Y)$. However, by the induction hypothesis and axiom SC7, it suffices to show that $\Sigma' \vdash C(pf_i : Y)$. There are four cases to be considered.

Case 1: where rule SCB2 applies. Then it holds that $S_{\langle C,\mathtt{Id}\rangle} \subseteq Z_i$. By the induction hypothesis, this implies $\Sigma' \vdash S_{\langle C,\mathtt{Id}\rangle}(pf_i : Y)$. Since $S_{\langle C,\mathtt{Id}\rangle}\downarrow = S_{\langle C,\mathtt{Id}\rangle}$ by (3.6), $S_{\langle C,\mathtt{Id}\rangle}(pf_i : Y)$ is equivalent to $S_{\langle C,\mathtt{Id}\rangle}\downarrow(pf_i : Y)$. Since $C(\mathtt{Id} : S_{\langle C,\mathtt{Id}\rangle}) \in \Sigma'$, it is trivial that $\Sigma' \vdash C(\mathtt{Id} : S_{\langle C,\mathtt{Id}\rangle})$. Thus $C(\mathtt{Id} : S_{\langle C,\mathtt{Id}\rangle})$ and $S_{\langle C,\mathtt{Id}\rangle}\downarrow(pf_i : Y)$ imply $C(\mathtt{Id} \circ pf_i : Y)$ by axiom SC3, where $\mathtt{Id} \circ pf_i = pf_i$. That is, $\Sigma' \vdash C(pf_i : Y)$.

Case 2: where rule SCB3 applies. Then it holds that $P_i \in \mathit{Props}^*(C)$ and $S_{\langle C,P_i\rangle} \subseteq Z_{i-1}$. By definition, the former implies $C(P_i : S_{\langle C,P_i\rangle}) \in \Sigma'$, thereby $\Sigma' \vdash C(P_i : S_{\langle C,P_i\rangle})$. By the induction hypothesis, the latter implies $\Sigma' \vdash S_{\langle C,P_i\rangle}(pf_{i-1} : Y)$. Since $S_{\langle C,P_i\rangle}\downarrow = S_{\langle C,P_i\rangle}$ by (3.6), $S_{\langle C,P_i\rangle}(pf_{i-1} : Y)$ is equivalent to $S_{\langle C,P_i\rangle}\downarrow(pf_{i-1} : Y)$. Hence $C(P_i : S_{\langle C,P_i\rangle})$ and $S_{\langle C,P_i\rangle}\downarrow(pf_{i-1} : Y)$ imply $C(P_i \circ pf_{i-1} : Y)$ by axiom SC3, where $P_i \circ pf_{i-1} = pf_i$. That is, $\Sigma' \vdash C(pf_i : Y)$.

Case 3: where rule SCB4 applies. Then it holds that $C{\downarrow_{P_i}} - C \subseteq Z_i$ and that $P_i \notin Props^*(C)$. By the induction hypothesis, the former implies $\Sigma' \vdash (C{\downarrow_{P_i}} - C)(pf_i : Y)$. Note that $pf_i = P_i \circ pf_{i-1}$. Thus $(C{\downarrow_{P_i}} - C)(pf_i : Y)$ and $P_i \notin Props^*(C)$ imply $C(pf_i : Y)$ by axiom SC10. That is, $\Sigma' \vdash C(pf_i : Y)$.

Case 4: where rule SCB5 applies. Then it holds that $C{\downarrow_{P_i}} - C \subseteq Z_i$ and that C is virtual. Along the same line as proving Case 3, we can show that $\Sigma' \vdash C(pf_i : Y)$. □

By Theorem 3 and Lemma 13, we have another characterization of $\Sigma \models X(pf : Y)$ as follows.

Theorem 4: If $\langle \mathbf{S}, \Sigma \rangle$ is well-formed, then $\Sigma \models X(pf : Y)$ *iff* $\Sigma \vdash X(pf : Y)$ *iff* $\Sigma' \vdash X(pf : Y)$.

