Representing and Reasoning on Conceptual Queries Over Image Databases

Mohand-Saïd Hacid Christophe Rigotti

LTCS-Report 99-02

This is an extended version of the article in: Eleventh International Symposium on Methodologies for Intelligent Systems, Warsaw, Poland, 1999.

Abstract

The problem of content management of multimedia data types (e.g., image, video, graphics) is becoming increasingly important with the development of advanced multimedia applications. Traditional database management systems are inadequate for the handling of such data types. They require new techniques for query formulation, retrieval, evaluation, and navigation. In this paper we develop a knowledge-based framework for modeling and retrieving image data by content. To represent the various aspects of an image object's characteristics, we propose a model which consists of three layers: **(1)** Feature and Content Layer, *intended to contain image visual features such as contours, shapes, etc.;* **(2)** Object Layer, *which provides the (conceptual) content dimension of images; and* **(3)** Schema Layer, *which contains the structured abstractions of images, i.e., a general schema about the classes of objects represented in the object layer. We propose two abstract languages on the basis of description logics: one for describing knowledge of the object and schema layers, and the other, more expressive, for making queries. Queries can refer to the form dimension (i.e., information of the* Feature and Content Layer*) or to the content dimension (i.e., information of the* Object Layer*). These languages employ a variable free notation, and they are well suited for the design, verification and complexity analysis of algorithms. As the amount of information contained in the previous layers may be huge and operations performed at the* Feature and Content Layer *are time-consuming, resorting to the use of materialized views to process and optimize queries may be extremely useful. For that, we propose a formal framework for testing containment of a query in a view expressed in our query language. The algorithm we propose is sound and complete and relatively efficient.*

Keywords: *Intelligent Information Retrieval, Knowledge Representation, Methodologies, Content-Based Access of Images, Query Containment, Description Logics, Intentional Reasoning.*

1 Introduction

With recent progress in compression technology, it is possible for a computer to store huge amount of images, audio and even video. If such media are widely used in today's communication (e.g. in the form of home movies, education and training, scholarly research, and corporate enterprise solutions), efficient computer exploitation is still lacking. Many databases should be created to face the increasing development of advanced applications, such as digital libraries, archival and processing of images captured by remote-sensing satellites and air photos, training and education, entertainment, medical databases, virtual reality, Internet video, interactive TV, group-ware applications, etc. Though only a partial list, these advanced applications indicate that the problem of efficiently and user-friendly accessing to image/video data is widely encountered in real-life applications and solutions to it is significant. Next generation information systems will need to provide support for both textual data and multimedia data (e.g., images, video, audio). These data types do not have the canonical representations that text based objects possess. Hence, they require systems designers to re-visit certain systems with completely different design tradeoffs than conventional ones. An important feature to be considered is content-based retrieval of the multimedia data types. For example, there are two essential questions associated with content-based query systems for imaging data [16]: (1) How to specify queries, and (2) How to access the intended data efficiently for given queries. These queries may be formulated in terms of a number of different image features, and can be grossly classified into three categories [25]: (1) *form queries*, addressing images on the basis of color, texture, sketch, or shape specifications; (2) *content queries*, focusing on domain concepts, spatial constraints, or various types of attributes; (3) *mixed queries*, which combine the two previous categories.

In order to deal with these questions, formal representations of information to enable users and query optimizers to take explicit advantage of the nature of image data are required. These formal representations must be accompanied with an abstract model of an architecture for image database systems. These tasks can be realized only by bringing together enabling technologies (e.g., databases, artificial intelligence, image processing, etc.).

The issue of designing architectures for content-based query systems for imaging data has been addressed in recent work. For instance, [2] proposed an object-oriented multimedia system for content-based retrieval. It focuses on the capabilities of object-oriented database features (their extension) for supporting content-based retrieval. [32] described a visual information system prototype for images and videos on the World-Wide Web. The system rests on the use of automated agents to collect visual information on the web. [15] proposed a four-layered abstract model for retrieving medical images by feature and content. These layers can be seen as basic components of a neutral architecture reference model that can be filled in according to many different formal models and strategies.

The problem of appropriately modeling image data to facilitate content-based access is also considered. For example, [25] proposed a data model which allows to integrate various characteristics inherent to image data. The underlying query language is based on a first-order logical language. It allows to express different types of queries involving visual, spatial, mapping, and content dimensions. [22] proposed a knowledge-based framework in which image representations, extracted image features, and image semantics can be integrated for feature and content-base retrieval. The framework is suited for conceptual image queries and flexible query answering as it supports Type Abstraction Hierarchies of image features. [17] proposed a data model and a rule-based constraint query language for image and video content based indexing and retrieval. The data model is designed around the object and first order constraint paradigms. The query language, with a clear declarative and operational semantics, is used to infer relationships about information represented in the model. [33] developed a conceptual model for representing multimedia objects based on Schank's conceptual dependencies [28]. The described tool is a hybrid information retrieval system that allows knowledge objects to be indexed via semantic category scheme and utilizes semantic relations to cluster related information together. The representation allows for combining a domain of shared knowledge objects, with personalized structures built by individual users. [26] presented a description logic based approach for image retrieval. Images are represented both at the form level, and at the content level. Queries are expressions of a fuzzy logical language obtained by extending $\mathcal{ALC}$ [29]. We will return to some of these works and others in Section 4.

In image database applications, queries often make reference to the data's content and concern huge amount of data. These queries may take long time to complete. Equally apparent is the need for new techniques capable of efficiently handling such queries.

Despite these proposals and others on finding appropriate representations of image data and systems architectures able to support such representations, there is little research work on finding semantic foundations for query optimization in image databases. This paper is a contribution in this direction. We take a new look at the problem of modeling and querying image data and find that knowledge representation and reasoning techniques for concept languages developed in Artificial Intelligence provide an interesting angle to attack such problems, and these techniques also provide a nice basis for semantic query optimization in image databases. We exploit the possibility of using two languages: one for defining the schema (i.e. the structure) of an image database and populating it, and the other, more expressive, for querying the database through the schema. These languages are equipped with *sound*, *complete*, and *terminating* inference procedures, that allow various forms of reasoning to be carried out on the intensional level of the image database.

We build on work by Chu et al. [15] to propose three layers for representing image content:

(1) *Feature and Content Layer:* It contains image features such as contours, spatial relationships, etc. This layer is characterized by a set of techniques allowing to retrieve images based on the similarity of physical features such as region, color, and shape.

(2) *Object Layer:* This layer contains objects of interest, their descriptions, and relationships among objects based on the extracted features[1]. This layer constitutes what we call the extensional part of an image database. Objects in an image are represented in the object layer as visual entities. Instances of visual objects consist of conventional attributes (e.g. name, patientID, date, etc.) as well as visual attributes (e.g. shape, size, etc.) of objects contained in the feature and content layer.

The interface between the *Feature and Content Layer* and the *Object Layer* is determined by the *Feature and Content Layer* itself and a set of predicates (e.g., similar-to predicates) over this Feature and Content Layer. Objects of the *Object Layer* are related to objects of the Feature and Content Layer via attributes.

(3) *Schema Layer:* This layer is intended to capture the structured abstractions and knowledge that are needed for image retrieval. It contains a general schema about the classes of objects stored in the object layer, their general properties and mutual relationships. In this layer, visual entities can be classified into a hierarchical structure known as a concept hierarchy on the basis of both conventional and visual attributes. This layer is well suited for integrating domain knowledge.

The part of a query that pertains to the Feature and Content Layer is processed by specialized signal processing procedures, and hence are time consuming. In addition, the amount of information contained in the object layer is huge. To enable quick response to the queries, the strategy based on the use of materialized[2] views to compute answers to queries can turn out to be useful. Supporting materialized views to process and optimize queries is the topic of much recent work on data-intensive applications (see, among others, [24, 6, 34]). Reasoning on queries in the case of image databases is not only relevant to determine views that can be used for answering queries, but it can be applied to organize large sets of queries into taxonomies which can be important to support navigation among constraint groups. It can help a user to find constraint groups similar or related to ones he/she is interested in, which can be particularly difficult if the groups have been defined by different users. For that, we develop a sound and complete algorithm for checking the *containment*[3] between a query and a view (which is seen as a query as well) expressed in our query language.

Although, in the basic form that we give here, the languages do not account for all aspects of image data, they constitute kernels to be extended. Showing how we can model and reason about the structure of image databases and queries is useful and significant. We hope that this work opens up a number of possible future research on modeling, querying and optimizing queries in multimedia databases.

To make the paper self-contained, we added all the proofs. These detailed proofs are intended to the referees and will not be included as such in the final version.

Paper outline: In Section 2, we develop our languages and give their Tarski-style extensional semantics. Section 3 provides a calculus for query containment and proofs for its soundness and completeness. Section 4 discusses related work. We conclude in Section 5 by anticipating on the necessary extensions. Some proofs are deferred to the Appendix.

[1]Features can be extracted *manually*, *semi-automatically* or *automatically*.

[2]A materialized view is a query whose a physical copy of each instance, answer to the query, is stored and maintained.

[3]Containment of queries is the problem of checking whether the result of one query is contained in what another query produces [1, 35].

2 The Languages

2.1 Preliminaries

As the representational formalisms presented in the following belong to the family of description logics, we first briefly introduce these logics. Description logics (also called concept logics, terminological logics, or concept languages) [9, 27] are a family of logics designed to represent the taxonomic and conceptual knowledge of a particular application domain on an abstract, logical level. They are equipped with well-defined, set-theoretic semantics. Furthermore, the interesting reasoning problems such as subsumption and satisfiability are, for most description logics, effectively decidable (see, for example, [18]).

Starting from atomic concepts and roles[4], complex concepts (and roles) are built by using a set of constructors. For example, from atomic concepts `Human` and `Female` and the atomic role `child` we can build the expression $\texttt{Human} \sqcap \forall\texttt{child}.\texttt{Female}$ which denotes the set of all `Human` whose children are (all) instances of `Female`. Here, the symbol $\sqcap$ denotes conjunction of concepts, while $\forall$ denotes (universal) value restriction.

A knowledge base in a description logic system is made up of two components: (1) the *TBox* is a general schema concerning the classes of individuals to be represented, their general properties and mutual relationships; (2) the *ABox* contains a partial description of a particular situation, possibly using the concepts defined in the *TBox*. It contains descriptions of (some) individuals of the situation, their properties and their interrelationships.

Retrieving information in a knowledge base system based on description logics is a deductive process involving both the schema (*TBox*) and its instantiation (*ABox*). In fact, the *TBox* is not just a set of constraints on possible *ABoxes*, but contains intensional information about classes. This information is taken into account when answering queries to the knowledge base. The following reasoning services are the most important ones provided by knowledge representation systems based on description logics (See [19] for an overview):

- *Concept satisfiability*: Given a knowledge base and a concept C, does there exist at least one model of the knowledge base assigning a non-empty extension to C?
- *Subsumption*: Given a knowledge base and two concepts C and D, is C more general than D? That is, is in all models of the knowledge base each instance of D also an instance of C?
- *Knowledge base satisfiability*: Are an *ABox* and a *TBox* consistent with each other? That is, does the knowledge base admit a model?
- *Instance checking*: Given a knowledge base, an individual o, its (partial) description, and a concept C, is o an instance of C in any model of the knowledge base?

Various database applications for which these reasoning services are useful are mentioned in [4, 10].

In the following, we are interested in concept satisfiability and subsumption. An *unsatisfiable* query is suggestive of an *empty answer*. A query containment problem will be reduced to a subsumption problem for concepts described in an appropriate description logic.

Before we give the syntax and semantics of our *abstract languages*, we define *concrete domains*, which are used to incorporate application-specific domains (i.e., strings, reals, integers, etc.) into the abstract domain of individuals.

Definition 1 (Concrete Domains) *A concrete domain* $\mathcal{D} = (\mathsf{dom}(\mathcal{D}), \mathsf{pred}(\mathcal{D}))$ *consists of:*

[4]A concept is interpreted as a class of objects in the domain of interest, and then can be considered as an unary predicate. Roles are interpreted as binary relations on individuals, and then considered as binary predicates.

- *the domain* $\mathsf{dom}(\mathcal{D})$,
- *a set of predicate symbols* $\mathsf{pred}(\mathcal{D})$, *where each predicate symbol* $P \in \mathsf{pred}(\mathcal{D})$ *is associated with an arity* n *and an* n*-ary relation* $P^{\mathcal{D}} \subseteq \mathsf{dom}(\mathcal{D})^n$,

In many applications (in particular when querying databases), one would like to be able to refer to concrete domains and predicates on these domains when defining queries. An example of a concrete domain could be the set of (nonnegative) integers with comparisons $(=, <, \leq, \geq, >)$.

In [5], concrete domains are restricted to so-called *admissible* concrete domains in order to keep the inference problems[5] decidable. We recall that, roughly spoken, a concrete domain $\mathcal{D}$ is called *admissible* iff (1) $\mathsf{pred}(\mathcal{D})$ is closed under negation and contains a unary predicate name $\top_{\mathcal{D}}$ for $\mathsf{dom}(\mathcal{D})$, and (2) satisfiability of finite conjunctions over $\mathsf{pred}(\mathcal{D})$ is decidable. For example, semi-algebraic sets defined by a finite number of polynomial equations or inequalities as defined in [7] are admissible concrete domains. In the following, we require only that *finite conjunctions* over $\mathsf{pred}(\mathcal{D})$ and *implication* (i.e., *entailment*) between finite conjunctions over $\mathsf{pred}(\mathcal{D})$ are *decidable.*

2.2 Schema Language ($\mathcal{SL}$)

We now introduce a simple description logic that will be used for describing the structure of an image data. Starting from atomic concepts and roles, complex concepts are built by using the universal quantification $(\forall)$ and predicate restrictions.