By Theorem 4, if Σ contains an SC $C(P : Y)$ such that $P \notin Props^*(C)$, then the SC is *redundant* to derive an SC from Σ. On the other hand, Σ' contains no such redundant SC. Since Procedure 1 is not efficient, we will refine it. It is time-consuming to find a class C and a set Z_i for which at least one of conditions SCB2 to SCB5 is not satisfied. The following procedure will find such C and Z_i *deterministically.* A sequence $\langle C_1, C_2, \cdots, C_K \rangle$ is *topologically sorted* with respect to $\stackrel{*}{\Rightarrow}$ if $i \leq j$ implies $C_j \stackrel{*}{\not\Rightarrow} C_i$, where $Classes(\mathbf{S}) = \{C_1, C_2, \cdots, C_K\}$. Since the generalization taxonomy of $\mathbf{S}$ is acyclic by assumption, there must be such a sequence. The following procedure computes $SCbase(Y, pf)$.

Procedure 2: Computing $SCbase(Y, pf)$. *(Refined)*
Input: a database schema $\langle \mathbf{S}, \Sigma \rangle$, a subset Y of $Classes(\mathbf{S})$, and a path function pf.
Output: $SCbase(Y, pf) = \langle Z_n, Z_{n-1}, \cdots, Z_0 \rangle$.
Method:

1. Let $Z_0 \leftarrow Y{\downarrow}$ and $Z_i \leftarrow \emptyset$ for $1 \leq i \leq n$.
2. Sort the classes in $Classes(\mathbf{S})$ topologically with respect to $\stackrel{*}{\Rightarrow}$.
   Let $\langle C_1, C_2, \cdots, C_K \rangle$ be the resulting sequence.
3. **for** $k \leftarrow 1$ **to** K **do**
4. **if** $S_{\langle C_k, \mathtt{Id} \rangle} \subseteq Z_0$ **then** add C_k to Z_0.
5. **for** $i \leftarrow 1$ **to** n **do**
6. **for** $k \leftarrow 1$ **to** K **do**
7. **if** $S_{\langle C_k, \mathtt{Id} \rangle} \subseteq Z_i$
   or ($P_i \in Props^*(C_k)$ and $S_{\langle C_k, P_i \rangle} \subseteq Z_{i-1}$)

or $(C_k{\downarrow}_{P_i} - C_k \subseteq Z_i$ and $P_i \notin Props^*(C_k))$
or $(C_k{\downarrow}_{P_i} - C_k \subseteq Z_i$ and C_k is virtual)
then add C_k to Z_i. □

Let us prove the correctness of Procedure 2. Clearly, steps 3 and 4 try to apply rule SCB2 to each C_k and Z_0, and steps 5 to 7 try to apply one of rules SCB2 to SCB5 to each C_k and each Z_i in a specific order. Note that rules SCB3, SCB4 and SCB5 cannot apply to any class in Z_0. Clearly, condition SCB1 is satisfied by step 1. Along the same line as proving Lemma 13, we can show that the output of Procedure 2 satisfies condition SCB6. We have to show that all of conditions SCB2 to SCB5 are satisfied by every C and every Z_i; that is, that if at least one of (1) $S_{\langle C,\mathtt{Id}\rangle} \subseteq Z_i$, (2) $P_i \in Props^*(C)$ and $S_{\langle C,P_i\rangle} \subseteq Z_{i-1}$, (3) $C{\downarrow}_{P_i} - C \subseteq Z_i$ and $P_i \notin Props^*(C)$, and (4) $C{\downarrow}_{P_i} - C \subseteq Z_i$ and C is virtual holds, then $C \in Z_i$. Induction on i.

Basis. Consider the case of $i = 0$. Note that conditions SCB3, SCB4 and SCB5 are independent of Z_0. Thus it suffices to consider only condition SCB2. Assume that there is a class that violates condition SCB2 with respect to Z_0; that is, that $S_{\langle C_l,\mathtt{Id}\rangle} \subseteq Z_0$ and $C_l \notin Z_0$ for some C_l, where $1 \leq l \leq K$. Let $Z_0^{(l)}$ denote the value of Z_0 when the value of k coincides with l in the for loop of step 3 in Procedure 2. Then $C_l \notin Z_0$ implies $S_{\langle C_l,\mathtt{Id}\rangle} \not\subseteq Z_0^{(l)}$. On the other hand, since $S_{\langle C_l,\mathtt{Id}\rangle} \subseteq Z_0$, there must exist a class $C_m \in S_{\langle C_l,\mathtt{Id}\rangle} - Z_0^{(l)}$ such that $l < m \leq K$. Since $S_{\langle C_l,\mathtt{Id}\rangle} \subseteq C_l{\downarrow}$ by (3.3) and (3.4), $C_m \in S_{\langle C_l,\mathtt{Id}\rangle} - Z_0^{(l)}$ implies $C_m \in C_l{\downarrow}$. On the other hand, since $\langle C_1, C_2, \cdots, C_K\rangle$ is topologically sorted with respect to $\stackrel{*}{\Rightarrow}$, $l < m$ implies $C_m \stackrel{*}{\not\Rightarrow} C_l$; that is, $C_m \notin C_l{\downarrow}$, which is a contradiction. Hence there is no class C_l such that $S_{\langle C_l,\mathtt{Id}\rangle} \subseteq Z_0$ and $C_l \notin Z_0$.

Induction. As an induction hypothesis, assume that conditions SCB2 to SCB5 are satisfied with respect to every C and every Z_i with $0 \leq i \leq j-1$, where $j \geq 1$. Consider the case of $i = j$. We must prove that if at least one of (1) $S_{\langle C,\mathtt{Id}\rangle} \subseteq Z_j$, (2) $P_j \in Props^*(C)$ and $S_{\langle C,P_j\rangle} \subseteq Z_{j-1}$, (3) $C{\downarrow}_{P_j} - C \subseteq Z_j$ and $P_j \notin Props^*(C)$, and (4) $C{\downarrow}_{P_j} - C \subseteq Z_j$ and C is virtual holds, then $C \in Z_j$. Along the same line as proving the basis above, it can be proved that if $S_{\langle C,\mathtt{Id}\rangle} \subseteq Z_j$, then $C \in Z_j$. Since the value of Z_{j-1} is *fixed* when the value of i coincides with j in the for loop of step 5, it must hold that if $P_j \in Props^*(C)$ and $S_{\langle C,P_j\rangle} \subseteq Z_{j-1}$, then $C \in Z_j$. Assume that $C_l{\downarrow}_{P_j} - C_l \subseteq Z_j$ and $P_j \notin Props^*(C_l)$ but $C_l \notin Z_j$ for some C_l. Let $Z_j^{(l)}$ be the value of Z_j when the value of k coincides with l in the for loop of step 6. Then $C_l \notin Z_j$ implies $C_l{\downarrow}_{P_j} - C_l \not\subseteq Z_j^{(l)}$. Furthermore, since $C_l{\downarrow}_{P_j} - C_l \subseteq Z_j$, there must exist a class $C_m \in (C_l{\downarrow}_{P_j} - C_l) - Z_j^{(l)}$ such that $l < m \leq K$. Then contradiction would be induced as in the basis. Thus if $C_l{\downarrow}_{P_j} - C_l \subseteq Z_j$ and $P_j \notin Props^*(C_l)$, then $C_l \in Z_j$. Finally, assume

that $C_l{\downarrow}_{P_j} - C_l \subseteq Z_j$ and C_l is virtual but $C_l \notin Z_j$ for some C_l. Let $Z_j^{(l)}$ be the value of Z_j when the value of k coincides with l in the for loop of step 6. Along the same line as in above, we can show that if $C_l{\downarrow}_{P_j} - C_l \subseteq Z_j$ and C_l is virtual, then $C_l \in Z_j$. Hence the induction proof follows. □