The syntax and the semantics of this description logic are given below.

Definition 2 (Syntax) *Let N_C, N_R, N_f be three pairwise disjoint sets of concept names, role names, and feature (i.e., functional role) names respectively, $\mathcal{D}_1, \ldots, \mathcal{D}_k$ be concrete domains. Let P be a role name, $f, f_1, \ldots, f_n$ be feature names, A be a concept name, A' be a concept name or a concrete domain name, and P_r be an n-ary predicate name. Concept terms C, D are defined by the following rules:*

$$\begin{array}{lll} C, D & \longrightarrow & A \mid \quad \textit{(primitive concept)} \\ & & \forall P.A \mid \quad \textit{(typing of role)} \\ & & \forall f.A' \mid \quad \textit{(typing of feature)} \\ & & P_r(f_1, \ldots, f_n) \quad \textit{(predicate restriction)} \end{array}$$

Let A, A_1, A_2 be concept names, A_3 be a concept name or a concrete domain name, D be a concept term, P be a role name, and f be a feature name. Then
$A \dot{\preceq} D$ (we say A is a subconcept of D), $P \dot{\preceq} A_1 \times A_2$, $f \dot{\preceq} A_1 \times A_3$

are called *axioms.*

$A \dot{\preceq} D$ is called a primitive concept specification, where D gives necessary conditions for membership in A. The axioms $P \dot{\preceq} A_1 \times A_2$ and $f \dot{\preceq} A_1 \times A_3$ give the definition of the role P and the feature f respectively. Domain and range of a role or a feature are restricted by concepts or concrete domains.

A $\mathcal{SL}$ schema $\mathcal{S}$ consists of a finite set of axioms. In the following, we consider only *acyclic* schemas. A schema $\mathcal{S}$ is acyclic if no concept name occurs–neither directly nor indirectly–within its own specification.

Definition 3 (Semantics) *The semantics is given by an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, which consists of an (abstract) interpretation domain $\Delta^{\mathcal{I}}$, and an interpretation function $\cdot^{\mathcal{I}}$. The abstract domain has to be disjoint from any given concrete domain, i.e., $\Delta^{\mathcal{I}} \cap \mathsf{dom}(\mathcal{D}_i) = \emptyset$ for all concrete domain $\mathcal{D}_i$ ($i \in [1, k]$),*

[5] Such as subsumption, instantiation, and consistency.

the concrete domains are pairwise disjoint, and $\mathsf{pred}(\mathcal{D}_i) \cap \mathsf{pred}(\mathcal{D}_j) = \emptyset$ *for* $i \neq j$*. The interpretation function* $\cdot^{\mathcal{I}}$ *associates each concept* C *with a subset* $C^{\mathcal{I}}$ *of* $\Delta^{\mathcal{I}}$*, each role* P *with a binary relation* $P^{\mathcal{I}}$ *on* $\Delta^{\mathcal{I}}$*, and each feature name* f *with a partial function* $f^{\mathcal{I}} : \Delta^{\mathcal{I}} \to (\Delta^{\mathcal{I}} \cup (\bigcup_{i=1}^{k} \mathsf{dom}(\mathcal{D}_i)))$*. Additionally,* $\mathcal{I}$ *has to satisfy the following equations:*

$$\begin{array}{rcl}
(\forall P.A)^{\mathcal{I}} & = & \{d \in \Delta^{\mathcal{I}} \mid \forall d' \in \Delta^{\mathcal{I}} : \\
 & & \quad (d^{\mathcal{I}}, d'^{\mathcal{I}}) \in P^{\mathcal{I}} \to d'^{\mathcal{I}} \in A^{\mathcal{I}}\} \\
(\forall f.A')^{\mathcal{I}} & = & \{d \in \Delta^{\mathcal{I}} \mid \text{ if } f^{\mathcal{I}}(d^{\mathcal{I}}) \text{ is defined then} \\
 & & \quad f^{\mathcal{I}}(d^{\mathcal{I}}) \in A'^{\mathcal{I}}\} \\
(P_r(f_1, \ldots, f_n))^{\mathcal{I}} & = & \{d \in \Delta^{\mathcal{I}} \mid (f_1^{\mathcal{I}}(d^{\mathcal{I}}), \ldots, f_n^{\mathcal{I}}(d^{\mathcal{I}})) \in P_r^{\mathcal{D}}\}
\end{array}$$

An interpretation $\mathcal{I}$ satisfies the axiom $A \dot{\preceq} D$ iff $A^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, the axiom $P \dot{\preceq} A_1 \times A_2$ iff $P^{\mathcal{I}} \subseteq A_1^{\mathcal{I}} \times A_2^{\mathcal{I}}$, and the axiom $f \dot{\preceq} A_1 \times A_3$ iff $f^{\mathcal{I}} \subseteq A_1^{\mathcal{I}} \times A_3^{\mathcal{I}}$. If A_3 is a concrete domain name then $A_3^{\mathcal{I}}$ stands for the domain of A_3 (i.e., $\mathsf{dom}(A_3)$) for all $\mathcal{I}$. In the following: **(1)** individuals of the abstract domain are called *abstract individuals*, and those of a concrete domain are called *concrete individuals*, and **(2)** we assume the *Unique Name Assumption* for abstract individuals but not for concrete individuals. If we want to treat a unique name assumption for the concrete individuals we have to require that the concrete domains contain a predicate name equality.

Definition 4 (Model) *An interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, .^{\mathcal{I}})$ *is a model, also called a valid interpretation, of a schema* $\mathcal{S}$ *iff it satisfies every axiom in* $\mathcal{S}$*.*

An interpretation $\mathcal{I}$ that satisfies all axioms in $\mathcal{S}$ is called an $\mathcal{S}$-interpretation.

Example 1 *Consider a traveling agency which sells stays on resorts. This agency has a database containing both textual information and images (compact representations) about resorts (such as cities, art galleries, lodging, etc.). Before selling a traveling to a customer, he/she is invited to virtually discover his tour by referring to information contained in the database. Figure 1 shows a simple fragment of the structure of such a database. Each inclusion assertion (introduced by* $\dot{\preceq}$*) imposes a constraint on the instances of the class it refers to. The concrete domains required here are* INTEGER*,* STRING*, and* IMAGE*. The domain* IMAGE *is a set of images structured by a set of predicates (e.g.,* similar-to *predicates).*

The language introduced previously allows to describe knowledge about classes of individuals and relationships between these classes. We can now turn our attention to the extensional level, which we call the *ABox*. The *ABox* essentially allows one to specify instance-of relations between individuals and classes (concepts), and between pairs of individuals and roles or features.

Definition 5 *Let* N_I *and* N_D *be two disjoint alphabets of symbols, called abstract individual names and concrete individual names respectively. Instance-of relationships are expressed in terms of* membership assertions *of the form:*

$$a : C, \ \ (a, b) : P, \ \ (a, b) : f, \ \ (a, z) : f, \ \ (z_1, \ldots, z_n) : P_r$$

where a *and* b *are abstract individual names,* $z, z_1, \ldots, z_n$ *are concrete individual names,* C *is a concept name or an arbitrary concept,* P *is a role name, and* P_r *is an* n*-ary predicate name of a concrete domain. Intuitively, the first form states that* a *is an instance of* C*, and the second form states that* a *is related to* b *by means of the role* P *(we also say* b *is a* P-successor *of* a*).*

In order to assign a meaning to membership assertions, the extension function $.^{\mathcal{I}}$ of an interpretation $\mathcal{I}$ is extended to individuals by mapping them to elements of $\Delta^{\mathcal{I}}$ in such a way that $a^{\mathcal{I}} \neq b^{\mathcal{I}}$ if $a \neq b$ (Unique Name Assumption). *For concrete individuals, the unique name assumption does not hold.*

$Country \;\dot{\preceq}\; \forall continent.Continent$
$Country \;\dot{\preceq}\; \forall population.\mathtt{INTEGER}$
$Country \;\dot{\preceq}\; \forall political\_situation.\mathtt{STRING}$
$Country \;\dot{\preceq}\; \forall area.\mathtt{INTEGER}$
$Country \;\dot{\preceq}\; \forall image.\mathtt{IMAGE}$
$Country \;\dot{\preceq}\; \forall average\_summer\_temperature.\mathtt{INTEGER}$

$City \;\dot{\preceq}\; \forall name.\mathtt{STRING}$
$City \;\dot{\preceq}\; \forall in\_country.Country$
$City \;\dot{\preceq}\; \forall accommodation.Accommodation$
$City \;\dot{\preceq}\; \forall image.\mathtt{IMAGE}$

$Hotel \;\dot{\preceq}\; Accommodation$
$Hotel \;\dot{\preceq}\; \forall price\_single.\mathtt{INTEGER}$
$Hotel \;\dot{\preceq}\; \forall price\_double.\mathtt{INTEGER}$
$Hotel \;\dot{\preceq}\; \forall image.\mathtt{IMAGE}$

$Room \;\dot{\preceq}\; Accommodation$
$Room \;\dot{\preceq}\; \forall price.\mathtt{INTEGER}$
$\ldots$

$Camping \;\dot{\preceq}\; Accommodation$
$Camping \;\dot{\preceq}\; \forall price.\mathtt{INTEGER}$
$Camping \;\dot{\preceq}\; \forall image.\mathtt{IMAGE}$
$Camping \;\dot{\preceq}\; \forall in\_city.City$

$Site \;\dot{\preceq}\; \forall name.\mathtt{STRING}$

$Monument \;\dot{\preceq}\; Site$
$Monument \;\dot{\preceq}\; \forall opening\_hour.\mathtt{INTEGER}$
$Monument \;\dot{\preceq}\; \forall closing\_hour.\mathtt{INTEGER}$
$Monument \;\dot{\preceq}\; \forall opening\_days.Day$
$Monument \;\dot{\preceq}\; \forall image.\mathtt{IMAGE}$
$Monument \;\dot{\preceq}\; \forall price.\mathtt{INTEGER}$

$Amusement\_parc \;\dot{\preceq}\; Site$
$Amusement\_parc \;\dot{\preceq}\; \forall price.\mathtt{INTEGER}$
$Amusement\_parc \;\dot{\preceq}\; \forall image.\mathtt{IMAGE}$

$Art\_galery \;\dot{\preceq}\; Site$
$Art\_galery \;\dot{\preceq}\; \forall price.\mathtt{INTEGER}$

$Beach \;\dot{\preceq}\; \forall image.\mathtt{IMAGE}$
$Beach \;\dot{\preceq}\; \forall average\_water\_temperature.\mathtt{INTEGER}$

$\ldots$

Figure 1: An image database structure

An interpretation $\mathcal{I}$ satisfies the assertion:

$$a : C \text{ iff } a^{\mathcal{I}} \in C^{\mathcal{I}}, \quad (a,b) : P \text{ iff } (a^{\mathcal{I}}, b^{\mathcal{I}}) \in P^{\mathcal{I}}$$

$$(a,b) : f \text{ iff } f^{\mathcal{I}}(a^{\mathcal{I}}) = b^{\mathcal{I}}, \quad (a,z) : f \text{ iff } f^{\mathcal{I}}(a^{\mathcal{I}}) = z^{\mathcal{I}}$$

$$(z_1, \ldots, z_n) : P_r \text{ iff } (z_1^{\mathcal{I}}, \ldots, z_n^{\mathcal{I}}) \in P_r^{\mathcal{D}}$$

An *ABox* $\mathcal{A}$ is a finite set of membership assertions.
An interpretation $\mathcal{I}$ is a model for an *ABox* $\mathcal{A}$ iff $\mathcal{I}$ satisfies all the assertions in $\mathcal{A}$.

Example 2 *On the basis of the schema given Figure 1, we can assert the following facts:*

$\mathsf{paris} : \mathsf{City}$, $(\mathsf{paris}, \mathsf{N}_1) : \mathsf{name}$, $\mathsf{N}_1 :=_{\text{"PARIS"}}$, $(\mathsf{paris}, \mathsf{P}) : \mathsf{population}$, $\mathsf{P} :=_{10000000}$, $(\mathsf{paris}, \mathsf{I}) : \mathsf{image}$, $\mathsf{I} :=_{\mathsf{picture}_{p01}}$, $\mathsf{eiffel\_tour} : \mathsf{Monument}$, $(\mathsf{eiffel\_tour}, \mathsf{N}_2) : \mathsf{name}$, $\mathsf{N}_2 :=_{\text{"Tour Eiffel"}}$, $\ldots$.

The first statement says that paris *is an individual, instance of the concept* City. *In the second assertion,* N_1 *is a concrete individual name. This concrete individual name is linked to the constant string "PARIS" through* $:=_{\text{"PARIS"}}$*, which stands for the unary predicate* $\{\mathsf{S} \mid \mathsf{S} = \text{"PARIS"}\}$.

In the following, we call the pair $\langle \mathcal{S}, \mathcal{A} \rangle$ composed of the database schema $\mathcal{S}$ and a possible instanciation $\mathcal{A}$ a knowledge base, and we denote it by $\mathcal{KB}$.

An interpretation $\mathcal{I}$ is a model for $\mathcal{KB}$ iff it is a model for $\mathcal{S}$ and a model for $\mathcal{A}$.