Let us now consider the running time of Procedure 2. Note that we have computed $S_{\langle C,\mathtt{Id}\rangle}$ and $S_{\langle C,P\rangle}$ for all C and for all $P \in Props^*(C)$ in order to test the well-formedness of $\langle \mathbf{S}, \Sigma\rangle$. In what follows, we assume that such sets are given. Since $Y{\downarrow}$ can be computed in $O(M)$ time, step 1 requires $O(M + n \cdot K)$ time. It is well-known that a topological sort of a directed acyclic graph can be executed in linear time. Step 2 corresponds to executing a topological sort of the GT-graph $G_{\mathtt{S}}(V_{\mathtt{S}}, A_{\mathtt{S}})$, whose size is $O(K + M)$. Thus step 2 requires $O(K + M)$ time. Since set containment "$S_{\langle C_k,\mathtt{Id}\rangle} \subseteq Z_0$" can be tested in $O(K)$ time, the if condition of step 4 can be tested in $O(K)$ time, thereby steps 3 and 4 require $O(K^2)$ time. As for the if condition of step 7, we must compute $C_k{\downarrow}_{P_i}$. Since $G_{\mathtt{S}}^{P_i}$ can be constructed in $O(M)$ time, $C_k{\downarrow}_{P_i}$ can be constructed in that time. Since set containment "$C_k{\downarrow}_{P_i} - C_k \subseteq Z_i$" can be tested in $O(K)$ time, the if condition of step 7 can be executed in $O(K + M)$ time. Then steps 5 to 7 require $O(n \cdot K \cdot (K + M))$ time, where $n = len(pf)$. Hence Procedure 2 can be executed in that time.

Theorem 5: For a database schema $\langle \mathbf{S}, \Sigma\rangle$ and an SC $X(pf : Y)$, it can be decided in $O(len(pf) \cdot K \cdot (K + M))$ time whether or not $\Sigma \models X(pf : Y)$.

5 Testing Possibility of Path Functions

In this section, we will show that if $\langle \mathbf{S}, \Sigma\rangle$ is well-formed, then it can be decided in polynomial time whether or not a path function pf is in $PossPaths(C)$ for a class C. For a class C and a path function pf, let us denote a predicate "$pf \in PossPaths(C)$" by $C(pf)$. It is straightforward that

$$\mathtt{Id} \in PossPaths(C) \text{ for every } C. \tag{5.1}$$

Lemma 14: Let $G(V, A)$ be a database over $\mathbf{S}$ satisfying Σ. For a path function pf, if $u \rightsquigarrow_{pf} v \in G$, then there is a pf-list $v_n \xrightarrow{P_n} v_{n-1} \xrightarrow{P_{n-1}} \cdots \xrightarrow{P_1} v_0$ over $\mathbf{S}$ satisfying Σ, $Cl(v_n) = Cl(u)$, and $Cl(v_0) = Cl(v)$, where $pf = P_n.P_{n-1}.\cdots.P_1$.

Proof: The path $u \rightsquigarrow_{pf} v$ can be denoted $u_n \xrightarrow{P_n} u_{n-1} \xrightarrow{P_{n-1}} \cdots \xrightarrow{P_1} u_0$, where $u_n = u$ and $u_0 = v$. Note that a vertex u_i may coincide with another u_j, since G may contain cycles. Let $v_n \xrightarrow{P_n} v_{n-1} \xrightarrow{P_{n-1}} \cdots \xrightarrow{P_1} v_0$ be a sequence such that $Cl(v_i) = Cl(u_i)$ for $0 \leq i \leq n$. We will prove that this is a desired pf-list.

Since G is a database over $\mathbf{S}$, it satisfies property value integrity. Thus $P_i \in Props^*(Cl(u_i))$ for $1 \leq i \leq n$. Since $Cl(v_i) = Cl(u_i)$ by definition, the sequence also satisfies property value integrity. Thereby the sequence is a pf-list. Since G satisfies Σ by assumption and the path $u_n \xrightarrow{P_n} u_{n-1} \xrightarrow{P_{n-1}} \cdots \xrightarrow{P_1} u_0$ is a subgraph of G, the path must also satisfy Σ. Thus the pf-list also satisfies Σ. Note that $Cl(v_n) = Cl(u)$ and $Cl(v_0) = Cl(v)$ by definition. Hence Lemma 14 holds. □

Lemma 15: Let C be a class and let $pf(= P_n.P_{n-1}.\cdots.P_1)$ be a path function. Then $pf \in PossPaths(C)$ *iff* there is a pf-list $v_n \xrightarrow{P_n} v_{n-1} \xrightarrow{P_{n-1}} \cdots \xrightarrow{P_1} v_0$ over $\mathbf{S}$ satisfying Σ and the following condition.