A knowledge base $\mathcal{KB}$ is *satisfiable* if it has a model. Recall that a schema $\mathcal{S}$ is a finite set of primitive concept specifications, each of the form $A \dot{\preceq} C$, where A is a concept name and C denotes an arbitrary concept, $P \dot{\preceq} A_1 \times A_2$ and $f \dot{\preceq} A_1 \times A_3$. Hence, the concept-subconcept (i.e., containment between concepts) relationships are explicitly stated. However, a given specification may contain a contradiction with regard to existing specifications. This is the case if the right hand side of a specification is unsatisfiable, and this should be detected, because it is probably due to an *error* in modeling. Additionally, a given *ABox* may contain incoherent information, for example a *multi-valuation* of a feature for a given individual. These are the reasons why we need to perform concept and knowledge base satisfiability tests.

It is well known that concept satisfiability can be reduced to the satisfiability of a knowledge base. Given a knowledge base $\mathcal{KB} = \langle \mathcal{S}, \mathcal{A} \rangle$, a concept C, and an individual o not appearing in $\mathcal{KB}$, we have:

$$C \text{ is satisfiable with respect to } \mathcal{KB} \text{ iff } \langle \mathcal{S}, \mathcal{A} \cup \{o : C\} \rangle \text{ is satisfiable}$$

2.3 A Calculus for Deciding Knowledge Base Satisfiability

In this section we provide a simple calculus for deciding knowledge base satisfiability. The method we use is based on a tableau-like calculus [20] that tries to build a model for the logical formula corresponding to a knowledge base. The interpretation being built is called a *canonical model* [11, 29].

In the following, in addition to and disjoint from the set of individual names in $N_I \cup N_D$, we have a set N_V of variables which will denote abstract individuals. For variables, the *unique name assumption* does not hold, hence two different variables can be interpreted as the same individual—in contrast to what was said for abstract individuals. Let $N_O = N_I \uplus N_V$ denote the set of variables and abstract individual names. We use the letters s, t to refer to an element of N_O (i.e., a variable or an abstract individual name) and we say simply that s and t are abstract individuals. As the unique name assumption does not hold for concrete individuals, we use $z, z', z_1, z_1^{'}, z_2, z_2^{'}, \ldots$ as names for concrete individuals. Finally, we use the letters w, w' to refer to variables, abstract individual names, or concrete individual names.

The algorithm works on a *generalized* knowledge base. A knowledge base $\mathcal{KB} = \langle \mathcal{S}, \mathcal{A} \rangle$ is translated into a generalized knowledge base $\mathcal{KB}_G$ which contains every axiom appearing in $\mathcal{S}$ and every assertion appearing in $\mathcal{A}$. Additionally, $\mathcal{KB}_G$ contains the assertion $a \not\doteq b$ for every pair of abstract individual names appearing in $\mathcal{A}$. An interpretation $\mathcal{I}$ satisfies $a \not\doteq b$ iff $a \neq b$. It is easy to see that $\mathcal{KB}$ is satisfiable if and only if $\mathcal{KB}_G$ is satisfiable.

An interpretation $\mathcal{I}$ satisfies a generalized knowledge base if it satisfies every assertion (also called constraint) in the knowledge base.

2.3.1 Propagation Rules

Let $\mathcal{KB}_G$ be a generalized knowledge base (in the following, we might omit "generalized"). The role of propagation rules applied to $\mathcal{KB}_G$ is to make *explicit* (by adding assertions to the knowledge base) the part of the knowledge which is *implicitly* contained in the knowledge base. This knowledge is implied by the semantics of the constructors of the language.

Before we can formulate the propagation rules we need a technical definition.

Definition 6 (Fork) *Let $\mathcal{KB}_G$ be a knowledge base, f be a feature name, s t be an abstract individual (i.e., an abstract individual name or a variable), b be an abstract individual name, z_1 and z_2 be concrete individual names, and x and y be variables. $\mathcal{KB}_G$ may contain the following constraints, which we call a* fork*:*

- $(s, b) : f$, *and* $(s, x) : f$. *Since f is interpreted as a partial function, such a fork means that b and x have to be interpreted as the same abstract individual. This fork can be deleted by replacing all occurrences of x in $\mathcal{KB}_G$ by b.*
- $(s, z_1) : f$ *and* $(s, z_2) : f$. *This fork is due to the fact that we do not handle unique name assumption for concrete individuals. This fork can be deleted by replacing all occurrences of z_2 in $\mathcal{KB}_G$ by z_1.*
- $(s, x) : f$ *and* $(s, y) : f$. *This fork can be deleted by replacing all occurrences of y in $\mathcal{KB}_G$ by x.*

Let P be a role name (resp. f be a feature name). In the following, we write equally sPt or $(s,t) : P$ (resp. sfw or $(s,w) : f$) to denote the fact that t is a P-successor (resp. w is a f-successor) of s.

Given a knowledge base, more than one rule might be applicable to it. Each rule takes a knowledge base $\mathcal{KB}_G$ and adds assertions to this knowledge base, thus producing a new knowledge base. We define the following strategy for the application of rules:
First apply the rules $\longrightarrow_{\dot{\preceq}}$ as long as possible, then apply the rule $\longrightarrow_{P_r}$ as long as possible, then apply $\longrightarrow_{\forall}$ as long as possible. The algorithm terminates if none of the rules can be applied.

Before starting the application of propagation rules, we suppose that fork elimination is performed on the initial generalized knowledge base.

$\mathcal{KB}_G \quad \longrightarrow_{1.\forall} \quad \{t : A\} \cup \mathcal{KB}_G$
if $s : \forall P.A, sPt$ are in $\mathcal{KB}_G$ and
$t : A$ is not in $\mathcal{KB}_G$

$\mathcal{KB}_G \quad \longrightarrow_{2.\forall} \quad \{w : A\} \cup \mathcal{KB}_G$
if $s : \forall f.A, sfw$ are in $\mathcal{KB}_G$ and
$w : A$ is not in $\mathcal{KB}_G$

$\mathcal{KB}_G \quad \longrightarrow_{1.\dot{\preceq}} \quad \{s : A_1, t : A_2\} \cup \mathcal{KB}_G$
if $sPt, P \dot{\preceq} A_1 \times A_2$ are in $\mathcal{KB}_G$ and
$s : A_1, t : A_2$ are not both in $\mathcal{KB}_G$

$\mathcal{KB}_G \quad \longrightarrow_{2.\dot{\preceq}} \quad \{s : A_1, w : A_3\} \cup \mathcal{KB}_G$
if $sfw, f \dot{\preceq} A_1 \times A_3$ are in $\mathcal{KB}_G$ and
$s : A_1, w : A_3$ are not both in $\mathcal{KB}_G$

$\mathcal{KB}_G \quad \longrightarrow_{\dot{\preceq}} \quad \{s : C\} \cup \mathcal{KB}_G$
if $s : A, A \dot{\preceq} C$ are in $\mathcal{KB}_G$ and
$s : C$ is not in $\mathcal{KB}_G$.

$\mathcal{KB}_G \quad \longrightarrow_{P_r} \quad \{s\, f_1\, z_1, \ldots, s\, f_n\, z_n, (z_1, \ldots, z_n) : P_r\} \cup \mathcal{KB}_G$
if $s : P_r(f_1, \ldots, f_n)$ is in $\mathcal{KB}_G$, and the following
does not hold
For the features $f_i, i = 1, \ldots, n,$ there are concrete
individual names $z_1^{'}, \ldots, z_n^{'}$
such that $\mathcal{KB}_G$ contains axioms $s\, f_1\, z_1^{'}, \ldots, s\, f_n\, z_n^{'}$,
$(z_1^{'}, \ldots, z_n^{'}) : P_r$, and
$z_1, \ldots, z_n$ are new concrete individual names

We may have created forks by this rule (i.e. $\longrightarrow_{P_r}$). If this is the case, we delete them as described before.

The rules are deterministic. Moreover, the rule $\longrightarrow_{P_r}$ is a generating rule, since it introduces new concrete individual names. A knowledge base is *complete* if no propagation rule applies to it. A complete knowledge base derived from a knowledge base $\mathcal{KB}_G$ is also called a *completion* of $\mathcal{KB}_G$, and denoted $\mathcal{KB}_G^{+}$.

A knowledge base contains a *clash* (contradiction) if it contains one of the following forms:

- $\{s\, f\, a, s\, f\, b\}$ for a feature f, where a and b are abstract individual names.
- $\{s\, f\, t, s\, f\, z\}$ for a feature f. t is an abstract individual and z is a concrete individual name.
- $\{z : \mathsf{dom}(\mathcal{D}_1), z : \mathsf{dom}(\mathcal{D}_2)\}$.

- $\{z : A\}$ where A is a concept name.
- $\{s : \mathsf{dom}(\mathcal{D})\}$ where s is an abstract individual.
- $\{(x_1^{(1)}, \dots, x_{n_1}^{(1)}) : P_1, \dots, (x_1^{(k)}, \dots, x_{n_k}^{(k)}) : P_k\}$ and

$$\bigwedge_{i=1}^{k} P_i^{\mathcal{D}}(\bar{x}^{(i)}) \text{ is not satisfiable in a given concrete domain } \mathcal{D}.$$

We are able to detect this clash because we have supposed that conjunction of predicates over $\mathcal{D}$ is *decidable.*

In the following we state some properties of the calculus to assess the *decidability* of knowledge base satisfiability.

Proposition 1 (Invariance) *Suppose $\mathcal{KB}_G^{'}$ has been derived from $\mathcal{KB}_G$ by application of a rule. Then $\mathcal{KB}_G$ is satisfiable if and only if $\mathcal{KB}_G^{'}$ is satisfiable.*

Proof See Appendix.

Proposition 2 (Termination) *Let $\mathcal{KB}_G$ be a knowledge base. A completion of $\mathcal{KB}_G$ is finite.*

Proof The proof follows from the following arguments: The schema $\mathcal{S}$ is a finite set of axioms. $\mathcal{S}$ is acyclic. The *ABox* $\mathcal{A}$ is also a finite set of assertions. All rules are never applied more that once to the same assertion. ■

Let $\mathcal{KB}_G$ be a knowledge base. The *canonical* interpretation $\mathcal{I}_{\mathcal{KB}_G}$ is defined as follows:

- Because the clash rule related to the concrete domains is not applicable, there is an assignment α that *satisfies* the conjunction of all occurring constraints of the form $P(z_1, \dots, z_n)$. The interpretation $\mathcal{I}_{\mathcal{KB}_G}$ interprets a concrete individual z_i as $\alpha(z_i)$.
- $\Delta^{\mathcal{I}_{\mathcal{KB}_G}} := \{s \mid s \text{ is in } \mathcal{KB}_G\}$
- $A^{\mathcal{I}_{\mathcal{KB}_G}} := \{s \mid s : A \text{ is in } \mathcal{KB}_G\}$
- $P^{\mathcal{I}_{\mathcal{KB}_G}} := \{(s,t) \mid (s,t) : P \text{ is in } \mathcal{KB}_G\}$
- $f^{\mathcal{I}_{\mathcal{KB}_G}} := \{(s,w) \mid (s,w) : f \text{ is in } \mathcal{KB}_G\}$

Proposition 3 *Let $\mathcal{KB}_G^{+}$ be a complete knowledge base. $\mathcal{KB}_G^{+}$ is satisfiable iff it contains no clash.*

Proof See Appendix.

Theorem 1 *Let $\mathcal{KB}$ be a knowledge base. Checking whether $\mathcal{KB}$ is satisfiable is a decidable problem.*

Proof This follows from Propositions 3 and 2 and the fact that $\mathcal{KB}$ is satisfiable if and only if $\mathcal{KB}_G$ is satisfiable. ■

2.4 Query Language ($\mathcal{QL}$)

Querying a database means retrieving stored objects that satisfy certain *conditions* or *qualifications* and hence are interesting for a user. In the case of relational databases, queries are constructed by means of algebra expressions defined on relations from the database. As a property, answers are also relations (i.e., sets of tuples). This correspondence between database entities and answer formats presents advantages that lead to the design and development of query optimization techniques. In object-oriented databases, classes are used to represent sets of objects. By analogy with the relational approach, classes can be used for *describing query results*. If such a possibility exists, then we can consider some kind of *reasoning* on the structure[6] of classes that will lead to reveal, for example, *subsumption* relationships between queries.

In our framework, we follow this approach. Queries are represented as concepts in our abstract language.
In the following, we give the syntax and semantics of a concept language for making queries.

Definition 7 (Syntax) *Let A be a concept name, P be an atomic role, d be an abstract individual name, $f_1, f_2, \ldots, g_1, g_2, \ldots$ be feature names, $P_r \in \mathsf{pred}(\mathcal{D}_i)$ for some $i \in [1,k]$ be an n-ary predicate name, and $P_{r_j} \in \mathsf{pred}(\mathcal{D}_i)$ for some $i \in [1,k]$ ($j \in [1,m]$) be a binary predicate name. Concepts C, D and roles R, R' can be formed by means of the following syntax:*

$$\begin{aligned} C, D \longrightarrow\ & \top \mid A \mid C \sqcap D \mid \{d\} \mid \exists R.C \mid P_r(f_1, \ldots, f_n) \mid \\ & \Theta(C, D, \{\langle f_1, P_{r_1}, g_1\rangle, \ldots, \langle f_m, P_{r_m}, g_m\rangle\}) \\ R, R' \longrightarrow\ & P \mid P^- \mid R \circ R' \end{aligned}$$

Definition 8 (Semantics) *The semantics is given by an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, which consists of an (abstract) interpretation domain $\Delta^{\mathcal{I}}$, and an interpretation function $\cdot^{\mathcal{I}}$. The abstract domain has to be disjoint from any given concrete domain, i.e., $\Delta^{\mathcal{I}} \cap \mathsf{dom}(\mathcal{D}_i) = \emptyset$ for all concrete domain $\mathcal{D}_i$ ($i \in [1,k]$), the concrete domains are pairwise disjoint, and $\mathsf{pred}(\mathcal{D}_i) \cap \mathsf{pred}(\mathcal{D}_j) = \emptyset$ for $i \neq j$. The interpretation function $\cdot^{\mathcal{J}}$ associates each concept C with a subset $C^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$, each role P with a binary relation $P^{\mathcal{I}}$ on $\Delta^{\mathcal{I}}$, and each feature name f with a partial function $f^{\mathcal{I}} : \Delta^{\mathcal{I}} \to (\Delta^{\mathcal{I}} \cup (\bigcup_{i=1}^{k} \mathsf{dom}(\mathcal{D}_i)))$. Additionally, $\mathcal{I}$ has to satisfy the following equations:*

$$\begin{aligned} \top^{\mathcal{I}} &= \Delta^{\mathcal{I}} \\ (\exists R.C)^{\mathcal{I}} &= \{d \in \Delta^{\mathcal{I}} \mid \exists d' : (d, d') \in R^{\mathcal{I}} \wedge d' \in C^{\mathcal{I}}\} \\ P_r(f_1, \ldots, f_n)^{\mathcal{I}} &= \{d \in \Delta^{\mathcal{I}} \mid (f_1^{\mathcal{I}}(d), \ldots, f_n^{\mathcal{I}}(d)) \in P^{\mathcal{D}}\} \\ \{d\}^{\mathcal{I}} &= \{d^{\mathcal{I}}\} \\ (R \circ R')^{\mathcal{I}} &= \{(d, d') \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid \exists c \in \Delta^{\mathcal{I}} \text{ such that} \\ &\qquad (d, c) \in R^{\mathcal{I}} \wedge (c, d') \in R'^{\mathcal{I}}\} \end{aligned}$$

$$\begin{aligned} &(\Theta(C, D, \{\langle f_1, P_{r_1}, g_1\rangle, \ldots, \langle f_m, P_{r_m}, g_m\rangle\}))^{\mathcal{I}} = \\ &\quad \{d \in \Delta^{\mathcal{I}} \mid d \in C^{\mathcal{I}} \text{ and } \exists d'\ d' \in D^{\mathcal{I}} \text{ such that} \\ &\quad (f_1^{\mathcal{I}}(d), g_1^{\mathcal{I}}(d')) \in P_{r_1}^{\mathcal{D}_{P_{r_1}}} \wedge \ldots \wedge (f_m^{\mathcal{I}}(d), g_m^{\mathcal{I}}(d')) \in P_{r_m}^{\mathcal{D}_{P_{r_m}}}\} \end{aligned}$$

Example 3 *Consider the database schema of Figure 1. The query:*

$$\begin{aligned} &\Theta(\mathsf{Camping}, \{\mathsf{d_{example}}\}, \\ &\quad \{\langle \mathsf{image}, \mathsf{same-texture}, \mathsf{image}\rangle, \langle \mathsf{image}, \mathsf{same-color}, \mathsf{image}\rangle\}) \sqcap \\ &\qquad \exists \mathsf{in\_city}.(=_{\text{"Germany"}} (\mathsf{name}) \sqcap <_{100} (\mathsf{price})) \end{aligned}$$

would be "Find a set of camping in Germany, with a price below 100, and the picture (i.e., the filler of the image feature) of each camping object in the answer set is similar to the picture associated with the individual $d_{example}$ regarding texture and color". Here image *is an attribute which links an abstract individual to a picture in the Feature and Content Layer. The predicates* same-texture *and* same-color *belong to this layer.*

[6]And hence the semantics of class hierarchies.

2.5 Query Answering Algorithm

Let $\mathcal{KB}$ be a knowledge base. In the following, $\mathcal{KB}^+$ denotes the completion of $\mathcal{KB}$. The query answering algorithm is presented in Figure 2.

CompAns($\mathcal{KB}$, Q) =

case of Q:

$$
\begin{array}{ccl}
A & : & \text{return } \{t \mid t : A \in \mathcal{KB}^+\} \\
\{d\} & : & \text{return } \{d\} \\
Q_1 \sqcap Q_2 & : & \text{return CompAns}(\mathcal{KB},\, Q_1) \cap \text{CompAns}(\mathcal{KB},\, Q_2) \\
\exists R.Q' & : & \text{return } \{t \mid (\exists t')\, t' \in \mathsf{Eval}(R(t)) \text{ and } t' \in \text{CompAns}(\mathcal{KB}, Q')\} \\
P_r(f_1,\ldots,f_n) & : & \text{return } \{t \mid t : P_r(f_1,\ldots,f_n) \in \mathcal{KB} \text{ or} \\
 & & t : P_r'(f_1,\ldots,f_n) \in \mathcal{KB} \text{ and } P_r'(f_1,\ldots,f_n) \text{ entails } P_r(f_1,\ldots,f_n) \text{ or} \\
 & & \exists z_1,\ldots,z_n \text{ such that } \{f_1(t,z_1),\ldots,f_n(t,z_n)\} \subseteq \mathcal{KB} \text{ and } ((z_1,\ldots,z_n) \in P_r^{\mathcal{D}} \\
 & & \text{or } ((z_1,\ldots,z_n) \in P_r'^{\mathcal{D}} \text{ and } P_r'(f_1,\ldots,f_n) \text{ entails } P_r(f_1,\ldots,f_n)))\} \\
\Theta(Q_1,Q_2,\{\langle f_1,P_{r_1},g_1\rangle,\ldots,\langle f_m,P_{r_m},g_m\rangle\}) & : & \text{return } \{t \mid t \in \text{CompAns}(\mathcal{KB},\, Q_1) \text{ and } \exists t'\, t' \in \text{CompAns}(\mathcal{KB},\, Q_2) \text{ and} \\
 & & \exists z_1, z_1',\ldots,z_m,z_m' \text{ such that} \\
 & & \{f_1(t,z_1), g_1(t',z_1'),\ldots,f_m(t,z_m),g_m(t',z_m')\} \subseteq \mathcal{KB} \text{ and} \\
 & & (z_1,z_1') \in P_{r_1}^{\mathcal{D}},\ldots,(z_m,z_m') \in P_{r_m}^{\mathcal{D}}\}
\end{array}
$$

$$
\mathsf{Eval}(R(t)) = \begin{cases} \{t' \mid tRt' \in \mathcal{KB}\} & : \text{ if } R \text{ is a primitive role} \\ \{t' \mid t \in R'(t')\} & : \text{ if } R = R'^{-} \\ \{t' \mid t' \in \mathsf{Eval}(R''(\mathsf{Eval}(R'(t))))\} & : \text{ if } R = R' \circ R'' \end{cases}
$$

Figure 2: The Query Answering Algorithm.

3 A Calculus for Deciding Query Containment

In this section we provide a calculus for deciding the containment of a query in a view (which is a query as well). In particular we present the calculus and state its correctness and completeness.

Definition 9 (Containment) *Given a $\mathcal{SL}$ schema $\mathcal{S}$, a query Q and a view V in $\mathcal{QL}$ language, are the answers to Q also answers to V for any database state obeying the schema $\mathcal{S}$.*

Example 4 *Given the databse schema of Figure 1, we can easily notice that the query*

$\Theta(\mathsf{Amusement\_parc}, \mathsf{Beach}, \{\langle \mathsf{image}, \mathsf{similar-to}, \mathsf{image}\rangle\}) \sqcap \leq_{90} (\mathsf{price})$

is $\mathcal{S}$-contained in the view

$\mathsf{Site} \sqcap \leq_{100} (\mathsf{price}) \sqcap \exists \mathsf{image}.\mathtt{IMAGE}$

A query Q is $\mathcal{S}$-satisfiable if there is an $\mathcal{S}$-interpretation $\mathcal{I}$ such that $Q^{\mathcal{I}} \neq \emptyset$. We say that Q is $\mathcal{S}$-*contained* in V (written $Q \dot{\preceq}_{\mathcal{S}} V$) if $Q^{\mathcal{I}} \subseteq V^{\mathcal{I}}$ for every $\mathcal{S}$-interpretation $\mathcal{I}$.

The basic idea for deciding the *containment* of a query Q in a view V is drawn from [12]. We take an object o and transform Q into a prototypical database state where o is an answer to Q. We do so by generating individuals, entering them into concepts in the schema, and relating them through roles and features. If o belongs to the answer of V, then Q is contained in V. If not, we have a state where an individual is in the answer to Q but not in the answer to V and therefore V does not contain Q.

In the following, we make three assumptions:

- The schema $\mathcal{S}$ is acyclic.
- Given a concept name A and a role (or a feature) name R, we do not allow axioms of the form $A \dot{\preceq} \exists R.A', A \dot{\preceq} \exists R.A''$ in the schema $\mathcal{S}$, where A' and A'' are two different concept names.
- We do not allow sub-expressions of the form $\exists R.C_1 \sqcap \ldots \sqcap \exists R.C_n$ for the same role (or feature) name R in a query, but we allow sub-expressions of the form $\exists R.(C_1 \sqcap \ldots \sqcap C_n)$.

3.1 Propagation Rules

According to the syntax of our concept languages, concepts describing queries make reference (through roles and features) to abstract individual names and/or concrete individual names. In the following, a and b are abstract individual names, x, y are variables denoting abstract individuals. In the sequel, we refer to abstract individual names and variables as abstract individuals, denoted by the letters $s, t, s', t', s_1, t_1, \ldots$. As the unique name assumption does not hold for concrete individual names, we use $z, z', z_1, z_1^{'}, z_2, z_2^{'}, \ldots$ as names for concrete individuals. Finally, we use $v, v', v_1, v_1^{'}, \ldots$ to refer to abstract individual names, variables, or concrete individual names.

The calculus works on syntactic entities called *constraints* which are of the form:

$$s : C,\ (s,t) : R,\ (s,t) : f,\ (s,z) : f,\ (z_1, \ldots, z_n) : P_r,\ a \neq b$$

where s, t are abstract individuals, a, b are abstract individual names, $z, z_1, \ldots, z_n$ are concrete individual names, R is a role name, f is a feature names, P_r is an n-ary predicate name, and C is a $\mathcal{QL}$ concept or a $\mathcal{SL}$ concept name.

A *constraint system* $\tilde{S}$ is a finite set of constraints.

The semantics is extended to constraints. An interpretation $\mathcal{I}$ maps a variable x to an element $x^{\mathcal{I}}$ of the abstract domain $\Delta^{\mathcal{I}}$ of $\mathcal{I}$ and a concrete individual name z to an element of its concrete domain. An interpretation $\mathcal{I}$ satisfies a constraint

$$s : C \text{ iff } s^{\mathcal{I}} \in C^{\mathcal{I}},\ (s,t) : R \text{ iff } (s^{\mathcal{I}}, t^{\mathcal{I}}) \in R^{\mathcal{I}}$$

$$(s,t) : f \text{ iff } f^{\mathcal{I}}(s^{\mathcal{I}}) = t^{\mathcal{I}},\ a \neq b \text{ iff } a \neq b$$

$$(s,z) : f \text{ iff } f^{\mathcal{I}}(s^{\mathcal{I}}) = z^{\mathcal{I}},\ (z_1, \ldots, z_n) : P_r \text{ iff } (z_1^{\mathcal{I}}, \ldots, z_n^{\mathcal{I}}) \in P_r^{\mathcal{D}}$$

A constraint system $\tilde{S}$ is *satisfiable* if there is an interpretation $\mathcal{I}$ that satisfies every constraint in $\tilde{S}$.

Let c and c' be constraints and $\mathcal{S}$ be a schema. we write

$$c \models_{\mathcal{S}} c'$$

if every $\mathcal{S}$-model of c is also an $\mathcal{S}$-model of c'.

Proposition 4 *Let $\mathcal{S}$ be a schema, Q be a query and V be a view, and x' be a variable. Then*

$$Q \dot{\preceq}_{\mathcal{S}} V \ \text{ iff } \ x' : Q \models_{\mathcal{S}} x' : V$$

Proof

" $\Rightarrow$ "

If $Q\dot{\preceq}_{\mathcal{S}}V$ then $Q^{\mathcal{I}} \subseteq V^{\mathcal{I}}$ for all model $\mathcal{I}$ of $\mathcal{S}$. This means that if x' denotes an individual from the interpretation domain of $\mathcal{I}$ such that $x'^{\mathcal{I}} \in Q^{\mathcal{I}}$, then $x'^{\mathcal{I}} \in V^{\mathcal{I}}$. It follows that $x' : Q \models_{\mathcal{S}} x' : V$.

" $\Leftarrow$ "

Suppose that $x'^{\mathcal{I}} \in Q^{\mathcal{I}}$ implies $x'^{\mathcal{I}} \in V^{\mathcal{I}}$ for all $\mathcal{I}$ model of $\mathcal{S}$. It follows that $Q^{\mathcal{I}} \subseteq_{\mathcal{S}} V^{\mathcal{I}}$. Hence, $Q\dot{\preceq}_{\mathcal{S}}V$. ■

Hence, to test Q and V for *containment*, we have to check the constraints $x' : Q$ and $x' : V$ for *entailment.*

Let V be a view. We call the constraint $x' : V$ a goal.