$$\begin{cases} Cl(v_n) \in C\downarrow & \text{if } n = 0, \\ Cl(v_n) \in C\downarrow_{P_n} & \text{if } n > 0; \end{cases}$$

Proof: *If part.* Assume that there is a pf-list $v_n \xrightarrow{P_n} v_{n-1} \xrightarrow{P_{n-1}} \cdots \xrightarrow{P_1} v_0$ over $\mathbf{S}$ satisfying the above condition. By Lemma 7, there is a database $G(V, A)$ over $\mathbf{S}$ that (1) satisfies Σ and that (2) contains the pf-list as a subgraph. There are two cases to be considered: $n = 0$ and $n > 0$.

If $n = 0$, then (2) implies that there exists a vertex $v_0 \in V$ such that $Cl(v_0) \in C\downarrow$. Thus $pf \in PossPaths(C)$ by definition. If on the other hand $n > 0$, then (2) implies $v_n \rightsquigarrow_{pf} v_0 \in G$, where $Cl(v_n) \in C\downarrow_{P_n}$. Thus $pf \in PossPaths(C)$ by definition.

Only if part. Assume $pf \in PossPaths(C)$. There are two cases to be considered: $n = 0$ and $n > 0$.

Case 1: where $n = 0$. By definition, there is a database $G(V, A)$ over $\mathbf{S}$ that satisfies Σ and that contains a vertex $u \in V$ such that $Cl(u) \in C\downarrow$. By Lemma 14, there is a pf-list v_0 over $\mathbf{S}$ satisfying Σ and $Cl(v_0) = Cl(u)$. Then $Cl(v_0) = Cl(u)$ implies $Cl(v_0) \in C\downarrow$.

Case 2: where $n > 0$. By definition, there is a database $G(V, A)$ over $\mathbf{S}$ that satisfies Σ and that contains a vertex u such that (1) $Cl(u) \in C\downarrow_{P_n}$ and (2) $u \rightsquigarrow_{pf} v \in G$ for some $v \in V$. By Lemma 14, there is a pf-list $v_n \xrightarrow{P_n} v_{n-1} \xrightarrow{P_{n-1}} \cdots \xrightarrow{P_1} v_0$ over $\mathbf{S}$ satisfying Σ, $Cl(v_n) = Cl(u)$, and $Cl(v_0) = Cl(v)$. Then $Cl(v_n) = Cl(u)$ implies $Cl(v_n) \in C\downarrow_{P_n}$. Hence Lemma 15 holds. □

Theorem 6: The three axioms in Table 2 are sound and complete for deciding whether or not $pf \in PossPaths(C)$; that is, $pf \in PossPaths(C)$ *iff* $C(pf)$ is derived by axioms PP1 to PP3.

PP1: $C(\mathtt{Id})$.
PP2: If $C'(pf)$ and $C' \in C{\downarrow_P}$, then $C(pf)$, where $pf = P \circ pf'$.
PP3: If $C'(pf)$, $P \in Props^*(C)$, $C' \in S_{\langle C,P\rangle}$ and C is actual, then $C(P \circ pf)$.

Table 2: Axioms for *PossPaths*.

Proof: *If part.* We prove that if $C(pf)$ is derived by axioms PP1 to PP3, then $pf \in PossPaths(C)$. Induction on the number of applying the axioms to derive $C(pf)$.

Basis. Assume that $C(pf)$ is derived by a single application of the axioms. Then $C(pf)$ should be derived by axiom PP1, thereby has the form $C(\mathtt{Id})$. Then $\mathtt{Id} \in PossPaths(C)$ by (5.1).

Induction. As an induction hypothesis, assume that if $C(pf)$ is derived by applying the axioms less than i times, then $pf \in PossPaths(C)$, where $i \geq 2$. Let $pf = P_n.P_{n-1}.\cdots P_0$. Assume that $C(pf)$ is derived by applying the axioms exactly i times. Consider the last applied axiom, which must be either PP2 or PP3.