As in [12], our method makes use of four kinds of rules: *decomposition*, *schema*, *goal*, and *composition* rules. Given a query Q and a view V, the rules work on pairs of constraint systems $\mathcal{Q}.\mathcal{V}$. We call $\mathcal{Q}$ (built from Q) the *facts* and $\mathcal{V}$ (built from V) the *goal.* To decide whether $Q\dot{\preceq}_{\mathcal{S}}V$, we take a variable x' and start with the facts $\{x' : Q\} \cup \{a \not\doteq b$ for all pairs of abstract individual names appearing in $Q\}$, and the goal $\{x' : V\}$.

The role of the propagation rules is to make explicit (by adding constraints to $\mathcal{Q}$ or $\mathcal{V}$) the part of the knowledge which is implicitly contained in $\mathcal{Q}$, $\mathcal{V}$ and the schema $\mathcal{S}$. They add facts and goals until no rule applies. Intuitively, the view V contains the query Q *if and only if* the final set of facts contains the constraint $x' : V$.

Given a pair of constraint systems $\mathcal{Q}.\mathcal{V}$, more than one rule might be applicable to it. We define the following strategy for the application of rules:

1. apply the decomposition rules as long as possible;
2. apply the schema rules as long as possible;
3. apply the goal rules as long as possible;
4. apply the composition rules.

Let R be a role name (resp. f be a feature name). In the following, we write equally sRt or $(s,t) : R$ (resp. sfv or $(s,v) : f$) to denote the fact that t is a R-successor (resp. v is an f-successor) of s.

Decomposition Rules

These rules add constraints to the constraint system $\mathcal{Q}$. They break up the initial fact $x' : Q$ into constraints involving primitive concepts.

D₁) $\mathcal{Q}.\mathcal{V} \quad \longrightarrow_{\sqcap} \quad \{s : C_1, s : C_2\} \cup \mathcal{Q}.\mathcal{V}$
    if $\quad s : C_1 \sqcap C_2$ is in $\mathcal{Q}$ and
    $s : C_1$ and $s : C_2$ are not both in $\mathcal{Q}$

D₂) $\mathcal{Q}.\mathcal{V} \quad \longrightarrow_{\exists} \quad \{sRy, y : C\} \cup \mathcal{Q}.\mathcal{V}$
    if $\quad s : \exists R.C$ is in $\mathcal{Q}$ and
    y is a new variable and
    there is no t such that t is an
    $R-$successor of s in $\mathcal{Q}$ and $t : C$ is in $\mathcal{Q}$

D₃) $\mathcal{Q}.\mathcal{V} \quad \longrightarrow_{P_r} \{s\, f_1\, z_1, \ldots, s\, f_n\, z_n, (z_1, \ldots, z_n) : P_r\} \cup \mathcal{Q}.\mathcal{V}$
if $s : P_r(f_1, \ldots, f_n)$ is in $\mathcal{Q}$, and $z_1, \ldots, z_n$
are new concrete individual names
and the following does not hold
For the features $f_i, i = 1, \ldots, n$, there are
concrete individual names $z_1^{'}, \ldots, z_n^{'}$ such that
$\mathcal{Q}$ contains constraints $s\, f_1\, z_1^{'}, \ldots, s\, f_n\, z_n^{'}$,
$(z_1^{'}, \ldots, z_n^{'}) : P_r$

We may have created forks by this rule. If this is the case, we delete them as described before.

D₄) $\mathcal{Q}.\mathcal{V} \quad \longrightarrow_{\Theta} \{s : C, y : C', s\, f_1\, z_1, \ldots, s\, f_m\, z_m,$
$y\, g_1\, z_1^{'}, \ldots, y\, g_m\, z_m^{'}, (z_1, z_1^{'}) : P_{r_1}, \ldots,$
$(z_m, z_m^{'}) : P_{r_m}\} \cup \mathcal{Q}.\mathcal{V}$
if $s : \Theta(C, C', \{\langle f_1, P_{r_1}, g_1\rangle, \ldots, \langle f_m, P_{r_m}, g_m\rangle\}$
is in $\mathcal{Q}$, and
$z_1, z_1^{'}, \ldots, z_m, z_m^{'}$ are new concrete individual
names and y is a new variable
and the following does not hold
there is an abstract individual s' and for the
features f_i $(i = 1, \ldots, m)$ and g_i $(i = 1, \ldots, m)$
there are concrete individual names $u_1, \ldots, u_m$,
$u_1^{'}, \ldots, u_m^{'}$ such that
$\mathcal{Q}$ contains constraints $s' : C', s\, f_1\, u_1, \ldots, s\, f_m\, u_m$,
$s'\, g_1\, u_1^{'}, \ldots, s'\, g_m\, u_m^{'}, (u_1, u_1^{'}) : P_{r_1}, \ldots, (u_m, u_m^{'}) : P_{r_m}$

We may have created forks by this rule. If this is the case, we delete them as described before.

D₅) $\mathcal{Q}.\mathcal{V} \quad \longrightarrow_{-} \quad \{tRs\} \cup \mathcal{Q}.\mathcal{V}$
if $sR^{-}t$ is in $\mathcal{Q}$ and tRs is not in $\mathcal{Q}$

D₆) $\mathcal{Q}.\mathcal{V} \quad \longrightarrow_{[]} \quad [y/a]\mathcal{Q}.\mathcal{V}[y/a]$
if $y : \{a\}$ is in $\mathcal{Q}$

D₆ is a substitution rule. We read $[y/a]$ as "a replaces y". This substitution applies to both constraint systems, i.e., $\mathcal{Q}$ and $\mathcal{V}$.

D₇) $\mathcal{Q}.\mathcal{V} \quad \longrightarrow_{\circ} \quad \{sRy, yR't\} \cup \mathcal{Q}.\mathcal{V}$
if $s\, R \circ R'\, t$ is in $\mathcal{Q}$ and
y is a new variable and
there is no t' such that $sRt', t'R't$ are in $\mathcal{Q}$

<u>Schema Rules</u>

These rules add constraints to the constraint system $\mathcal{Q}$. They add information derivable from the schema $\mathcal{S}$ and current facts contained in $\mathcal{Q}$.

S₁) $\mathcal{Q}.\mathcal{V} \quad \longrightarrow_{1.\dot{\preceq}} \quad \{s : A'\} \cup \mathcal{Q}.\mathcal{V}$
if $s : A$ is in $\mathcal{Q}, A \dot{\preceq} A'$ is in $\mathcal{S}$ and
$s : A'$ is not in $\mathcal{Q}$

S₂) $\mathcal{Q}.\mathcal{V} \quad \longrightarrow_{\forall} \quad \{v : A'\} \cup \mathcal{Q}.\mathcal{V}$
if $s : A, s\, P\, v$ are in $\mathcal{Q}, A \dot{\preceq} \forall P.A'$ is in $\mathcal{S}$
(respectively $s : A, sfv$ are in $\mathcal{Q}$ and $A \dot{\preceq} \forall f.A'$ is in $\mathcal{S}$)
and $v : A'$ is not in $\mathcal{Q}$, where
v is an abstract individual (resp. a concrete individual name)

<u>Goal Rules</u>

These rules add constraints to the constraint system $\mathcal{V}$. They guide the evaluation of V by deriving subgoals from the original goal $x' : V$.

G₁) $\mathcal{Q}.\mathcal{V} \longrightarrow_{\sqcap} \mathcal{Q}.\mathcal{V} \cup \{s : C_1, s : C_2\}$
if $s : C_1 \sqcap C_2$ is in $\mathcal{V}$ and
$s : C_1, s : C_2$ are not in $\mathcal{Q} \cup \mathcal{V}$

G₂) $\mathcal{Q}.\mathcal{V} \longrightarrow_{\exists} \mathcal{Q}.\mathcal{V} \cup \{t : C\}$
if $s : \exists R.C$ is in $\mathcal{V}$ and sRt is in $\mathcal{Q}$ and
$t : C$ is not in $\mathcal{Q} \cup \mathcal{V}$

Composition Rules

These rules add constraints to the constraint system $\mathcal{Q}$. They compose[7] complex facts from simpler ones directed by the goals.

C₁) $\mathcal{Q}.\mathcal{V} \longrightarrow_{\sqcap} \{s : C_1 \sqcap C_2\} \cup \mathcal{Q}.\mathcal{V}$
if $s : C_1, s : C_2$ are in $\mathcal{Q}$, and
$s : C_1 \sqcap C_2$ is in $\mathcal{V}$, but not in $\mathcal{Q}$

Let $\mathcal{D}$ be a concrete domain, $f_1, \ldots, f_n$ be feature names, and P_r, P'_r be predicate names in $\mathsf{pred}(\mathcal{D})$. $P'_r(f_1, \ldots, f_n)$ *entails* $P_r(f_1, \ldots, f_n)$ iff $\forall e_1, \ldots, e_n \in \mathsf{dom}(\mathcal{D}), (e_1, \ldots, e_n) \in {P'_r}^{\mathcal{D}} \Rightarrow (e_1, \ldots, e_n) \in {P_r}^{\mathcal{D}}$. We are able to decide this because we have supposed that the *implication* between finite conjunctions over $\mathsf{pred}(\mathcal{D})$ is *decidable.*

C₂) $\mathcal{Q}.\mathcal{V} \longrightarrow_{P_r} \{s : P_r(f_1, \ldots, f_n)\} \cup \mathcal{Q}.\mathcal{V}$
if $s : P'_r(f_1, \ldots, f_n)$ is in $\mathcal{Q}$ and
$s : P_r(f_1, \ldots, f_n)$ is in $\mathcal{V}$ but not in $\mathcal{Q}$, and
$P'_r(f_1, \ldots, f_n)$ entails $P_r(f_1, \ldots, f_n)$

C₃) $\mathcal{Q}.\mathcal{V} \longrightarrow_{\top} \{s : \top\} \cup \mathcal{Q}.\mathcal{V}$
if $s : \top$ is in $\mathcal{V}$ but not in $\mathcal{Q}$

C₄) $\mathcal{Q}.\mathcal{V} \longrightarrow_{\exists} \{s : \exists R.C\} \cup \mathcal{Q}.\mathcal{V}$
if $s : \exists R.C$ is in $\mathcal{V}$ but not in $\mathcal{Q}$ and
$sRt, t : C$ are in $\mathcal{Q}$

C₅) $\mathcal{Q}.\mathcal{V} \longrightarrow_{\Theta} \{s : \Theta(C, C', \{\langle f_1, P_{r_1}, g_1\rangle, \ldots, \langle f_m, P_{r_m}, g_m\rangle\})\} \cup \mathcal{Q}.\mathcal{V}$
if $s : C, t : C', s\, f_1\, z_1, \ldots, s\, f_m\, z_m, t\, g_1\, z'_1, \ldots, t\, g_m\, z'_m, (z_1, z'_1) : P'_{r_1}, \ldots, (z_m, z'_m) : P'_{r_m}$ are in $\mathcal{Q}$
and $P'_{r_1}(f_1, g_1)$ entails $P_{r_1}(f_1, g_1), \ldots,$
$P'_{r_m}(f_m, g_m)$ entails $P_{r_m}(f_m, g_m)$ and
$s : \Theta(C, C', \{\langle f_1, P_{r_1}, g_1\rangle, \ldots, \langle f_m, P_{r_m}, g_m\rangle\})$
is in $\mathcal{V}$ but not in $\mathcal{Q}$

All rules are deterministic. Moreover, rules $\mathbf{D_2}$, $\mathbf{D_3}$, $\mathbf{D_4}$, $\mathbf{D_7}$ are generating ones.

A constraint system $\tilde{S}$ is *complete* if no propagation rule applies to it. A constraint system contains a clash if it displays one of the following situations:

[7]This can be seen as a bottom up evaluation of V over $\mathcal{Q}$.

- It contains the constraints $(x_1^{(1)}, \ldots, x_{n_1}^{(1)}) : P_{r_1}$, …, $(x_1^{(k)}, \ldots, x_{n_k}^{(k)}) : P_{r_k}$ and

$$\bigwedge_{i=1}^{k} P_{r_i}^{\mathcal{D}}(\bar{x}^{(i)}) \text{ is not satisfiable in a concrete domain } \mathcal{D}.$$

- It contains the constraints $s : \{t\}, s \neq t$.
- It contains the constraints sft, sfz, where t is an abstract individual and z is a concrete individual name.
- It contains the constraint $s : \mathsf{dom}(\mathcal{D})$, where s is an abstract individual.
- It contains the constraint $z : A$ where z is a concrete individual name and A a concept name.

Therefore, any constraint system containing a clash is unsatisfiable.

In the following, x' is a variable, $\mathcal{F}_{\mathcal{Q}}.\mathcal{G}_{\mathcal{V}}$ is the *completion* of $\{x' : Q\}.\{x' : V\}$, and o is an abstract individual name such that $o : V$ is in $\mathcal{G}_{\mathcal{V}}$.

Let $\mathcal{F}_{\mathcal{Q}}.\mathcal{G}_{\mathcal{V}}$ be a complete pair derivable from an initial pair $\{x' : Q\}.\{x' : V\}$. By construction, $\mathcal{G}_{\mathcal{V}}$ contains exactly one constraint of the form $s : V$. In addition, as goal rules are not generating ones and by examining all other rules, we observe that if $s : V \in \mathcal{G}_{\mathcal{V}}$ then $s : Q \in \mathcal{F}_{\mathcal{Q}}$.