Case 1: where the last axiom is PP2. Then there is a class C' such that $C'(pf)$ and $C' \in C{\downarrow_{P_n}}$. By the induction hypothesis, $C'(pf)$ implies $pf \in PossPaths(C')$. Thus there is a database $G(V, A)$ over $\mathbf{S}$ that satisfies Σ and that contains a path from a vertex u described by pf, where $Cl(u) \in C'{\downarrow_{P_n}}$. Since $C' \in C{\downarrow_{P_n}}$ implies $C'{\downarrow_{P_n}} \subseteq C{\downarrow_{P_n}}$, it holds that $Cl(u) \in C{\downarrow_{P_n}}$. Thus $pf \in PossPaths(C)$ by definition.

Case 2: where the last axiom is PP3. Then C is actual, and there is a class C' such that $C'(pf)$, $P \in Props^*(C)$ and $C' \in S_{\langle C,P\rangle}$. We have to show that $P \circ pf \in PossPaths(C)$. By the induction hypothesis, $C'(pf)$ implies $pf \in PossPaths(C')$. Thus by Lemma 15, there is a pf-list $v_n \xrightarrow{P_n} v_{n-1} \xrightarrow{P_{n-1}} \cdots \xrightarrow{P_1} v_0$ over $\mathbf{S}$ satisfying (1) Σ and (2) $Cl(v_n) \in C'{\downarrow_{P_n}}$ if $n > 0$, $Cl(v_n) \in C'{\downarrow}$ if $n = 0$. By adding a vertex v_{n+1} such that $Cl(v_{n+1}) = C$ and an arc $v_{n+1} \xrightarrow{P} v_n$ to the pf-list, we can construct a $(P \circ pf)$-list. Since $C' \in S_{\langle C,P\rangle}$ implies $Cl(v_n) \in S_{\langle Cl(v_{n+1}),P\rangle}$, it follows from Lemma 6 that the $(P \circ pf)$-list satisfies Σ. Since $Cl(v_{n+1}) = C$ implies $Cl(v_{n+1}) \in C{\downarrow_P}$, it follows from Lemma 15 that $P \circ pf \in PossPaths(C)$. This completes the induction proof.

Only if part. We prove that if $pf \in PossPaths(C)$, then $C(pf)$ is derived by axioms PP1 to PP3. Induction on $len(pf)$.

Basis. If $pf = \mathtt{Id}$, then $C(pf)$ is derived by axiom PP1.

Induction. As an induction hypothesis, assume that if $len(pf) < n$ and $pf \in PossPaths(C)$, then $C(pf)$ is derived by axioms PP1 to PP3, where $n \geq 1$. Consider the case of $len(pf) = n$ and let $pf = P_n.P_{n-1}.\cdots.P_1$. By Lemma 15, $pf \in PossPaths(C)$ implies that there is a pf-list $v_n \xrightarrow{P_n} v_{n-1} \xrightarrow{P_{n-1}} \cdots \xrightarrow{P_1} v_0$ over $\mathbf{S}$ satisfying (1) Σ and (2) $Cl(v_n) \in C{\downarrow}_{P_n}$. We claim that $Cl(v_n)(pf)$ is derived by axioms PP1 to PP3.

Clearly, $v_{n-1} \xrightarrow{P_{n-1}} v_{n-2} \xrightarrow{P_{n-2}} \cdots \xrightarrow{P_1} v_0$ is a pf_{n-1}-list over $\mathbf{S}$ satisfying (1) Σ, (2) $Cl(v_{n-1}) \in Cl(v_{n-1}){\downarrow}$, and (3) $Cl(v_{n-1}) \in Cl(v_{n-1}){\downarrow}_{P_{n-1}}$ if $n > 1$. Thus $pf_{n-1} \in PossPaths(Cl(v_{n-1}))$ by Lemma 15. Since $len(pf_{n-1}) = n - 1$, it follows from the induction hypothesis that $Cl(v_{n-1})(pf_{n-1})$ is derived by axioms PP1 to PP3. To prove that $Cl(v_n)(pf)$ is derived by axioms PP1 to PP3, we show that $Cl(v_n)(P_n \circ pf_{n-1})$ is derived from $Cl(v_{n-1})(pf_{n-1})$ by axiom PP3, where $pf = P_n \circ pf_{n-1}$. By definition of axiom PP3, it suffices to show that (1) $P_n \in Props^*(Cl(v_n))$, (2) $Cl(v_{n-1}) \in S_{\langle Cl(v_n), P_n \rangle}$ and (3) $Cl(v_n)$ is actual. Since $v_n \xrightarrow{P_n} v_{n-1} \xrightarrow{P_{n-1}} \cdots \xrightarrow{P_1} v_0$ is a pf-list over $\mathbf{S}$, conditions (1) and (3) follow from definition of the pf-list. Furthermore, since the pf-list satisfies Σ, condition (2) follows from Lemma 6.