Proposition 5 (Invariance) *Suppose $\mathcal{F}.\mathcal{G}$ has been derived from $\{x' : Q\}.\{x' : V\}$, and $\mathcal{F}'.\mathcal{G}'$ is obtained from $\mathcal{F}.\mathcal{G}$ by applying a rule. Then $\mathcal{F}$ is satisfiable if and only if $\mathcal{F}'$ is satisfiable.*

Proof See Appendix.

Corollary 1 *Every $\mathcal{S}$-model $\mathcal{I}$ of $x' : Q$ can be turned into an $\mathcal{S}$-model $\mathcal{I}'$ of $\mathcal{F}_{\mathcal{Q}}$ by modifying the interpretation of variables and concrete individual names. Moreover, $\mathcal{I}'$ can be chosen such that $o^{\mathcal{I}'} = x'^{\mathcal{I}}$.*

Proof It follows by induction from the preceding Proposition. ■

Corollary 2 *Let $\mathcal{F}_{\mathcal{Q}}$ be the complete constraint system derived from $\{x' : Q\}$, and let o be an abstract individual name. The following holds:*

$$x' : Q \models_{\mathcal{S}} x' : V \Leftrightarrow \mathcal{F}_{\mathcal{Q}} \models_{\mathcal{S}} o : V$$

Proof See Appendix.

Let $\tilde{S}$ be a clash-free constraint system. We define the canonical interpretation $\mathcal{I}_{\tilde{S}}$ as follows:

- Because the clash rule related to concrete domains is not applicable, there is an assignment α that *satisfies* the conjunction of all occurring constraints of the form $P_r(z_1, \ldots, z_n)$. The interpretation $\mathcal{I}_{\tilde{S}}$ interprets a concrete individual name z as $\alpha(z)$.
- the domain $\Delta^{\mathcal{I}_{\tilde{S}}}$ consists of all abstract individuals occurring in $\tilde{S}$.
- Let A be a primitive concept name. Then we set $s \in A^{\mathcal{I}}$ iff $s : A$ occurs in $\tilde{S}$.
- Let R be a role or a feature name. Then we set $(s, v) \in R^{\mathcal{I}_{\tilde{S}}}$ iff $(a, v) : R$ occurs in $\tilde{S}$. This is well defined even if R is a feature, because there is no clash related to the features. Here v is an abstract individual or a concrete individual name.

Proposition 6 *Let $\mathcal{F}_Q.\mathcal{G}_\mathcal{V}$ be a complete pair that has been derived from $\{x' : Q\}.\{x' : V\}$. If $\mathcal{F}_Q$ is clash-free, then the canonical interpretation $\mathcal{I}_{\mathcal{F}_Q}$ is an $\mathcal{S}$-model of $\mathcal{F}_Q$.*

Proof See Appendix.

Proposition 7 *Let $\mathcal{I}_{\mathcal{F}_Q}$ be the canonical interpretation of $\mathcal{F}_Q$ and $s : C$ be a constraint in $\mathcal{G}_\mathcal{V}$. If $\mathcal{F}_Q$ is clash-free, then*

$$\mathcal{I}_{\mathcal{F}_Q} \text{ satisfies } s : C \Longrightarrow s : C \in \mathcal{F}_Q$$

Proof See Appendix.

Definition 10 (Size of a concept) *For a concept C, the size $|C|$ is inductively defined as:*

- $|P_r(f_1, \dots, f_n)| = n + 1$ for all n-ary predicates of the concrete domains and features $f_1, \dots, f_n$.
- $|\Theta(C, C', \{\langle f_1, P_{r_1}, g_1\rangle, \dots, \langle f_m, P_{r_m}, g_m\rangle\})| = 2m + |C| + |D|$.
- $|A| = 1$.
- $|\exists R.C| = 1 + |C|$.
- $|\{d\}| = 1$.
- $|C \sqcap D| = |C| + |D|$.

The size of the schema $\mathcal{S}$, noted $|\mathcal{S}|$, is given by the number of axioms in $\mathcal{S}$.

Proposition 8 (Termination) *Let Q and V be a query and a view respectively. Then there is no infinite chain of completion steps issuing from $\{x' : Q\}$ and $\{x' : V\}$.*

Proof See Appendix.

Theorem 2 (Soundness and Completeness)

$$Q \dot{\preceq}_\mathcal{S} V \text{ iff } o : V \in \mathcal{F}_Q \text{ or } \mathcal{F}_Q \text{ contains a clash}$$

Proof See Appendix.

Now we turn to the complexity of deciding $\mathcal{S}$-containment.

Proposition 9 (Number of individuals) *The number of individuals occurring in $\mathcal{F}_Q.\mathcal{G}_\mathcal{V}$ is at most $|Q| + |V|$.*

Proof Any constant in the pair $\mathcal{F}_Q.\mathcal{G}_\mathcal{V}$ must appear in Q or V. The goal and composition rules are not generating ones. The number of variables generated by the decomposition rules is finite and bounded by the size of Q. Hence, the number of constants plus the number of variables generated by the decomposition rules is less or equal to $|Q| + |V|$. The other rules are not generating ones. ■

Theorem 3 *Containment between concepts in our query language can be decided in time polynomial to the size of Q, V and $\mathcal{S}$.*

Proof The propagation rules can be divided in two categories: rules that add constraints and rules that reduce the number of variables (i.e., $\mathbf{D_6}$). The number of application of rules that reduce the number of variables is finite and bounded by the size of Q. The number of application of the other decomposition rules is finite and also bounded by the size of Q. The number of application of the goal rules is finite and bounded by the size of V. The number of application of the composition rules is finite and bounded by the size of V. The number of application of the schema rules is finite and bounded by $(|Q| + |V|).|\mathcal{S}|$. ■

4 Related Work

Our work relates to several fields of research in databases and Artificial Intelligence. We shortly discuss the relationship to modeling and retrieving image data by content, multimedia databases and query optimization.

Modeling and Retrieving Image Data by Content[8]**.** Modeling and retrieving image data by content has been considered from both database and artificial intelligence points of views. Meghini *et al.* [26] have investigated the use of a description logic as a conceptual tool for modeling and querying image data. Their language is a fragment of $\mathcal{ALC}$[29] extended to accommodate fuzzy aspects. The visual part in a query is captured through a mechanism of procedural attachment which is a king of logical interface between the conceptual part and the visual part of an image. The problem with this language is that subsumption between concepts is $PSPACE$-complete. In addition, They do not consider predicate restrictions over concrete domains. Hsu *et al.* [22] proposed a knowledge-based approach for retrieving images by content. The knowledge-based query processing is based on a query relaxation technique which exploits a Type Abstraction Hierarchy of image features. The query language is an extension of OQL[13] to include specific predicates (e.g., *similar-to* predicates). An interesting direction is to investigate a concept language for expressing queries of [22] and adapting the approach presented in [12] for testing query containment.

Multimedia. Goble *et al.* [21] proposed a description logic, called, GRAIL, for describing the image and video semantic content. A set of dedicated constructors are used to capture the structural part of these media objects. The aim is to support the coherent and incremental development of a coarse index on the semantic annotations of media documents. Lambrix and Padgham [23] described an extended description logic for representing and retrieving documents. The description logic includes part-of relations and allows for ordering information between the parts.

In these two proposals, the underlying query languages support only queries based on the structure of the documents (i.e., conceptual queries). None of them supports visual queries. Together with [26] they do not take into account predicate restrictions over concrete domains, which are extremely useful when querying multimedia repositories. In addition, they did not address the questions of decidability and complexity of reasoning services in their languages.

Query Optimization in Multimedia databases. The problem of optimizing queries over multimedia repositories has been addressed in recent works (see, among others, [14]). In summary, these works consider the indexes used to search the repository and user-defined filter conditions to define an execution space that is search-minimal. Semantic query optimization considers semantic knowledge for constructing query evaluation plans, and the framework for testing query containment presented in this paper is relevant due to the incorporation of schema knowledge in our algorithm. In multimedia applications where meta-data play an important role [31], these two kinds of query optimization have to cohabit.

5 Conclusion

There is now intense interest in multimedia systems. These interests span across vast areas in computer science, such as computer networks, databases, distributed computing, data compression, document processing, user interfaces, artificial intelligence, etc. In the long run, we expect that intelligent-solving systems will access information stored in a variety of formats, on a wide variety of media.

[8]Due to space limitation, we chose to not overview all the fascinating aspects of database support for image data.

Multimedia information is inherently complex. Traditional database techniques do not apply since, for example, they do not deal with content-based retrieval. We believe that the combination of database techniques and intelligent information retrieval will contribute to the realization of intelligent multimedia systems. Artificial intelligence, and more specifically knowledge representation, will play an important role in this task.

Our work focuses on a fundamental problem, namely, a content-based retrieval of image data. We have merely laid a formal and flexible framework which is appropriate for modeling and reasoning about meta-data and queries in image databases. Expressiveness and services of the meta data schema are crucial for image database quality[9]. The framework is general in that little needs to be changed when making extensions or taking other constructors for the abstract languages. In addition, this framework is appropriate for supporting semantic indexing [30], conceptual queries [36] and intensional queries [8]. Indeed, as the information structure that is contained in image databases is usually complicated and amount of information is huge, users may prefer to express queries with more general and abstract information instead of primitive terms directly based on the data stored in a database.

There are many interesting directions to pursue. **(1)** An important direction of active research is to significantly extend this framework to support part-whole relations. The result reported in [3] constitutes a nice basis; **(2)** Due to the visual nature of the data, a user may be interested in results that are similar to the query, thus, the query system should be able to perform exact as well as partial or fuzzy matching.

We are investigating these important research directions.

References

[1] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases.* Addison-Wesley, 1995.

[2] Timothy Arndt and Angela Guercio. An Object-Oriented Multimedia Database System with Versioning and Content-Based Retrieval. In Wayne Niblack and Ramesh C. Jain, editors, *Storage and retrieval for image and video database III (SPIE'95), San Jose, California*, pages 340 – 351, February 1995.

[3] Alessandro Artale, Enrico Franconi, Nicola Guarino, and Luca Pazzi. Part-Whole Relations in Object-Centered Systems: An Overview. *Data & Knowledge Engineering*, 20(3):347–383, December 1996.

[4] Franz Baader, Martin Bucheit, Manfred Jeusfeld, and Werner Nutt. Reasoning about Structured Objects: Knowledge Representation meets Databases. In F. Baader, M. Bucheit, M. Jeusfeld, and W. Nutt, editors, *proc*, D-94-11 in DFKI Documents, URL: http://SunSite.Informatik.RWTH-Aachen.DE/Publications/CEUR-WS/Vol-1/, September 1994. CEUR. 2p.

[5] Franz Baader and Philipp Hanschke. A Scheme for Integrating Concrete Domains into Concept Languages. In *Proceedings of the 12th International Joint Conference on Artificial Intelligent (IJCAI'91). Sydney, Australia*, pages 452–457, 1991.

[6] Catriel Beeri, Alon Y. Levy, and Marie-Christine Rousset. Rewriting Queries Using Views in Description Logics. In *Proceedings of the 1997 Symposium on Principles of Database Systems (PODS'97), Tucson, Arizona, USA*, pages 99–108, 1997.

[7] Riccardo Benedetti and Jean-Jacques Risler. *Real Algebraic and Semi-Algebraic Sets.* Hermann, editeurs des sciences et des arts, 293 rue Lecourbe, 75015 Paris, 1990. 340 pages.

[8] Sonia Bergamaschi, Claudio Sartori, and Maurizio Vincini. Description Logic Techniques for Intensional Query Answering in OODBs. In Franz Baader, Martin Buchheit, Manfred Jeusfeld, and Werner Nutt, editors, *Proceedings of the 2nd Workshop on Knowledge Representation meets DataBases (KRDB'95), Bielefeld, Germany*, September 1995.

[9] Ronard J. Brachman and James G. Schmolze. An Overview of the KL-ONE Knowledge Representation System. *Cognitive Science*, 9(2):171–216, 1985.

[10] P. Bresciani. Some Research Trends in KR&DB (position paper). In F. Baader, M. Bucheit, M. Jeusfeld, and W. Nutt, editors, *proc*, pages 1–3, URL: http://SunSite.Informatik.RWTH-Aachen.DE/Publications/CEUR-WS/Vol-4/, August 1996. CEUR.

[9]Quality implies accessibility, optimization, validation, etc.

[11] Martin Buchheit, Francesco M. Donini, and Andrea Schaerf. Decidable Reasoning in Terminological Knowledge Representation Systems. *Journal of Artificial Intelligence Research*, (1):109–138, 1993.

[12] Martin Buchheit, Manfred A. Jeusfeld, Werner Nutt, and Martin Staudt. Subsumption Between Queries to Object-Oriented Databases. In *Proceedings of the 4th International Conference on Extending Database Technology (EDBT'94), Cambridge, UK*, March 1994. (Also in Information Systems 19(1), pp. 33-54, 1994).

[13] Rick Cattel. *The Object Database Standard: ODMG-93.* Morgan Kaufmann, San Mateo, CA, 1994.

[14] Surajit Chaudhuri and Luis Gravano. Optimizing Queries over Multimedia Repositories. In H.V. Jagadish and Inderpal Singh Mumick, editors, *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data (SIGMOD'96), Montréal, Québec, Canada,*, pages 91–102, June 1996.

[15] Wesley W. Chu, Alfonso F. Cárdinas, and Ricky K. Taira. Knowledge-Based Image Retrieval with Spatial and Temporal Constructs. In Zbigniew W. Raś and Andrzej Skowron, editors, *Proceedings of the 10th International Symposium on Methodologies for Intelligent Systems (ISMIS'97), Charlotte, North Carolina, USA*, LNAI - 1325, pages 17–34. Springer, October 1997.