Note that $Cl(v_n) \in C{\downarrow}_{P_n}$. Thus $C(pf)$ is derived from $Cl(v_n)(pf)$ and $Cl(v_n) \in C{\downarrow}_{P_n}$ by axiom PP2. Hence $C(pf)$ is derived by axioms PP1 to PP3. This completes proving the induction proof. □

We present how to decide whether or not $pf \in PossPaths(C)$. For a path function $pf(= P_n.P_{n-1}.\cdots.P_1)$, let $PPbase(pf)$ be a sequence $\langle Z_n, Z_{n-1}, \cdots, Z_0 \rangle$ such that $Z_i = \{C \mid pf_i \in PossPaths(C)\}$. We can compute $PPbase(pf)$ by the following procedure.

Procedure 3: Computing $PPbase(pf)$.
Input: a well-formed database schema $\langle \mathbf{S}, \Sigma \rangle$ and a path function pf.
Output: $PPbase(pf) = \langle Z_n, Z_{n-1}, \cdots, Z_0 \rangle$.
Method:

1. Let $Z_0 \leftarrow Classes(\mathbf{S})$ and $Z_i \leftarrow \emptyset$ for $1 \leq i \leq n$.
2. **for** $i \leftarrow 1$ **to** n **do**
3. **for** each actual class C **do**
4. **if** $P_i \in Props^*(C)$ and $Z_{i-1} \cap S_{\langle C, P_i \rangle} \neq \emptyset$ **then**
   add all C' such that $C \in C'{\downarrow}_{P_i}$ to Z_i. □

We prove the correctness of Procedure 3. Let $\langle Z_n, Z_{n-1}, \cdots, Z_0 \rangle$ be the output of the procedure. We must show that $C \in Z_i$ *iff* $pf_i \in PossPaths(C)$. Induction on i.

Basis. Consider the case of $i = 0$. Since $Z_0 = Classes(\mathbf{S})$ by step 1 and $\mathtt{Id} \in PossPaths(C)$ for every C by (5.1), the basis holds.

Induction. As an induction hypothesis, assume that if $i < j$, then $C \in Z_i$ *iff* $pf_i \in PossPaths(C)$, where $j \geq 1$. Consider the case of $i = j$.

We first prove that $C \in Z_i$ implies $pf_i \in PossPaths(C)$. Let $C \in Z_i$. By steps 3 to 4, there must exist an actual class C' such that (1) $P_i \in Props^*(C')$, (2) $Z_{i-1} \cap S_{\langle C', P_i \rangle} \neq \emptyset$, and (3) $C' \in C{\downarrow}_{P_i}$. By the induction hypothesis, $Z_{i-1} \cap S_{\langle C', P_i \rangle} \neq \emptyset$ implies that there is a class $C_{i-1} \in S_{\langle C', P_i \rangle}$ such that $pf_{i-1} \in PossPaths(C_{i-1})$. By Theorem 6, $pf_{i-1} \in PossPaths(C_{i-1})$ implies that $C_{i-1}(pf_{i-1})$ is derived by axiom PP1 to PP3. Since $P_i \in Props^*(C')$, $C_{i-1} \in S_{\langle C', P_i \rangle}$ and C' is actual, $C'(P_i \circ pf_{i-1})$ is derived from $C_{i-1}(pf_{i-1})$ by axiom PP3, where $P_i \circ pf_{i-1} = pf_i$. Note that $C' \in C{\downarrow}_{P_i}$ by (3). Thus $C(pf_i)$ is derived from $C'(pf_i)$ and $C' \in C{\downarrow}_{P_i}$ by axiom PP2. That is, $C(pf_i)$ is derived by axioms PP1 to PP3. Hence $pf_i \in PossPaths(C)$ by Theorem 6.