[16] Tzi cker Chiueh. Content-Based Image Indexing. In Jorge Bocca, Matthias Jarke, and Carlo Zaniolo, editors, *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB'94), Santiago, Chile*, pages 582–593, September 1994.

[17] Cyril Decleir, Mohand-Saïd Hacid, and Jacques Kouloumdjian. Modeling and Querying Video Data: A Hybrid Approach. In *Proceedings of the IEEE Workshop on Content-Based Access of Image & Video Libraries (CBAIVL'98), Santa Barbara, CA, USA*, pages 86–90. IEEE Computer Society, June 1998.

[18] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, and Werner Nutt. The Complexity of Concept Languages. Technical Report RR-95-07, Deutsches Forschunggszentrum für Künstliche Intelligenz (DFKI), Kaiserslautern, Germany, June 1995.

[19] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, and Andrea Schaerf. Reasoning in Description Logics. In *Foundation of Knowledge Representation.* Cambrige University Press, 1995.

[20] Melvin Fitting. *First-Order Logic and Automated Theorem Proving.* Springer-Verlag, 1990.

[21] C. A. Goble, C. Haul, and S. Bechhofer. Describing and Classifying Multimedia Using the Description Logic GRAIL. In Ishwar K. Sethi and Ramesh C. Jain, editors, *Storage and retrieval for image and video database IV (SPIE'96), San Jose, California*, pages 132–143, February 1996.

[22] Chih-Cheng Hsu, Wesley W. Chu, and Ricky K. Taira. A Knowledge-Based Approach for Retrieving Images by Content. *IEEE Transactions on Knowledge and Data Engineering*, August 1996.

[23] Patrick Lambrix and Lin Padgham. A Description Logic Model for Querying Knowledge Bases for Structured Documents. In Zbigniew W. Raś and Andrzej Skowron, editors, *Proceedings of the 10th International Symposium on methodologies for Intelligent Systems (ISMIS'97), Charlotte, North Carolina, USA*, LNAI 1325, pages 72–83. Springer, October 1997.

[24] Alon Y. Levy, Alberto O. Mendelzon, Yehoshua Sagiv, and Divesh Srivastava. Answering Queries Using Views. In *Proceedings of the 1995 Symposium on Principles of Database Systems (PODS'95), San Jose, CA, USA*, pages 95–104, May 1995.

[25] Carlo Meghini. Towards a Logical Reconstruction of Image Retrieval. In Ishwar K. Sethi and Ramesh C. Jain, editors, *Storage and retrieval for image and video database IV (SPIE'96), San Jose, California*, pages 108–119, February 1996.

[26] Carlo Meghini, Fabrizio Sebastiani, and Umberto Straccia. The Terminological Image Retrieval Model. In Alberto Del Bimbo, editor, *Proceedings of ICIAP'97, 9th International Conference On Image Analysis And Processing*, volume II, pages 156–163, Florence, I, September 1997.

[27] Bernhard Nebel. *Reasoning and Revision in Hybrid Representation Systems.* LNCS-422. Springer-Verlag, 1990.

[28] Roger C. Schank and Christopher K. Riesbeck. *Inside Conputer Understanding: Five Programs Plus Miniatures.* Lawrence Erlbaum & Associates, Millsdale, N.J., USA, 1981.

[29] Manfred Schmidt-Schauß and Gert Smolka. Attributive Concept Descriptions with Complements. *Artificial Intelligence*, 48(1):1–26, 1991.

[30] Albrecht Schmiedel. Semantic Indexing Based on Description Logics. In Franz Baader, Martin Buchheit, Manfred Jeusfeld, and Werner Nutt, editors, *Proceedings of the 1st Worshop on Knowledge Representation meets DataBases (KRDB'94), Saarbrücken, Germany*, September 1994.

[31] Amit Sheth and Wolfgang Klas. *Multimedia Data Management: Using Metadata to Integrate and Apply Digital Media.* Mc Graw Hill, 1998.

[32] John R. Smith and Shih-Fu Chang. An Image and Video Search Engine for the World-Wide Web. In Ishwar K. Sethi and Ramesh C. Jain, editors, *Proceedings of the Storage and Retrieval for Image and Video Databases V, San Jose, California, USA*, volume 3022, pages 84–95, February 1997.

[33] Roger W. Smith, Dorota Kieronska, and Svetha Venkatesh. Media-Independent Knowledge Representation via UMART: Unified Mental Annotation and Retrieval Tool. In Ishwar K. Sethi and Ramesh C. Jain, editors, *Proceedings of the Storage and Retrieval for Image and Video Databases IV, San Jose, California USA*, volume 2670, pages 96–107, February 1996.

[34] Divesh Srivastava, Shaul Dar, H. V. Jagadish, and Alon Y. Levy. Answering Queries with Aggregation Using Views. In *Proceedings of the 22nd International Conference on Very Large Databases (VLDB'96), Bombay, India*, September 1996.

[35] Jeffrey D. Ullman. *Principles of Database and Knowledge-Base Systems*, volume I and II. Computer Science Press, 1989.

[36] S.C. Yoon. Towards Conceptual Query Answering. In Zbigniew W. Raś and Andrzej Skowron, editors, *Proceedings of the 10th International Symposium on methodologies for Intelligent Systems (ISMIS'97), Charlotte, North Carolina, USA*, LNAI 1325, pages 187–196. Springer, October 1997.

Appendix

In this Appendix we give the proofs of some of the results stated in the previous sections.

Proof (Proposition 1)
" $\Leftarrow$ "
Propagation rules add (and never remove) assertions to a knowledge base. If $\mathcal{KB}_G^{'}$ is obtained from $\mathcal{KB}_G$ by applying a rule, then $\mathcal{KB}_G^{'}$ contains $\mathcal{KB}_G$. It follows that if $\mathcal{I}$ is a model for $\mathcal{KB}_G^{'}$, it is also a model for $\mathcal{KB}_G$.

" $\Rightarrow$ " The proof is by case analysis.
Let $\mathcal{I}$ be a model for $\mathcal{KB}_G$. Then $\mathcal{I}$ can be turned into a model of $\mathcal{KB}_G^{'}$ by modifying the interpretation of fresh variables, or new concrete individual names. We have to consider all the rules.

$\longrightarrow_{\dot\preceq}$) $\mathcal{I}$ satisfies $s : A$. That is $s^{\mathcal{I}} \in A^{\mathcal{I}}$ and as $\mathcal{I}$ is a model for $\mathcal{KB}_G$ it also satisfies every assertion of the form $A \dot\preceq C$ in $\mathcal{KB}_G$. Hence $s^{\mathcal{I}} \in C^{\mathcal{I}}$. That is, $\mathcal{I}$ satisfies the assertion $s : C$. It follows that $\mathcal{I}$ is a model for $\mathcal{KB}_G^{'}$.

$\longrightarrow_{1.\dot\preceq}$) $\mathcal{I}$ satisfies sPt. As $\mathcal{I}$ is a model for $\mathcal{KB}_G$ it also satisfies every assertion of the form $P \dot\preceq A_1 \times A_2$ in $\mathcal{KB}_G$. Hence $s^{\mathcal{I}} \in {A_1}^{\mathcal{I}}$ and $t^{\mathcal{I}} \in {A_2}^{\mathcal{I}}$. That is, $\mathcal{I}$ satisfies the assertions $s : A_1, t : A_2$. It follows that $\mathcal{I}$ is a model for $\mathcal{KB}_G^{'}$.

$\longrightarrow_{2.\dot\preceq}$) A similar reasoning as for $\longrightarrow_{1.\dot\preceq}$ can be used.

$\longrightarrow_{1.\forall}$) $\mathcal{I}$ satisfies $s : \forall P.A$ and sPt. Then, $t^{\mathcal{I}} \in A^{\mathcal{I}}$, i.e., $\mathcal{I}$ satisfies $t : A$. It follows that $\mathcal{I}$ is a model for $\mathcal{KB}_G^{'}$.

$\longrightarrow_{2.\forall}$) A similar reasoning as for $\longrightarrow_{1.\forall}$ can be used.

$\longrightarrow_{P_r}$) $\mathcal{I}$ satisfies $s : P_r(f_1, \ldots, f_n)$. That is, if z_1, …, z_n are concrete individual names such that $f_1^{\mathcal{I}}(s^{\mathcal{I}}) = z_1^{\mathcal{I}}$, …, $f_n^{\mathcal{I}}(s^{\mathcal{I}}) = z_n^{\mathcal{I}}$ we have $(z_1^{\mathcal{I}}, \ldots, z_n^{\mathcal{I}}) \in {P_r}^{\mathcal{D}}$. It follows that $\mathcal{I}$ is a model for $\mathcal{KB}_G^{'}$.

■

Proof (Proposition 3)
" $\Rightarrow$ " Clearly, a knowledge base containing a clash is unsatisfiable. Hence, $\mathcal{KB}_G^{+}$ is satisfiable only if it contains no clash.

" $\Leftarrow$ " Suppose $\mathcal{KB}_G^{+}$ contains no clash. Let $\mathcal{I}_{\mathcal{KB}_G^{+}}$ be the canonical interpretation for $\mathcal{KB}_G^{+}$. We have to prove that $\mathcal{I}_{\mathcal{KB}_G^{+}}$ satisfies every assertion in $\mathcal{KB}_G^{+}$. By definition $\mathcal{I}_{\mathcal{KB}_G^{+}}$ satisfies all the assertions of the form $s\,f\,w, sPt$ and $a \not\doteq b$ (a and b are two abstract individual names). If the assertion is of the form $s : C$, we show by induction on the structure of C that $s^{\mathcal{I}} \in C^{\mathcal{I}_{\mathcal{KB}_G^{+}}}$.

- C has the form $\forall P.A$. If $s : \forall P.A$ is in $\mathcal{KB}_G^{+}$, then if $\mathcal{KB}_G^{+}$ contains sPt for some t, then as it is complete it contains also $t : A$ since otherwise rule $\longrightarrow_{1.\forall}$ would be applicable. It follows that $s^{\mathcal{I}} \in (\forall P.A)^{\mathcal{I}}$.
- C has the form $\forall f.A$. Same reasoning like before.
- C has the form $P_r(f_1, \ldots, f_n)$. If $s : P_r(f_1, \ldots, f_n)$ is in $\mathcal{KB}_G^{+}$, then as $\mathcal{KB}_G^{+}$ is complete it contains also the assertions $s\,f_1\,z_1$, …, $s\,f_n\,z_n$, $(z_1, \ldots, z_n) : P_r$ since otherwise rule $\longrightarrow_{P_r}$ would be applicable. As $\mathcal{KB}_G^{+}$ does not contain a clash related to concrete domains, it follows that $s^{\mathcal{I}} \in P_r(f_1, \ldots, f_n)^{\mathcal{I}_{\mathcal{KB}_G^{+}}}$.

■

Proof (Proposition 5)

The proof is by case analysis.

" $\Rightarrow$ "

Let $\mathcal{I}$ be an $\mathcal{S}$-model of $\mathcal{F}$. Then $\mathcal{I}$ can be turned into an $\mathcal{S}$-model of $\mathcal{F}'$ by modifying the interpretation of new variables and new concrete individual names. We have to consider all the rules that alter the set of facts (i.e., decomposition, schema, and composition rules).