Conversely, assume $pf_i \in PossPaths(C)$. By Lemma 15, there is a pf_i-list $v_i \xrightarrow{P_i} v_{i-1} \xrightarrow{P_{i-1}} \cdots \xrightarrow{P_1} v_0$ over $\mathbf{S}$ satisfying Σ and $Cl(v_i) \in C{\downarrow}_{P_i}$. We claim that any class C such that $Cl(v_i) \in C{\downarrow}_{P_i}$ is in Z_i.

By steps 3 to 4, it suffices to show that (1) $P_i \in Props^*(Cl(v_i))$, and (2) $Z_{i-1} \cap S_{\langle Cl(v_i), P_i \rangle} \neq \emptyset$. Since $v_i \xrightarrow{P_i} v_{i-1} \xrightarrow{P_{i-1}} \cdots \xrightarrow{P_1} v_0$ is a pf_i-list over $\mathbf{S}$, condition (1) follows from definition of the pf_i-list. As for condition (2), we will show that $Cl(v_{i-1}) \in Z_{i-1} \cap S_{\langle Cl(v_i), P_i \rangle}$. Since the pf_i-list satisfies Σ, $Cl(v_{i-1}) \in S_{\langle Cl(v_i), P_i \rangle}$ by Lemma 6. It remains to show that $Cl(v_{i-1}) \in Z_{i-1}$. Clearly, $v_{i-1} \xrightarrow{P_{i-1}} v_{i-2} \xrightarrow{P_{i-2}} \cdots \xrightarrow{P_1} v_0$ is a pf_{i-1}-list over $\mathbf{S}$ satisfying (1) Σ, (2) $Cl(v_{i-1}) \in Cl(v_{i-1}){\downarrow}_{P_{i-1}}$ if $i > 1$, and (3) $Cl(v_{i-1}) \in Cl(v_{i-1}){\downarrow}$. Thus $pf_{i-1} \in PossPaths(Cl(v_{i-1}))$ by Lemma 15. Hence $Cl(v_{i-1}) \in Z_{i-1}$ by the induction hypothesis. This completes proving the claim. Consequently, Procedure 3 correctly computes $PPbase(pf)$.

We now consider the running time of Procedure 3. Step 1 requires $O(n \cdot K)$ time. Note that the if condition of step 4 can be tested in $O(K)$ time and that we can compute the set of superclasses of C with respect to P in $O(M)$ time using $G^P_{\mathtt{S}}$ graph $G^P_{\mathtt{S}}(V^P_{\mathtt{S}}, A^P_{\mathtt{S}})$. Thus steps 2 to 4 require $O(n \cdot K \cdot (K + M))$ time. Hence Procedure 3 can also be executed in that time.

Theorem 7: For a database schema $\langle \mathbf{S}, \Sigma \rangle$, a class C, and a path function pf, it can be decided in $O(len(pf) \cdot K \cdot (K + M))$ time whether or not $pf \in PossPaths(C)$.

6 Conclusion

In this paper, we showed a polynomial-time algorithm for deciding whether or not a given database schema is well-formed. Then we gave a polynomial-time algorithm that decides whether or not a given SC is a logical consequence of Σ. Finally, we gave another polynomial-time algorithm that decides whether or not a given path function is possible.

References

[1] Ito, M. and Nakanishi, M., "On Deriving Specialization Constraints over Complex Objects," in *Proc. 3rd Int. Symposium on Database System for Advanced Applications*, pp.335–342, Apr. 1993.

[2] Ito, M, Weddell, G. E. and Coburn, N., "On specialization constraints over complex objects," *Technical Report, University of Waterloo*, CS91-62, 1991.

[3] Kiffer, M., Lausen, G. and Wu, J., "Logical Foundations of Object Oriented and Frame-Based Languages," *Technical Report 90/14, State University of New York at Stony Brook*, Jun. 1990.

[4] Weddell, G. E. and Coburn, N., "A theory of specialization on constraints for complex objects," in *Proc. 3rd Int. Conf. on Database Theory*, pp.229–244, Dec. 1990.