- **D₁**) $\mathcal{I}$ satisfies $s : C_1 \sqcap C_2$. That is $s^{\mathcal{I}} \in (C_1 \sqcap C_2)^{\mathcal{I}}$. This means that $s^{\mathcal{I}} \in (C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}})$, and then $s^{\mathcal{I}} \in C_1^{\mathcal{I}}$ and $s^{\mathcal{I}} \in C_2^{\mathcal{I}}$. Hence $\mathcal{I}$ satisfies $s : C_1$ and $s : C_2$. It follows that $\mathcal{I}$ is an $\mathcal{S}$-model for $\mathcal{F}'$.
- **D₂**) $\mathcal{I}$ satisfies $s : \exists R.C$. There exists an abstract individual t such that t is an R-successor of s and $t^{\mathcal{I}} \in C^{\mathcal{I}}$. It follows that $\mathcal{I}$ is an $\mathcal{S}$-model for $\mathcal{F}'$.
- **D₃**) $\mathcal{I}$ satisfies $s : P_r(f_1, \ldots, f_n)$. That is, if $z_1, \ldots, z_n$ are concrete individual names such that $f_1^{\mathcal{I}}(s^{\mathcal{I}}) = z_1^{\mathcal{I}}, \ldots, f_n^{\mathcal{I}}(s^{\mathcal{I}}) = z_n^{\mathcal{I}}$ we have $(z_1^{\mathcal{I}}, \ldots, z_n^{\mathcal{I}}) \in {P_r}^{\mathcal{D}}$. It follows that $\mathcal{I}$ is an $\mathcal{S}$-model for $\mathcal{F}'$.
- **D₄**) $\mathcal{I}$ satisfies $s : \Theta(C, C', \{\langle f_1, P_{r_1}, g_1\rangle, \ldots, \langle f_m, P_{r_m}, g_m\rangle\})$. That is, $s^{\mathcal{I}} \in C^{\mathcal{I}}$ and there exists an abstract individual s' and concrete individual names $z_1, \ldots, z_m, z_1', \ldots, z_m'$ such that $f_1^{\mathcal{I}}(s^{\mathcal{I}}) = z_1^{\mathcal{I}}$, …, $f_m^{\mathcal{I}}(s^{\mathcal{I}}) = z_m^{\mathcal{I}}$, $g_1^{\mathcal{I}}(s'^{\mathcal{I}}) = {z_1'}^{\mathcal{I}}$, …, $g_m^{\mathcal{I}}(s'^{\mathcal{I}}) = {z_m'}^{\mathcal{I}}$ and $s'^{\mathcal{I}} \in C'^{\mathcal{I}}$, $(z_1^{\mathcal{I}}, {z_1'}^{\mathcal{I}}) \in P_{r_1}^{\mathcal{D}}$, …, $(z_m^{\mathcal{I}}, {z_m'}^{\mathcal{I}}) \in P_{r_m}^{\mathcal{D}}$. It follows that $\mathcal{I}$ is an $\mathcal{S}$-model for $\mathcal{F}'$.
- **D₅**) $\mathcal{I}$ satisfies $sR^{-}t$. Then by definition it satisfies tRs. Hence $\mathcal{I}$ is an $\mathcal{S}$-model for $\mathcal{F}'$.
- **D₆**) Obvious.
- **D₇**) Obvious.
- **C₁**) It follows from **D₁**.
- **C₂**) $\mathcal{I}$ satisfies $s : P_r'(f_1, \ldots, f_n)$. That is, there exist concrete individual names $z_1, \ldots, z_n$ such that $f_1^{\mathcal{I}}(s^{\mathcal{I}}) = z_1^{\mathcal{I}}$, …, $f_n^{\mathcal{I}}(s^{\mathcal{I}}) = z_n^{\mathcal{I}}$ and $(z_1^{\mathcal{I}}, \ldots, z_n^{\mathcal{I}}) \in {P_r'}^{\mathcal{D}}$. It follows that $(z_1^{\mathcal{I}}, \ldots, z_n^{\mathcal{I}}) \in {P_r}^{\mathcal{D}}$ for all P_r such that $P_r'(f_1, \ldots, f_n)$ *entails* $P_r(f_1, \ldots, f_n)$. Hence $\mathcal{I}$ is an $\mathcal{S}$-model for $\mathcal{F}'$.
- **C₃**) Goal rules are not generating ones. Hence if $s : \top$ is in $\mathcal{V}$ then s appears in $\mathcal{Q}$, and then in $\mathcal{F}'$. As all abstract individuals are instances of $\top$ it follows that $\mathcal{I}$ is an $\mathcal{S}$-model for $\mathcal{F}'$.
- **C₄**) $\mathcal{I}$ satisfies sRt and $t : C$. By definition it satisfies $s : \exists R.C$. It follows that $\mathcal{I}$ is an $\mathcal{S}$-model for $\mathcal{F}'$.
- **C₅**) $\mathcal{I}$ satisfies $s : C$, $s' : C'$, $s\, f_1\, z_1$, …, $s\, f_m\, z_m$, $s'\, g_1\, z_1'$, …, $s'\, g_m\, z_m'$, $(z_1, z_1') : P_{r_1}$, …, $(z_m, z_m') : P_{r_m}$ for some abstract individual s' and concrete individual names z_1, …, z_m, z_1', …, z_m'. Then by definition, $\mathcal{I}$ satisfies $s : \Theta(C, C', \langle f_1, P_{r_1}, g_1\rangle, \ldots, \langle f_m, P_{r_m}, g_m\rangle\})$. It follows that $\mathcal{I}$ is an $\mathcal{S}$-model for $\mathcal{F}'$.
- **S₁**) $\mathcal{I}$ satisfies $s : A$. As $\mathcal{I}$ is an $\mathcal{S}$-model of $\mathcal{F}$, it satisfies all the axioms in $\mathcal{S}$. Hence $\mathcal{I}$ satisfies $A \dot{\preceq} A'$. That is, $s^{\mathcal{I}} \in A'^{\mathcal{I}}$. It follows that $\mathcal{I}$ is an $\mathcal{S}$-model for $\mathcal{F}'$.
- **S₂**) Obvious.

" $\Leftarrow$ "

The propagation rules add (and never remove) constraints to (from) a constraint system. If $\mathcal{F}'$ is obtained from $\mathcal{F}$ by applying a rule, then $\mathcal{F}'$ is a superset of $\mathcal{F}$. It follows that if $\mathcal{I}$ is an $\mathcal{S}$-model for $\mathcal{F}'$, it is also an $\mathcal{S}$-model for $\mathcal{F}$. ■

Proof (Corollary 2)

First, note that by examining all the rules, we remark that if $s : V$ is in $\mathcal{G}_{\mathcal{V}}$ then $s : Q$ is in $\mathcal{F}_{\mathcal{Q}}$. In addition, there is only one constraint of the form $s : V$ in $\mathcal{G}_{\mathcal{V}}$.

" $\Rightarrow$ "
Let $\mathcal{I}$ be an $\mathcal{S}$-model of $\mathcal{F}_{\mathcal{Q}}$. Since $\mathcal{F}_{\mathcal{Q}}$ is a complete system, it contains $o : Q$. Hence, $\mathcal{I}$ is an $\mathcal{S}$-model of $o : Q$. Let us consider $\mathcal{I}'$ such that $x'^{\mathcal{I}'} = o^{\mathcal{I}}$ and $\mathcal{I}'$ coincides with $\mathcal{I}$ otherwise. $\mathcal{I}'$ is an $\mathcal{S}$-model of $x' : Q$ and then of $x' : V$. If $\mathcal{I}'$ is an $\mathcal{S}$-model of $x' : V$ then it is also an $\mathcal{S}$-model of $o : V$. As $\mathcal{I}'$ is chosen such that $x'^{\mathcal{I}'} = o^{\mathcal{I}}$ and $\mathcal{I}$ and $\mathcal{I}'$ coincide on all other symbols, we conclude that $\mathcal{I}$ is an $\mathcal{S}$-model of $o : V$.

" $\Leftarrow$ "
Let $\mathcal{I}$ be an $\mathcal{S}$-model of $x' : Q$. As $\mathcal{F}_{\mathcal{Q}}$ is a complete system of $x' : Q$, we can build an $\mathcal{S}$-model $\mathcal{I}'$ for $\mathcal{F}_{\mathcal{Q}}$, from $\mathcal{I}$, with $o^{\mathcal{I}'} = x'^{\mathcal{I}}$ and by modifying the interpretation of the new generated variables and concrete individual names. By hypothesis $\mathcal{I}'$ is also an $\mathcal{S}$-model of $o : V$. We have $V^{\mathcal{I}} = V^{\mathcal{I}'}$ and $o^{\mathcal{I}'} = x'^{\mathcal{I}}$. Hence we can conclude that $\mathcal{I}$ is also an $\mathcal{S}$-model of $x' : V$. ■

Proof (Proposition 6)

We have to verify that $\mathcal{I}_{\mathcal{F}_Q}$ satisfies every axiom in $\mathcal{S}$ and every constraint in $\mathcal{F}_Q$.
First, consider the schema axioms. Suppose that $\mathcal{S}$ contains $A\dot{\preceq}\forall P.A'$. Let $s \in A^{\mathcal{I}_{\mathcal{F}_Q}}$ and $(s,v) \in P^{\mathcal{I}_{\mathcal{F}_Q}}$. Then there is a constraint $v : A'$ in $\mathcal{F}_Q$ since otherwise rule $\mathbf{S_2}$ would be applicable. Thus the axiom is satisfied. We use a similar reasoning for the other forms of axioms.
Next we consider the different constraints in $\mathcal{F}_Q$. By definition of $\mathcal{I}_{\mathcal{F}_Q}$, every constraint $s : A$, $s : \top$, $(s,t) : R, (s,v) : f$ is satisfied. To prove that more complex constraints are satisfied, we proceed by induction. Suppose $\mathcal{F}_Q$ contains $s : P_r(f_1,\ldots,f_n)$. Then because of the rule $\mathbf{D_3}$ it contains as well $s\,f_1\,z_1,\ldots,s\,f_n\,z_n,(z_1,\ldots,z_n) : P_r$ which are satisfied by inductive hypothesis. Hence, $\mathcal{I}_{\mathcal{F}_Q}$ satisfies also $s : P_r(f_1,\ldots,f_n)$.
Suppose $\mathcal{F}_Q$ contains $s : \Theta(C, C', \{\langle f_1, P_{r_1}, g_1\rangle, \ldots, \langle f_m, P_{r_m}, g_m\rangle\})$. Then because of the rule $\mathbf{D_4}$ it contains as well $s : C$, $y : C'$, $s\,f_1\,z_1$, …, $s\,f_m\,z_m$, $y\,g_1\,z_1^{'}$, …, $y\,g_m\,z_m^{'}$, $(z_1, z_1^{'}) : P_{r_1}$, …, $(z_m, z_m^{'}) : P_{r_m}$ which are satisfied by inductive hypothesis. Hence, $\mathcal{I}_{\mathcal{F}_Q}$ satisfies also $s : \Theta(C, C', \{\langle f_1, P_{r_1}, g_1\rangle, \ldots, \langle f_m, P_{r_m}, g_m\rangle\})$

The remaining cases require similar reasoning and are therefore dismissed. ■

Proof (Proposition 7)

The proof can be obtained by induction on the structure of the concept C. Suppose that $\mathcal{F}_Q$ is clash-free and that $\mathcal{I}_{\mathcal{F}_Q}$ satisfies $s : C$.
Suppose $C = A$ (i.e., a concept name), then $s : A \in \mathcal{F}_Q$ by definition of $\mathcal{I}_{\mathcal{F}_Q}$, since $\mathcal{I}_{\mathcal{F}_Q}$ satisfies $s : A$.
If C is of the form $C_1 \sqcap C_2$. Then, $\mathcal{I}_{\mathcal{F}_Q}$ satisfies $s : C_1 \sqcap C_2$ iff $\mathcal{I}_{\mathcal{F}_Q}$ satisfies both $s : C_1$ and $s : C_2$. By inductive hypothesis, this is the case iff $s : C_1 \in \mathcal{F}_Q$ and $s : C_2 \in \mathcal{F}_Q$. Since $s : C_1 \sqcap C_2 \in \mathcal{G}_\mathcal{V}$, we have $s : C_1 \sqcap C_2 \in \mathcal{F}_Q$, since otherwise rule $\mathbf{C_1}$ would be applicable.
If C is of the form $P_r(f_1,\ldots,f_n)$. Then, $\mathcal{I}_{\mathcal{F}_Q}$ satisfies $s : P_r(f_1,\ldots,f_n)$ iff $\mathcal{I}_{\mathcal{F}_Q}$ satisfies $(s,z_1) : f_1$, …, $(s,z_n) : f_n$, $(z_1,\ldots,z_n) : P'_r$ for some $z_1,\ldots,z_n$ where $P'_r(f_1,\ldots,f_n)$ entails $P_r(f_1,\ldots,f_n)$. By inductive hypothesis, this is the case iff $(s,z_1) : f_1 \in \mathcal{F}_Q$, …, $(s,z_n) : f_n \in \mathcal{F}_Q$, $(z_1,\ldots,z_n) : P'_r \in \mathcal{F}_Q$. Since $s : P_r(f_1,\ldots,f_n) \in \mathcal{G}_\mathcal{V}$, we have $s : P_r(f_1,\ldots,f_n) \in \mathcal{F}_Q$ since otherwise rule $\mathbf{C_2}$ would be applicable.
If C is of the form $\{d\}$. Recall that the calculus starts with a pair $\{x' : Q\}.\{x' : V\}$. We make the following remarks: **(1)** By inspecting all the rules of the calculus we see that any individual t occurring in a constraint $t : C'$ in $\mathcal{G}_\mathcal{V}$ occurs also in $\mathcal{F}_Q$. **(2)** By analyzing the rules we see that if a constant (i.e., an abstract individual name) a occurs in $\mathcal{F}_Q$, then $\mathcal{F}_Q$ contains a constraint of the form $t : \{d\}$. Hence, if $\mathcal{I}_{\mathcal{F}_Q}$ satisfies $s : \{d\}$, then by definition $s^\mathcal{I} = d$. The remark (1) leads to the fact that d occurs also in $\mathcal{F}_Q$, and the remark (2) leads to the fact that $\mathcal{F}_Q$ contains $t : \{d\}$. It follows that $t^\mathcal{I} = a$ and then $d : \{d\}$ is in $\mathcal{F}_Q$.
If C is of the form
$\Theta(C, C', \{\langle f_1, P_{r_1}, g_1\rangle, \ldots, \langle f_m, P_{r_m}, g_m\rangle\})$. Then $\mathcal{I}_{\mathcal{F}_Q}$ satisfies $s : \Theta(C, C', \{\langle f_1, P_{r_1}, g_1\rangle, \ldots, \langle f_m, P_{r_m}, g_m\rangle\})$ iff $\mathcal{I}_{\mathcal{F}_Q}$ satisfies $s : C$, $s' : C'$, $s\,f_1\,z_1$, …, $s\,f_m\,z_m$, $s'\,g_1\,z_1^{'}$, $s'\,g_m\,z_m^{'}$, $(z_1, z_1^{'}) : P_{r_1}{}^{'}$, …, $(z_m, z_m^{'}) : P_{r_m}{}^{'}$ for some abstract individual s' and concrete individual names z_1, …, z_m, $z_1^{'}$, …, $z_m^{'}$ and where $P_{r_1}{}^{'}(f_1, g_1)$ entails $P_{r_1}(f_1, g_1)$, …, $P_{r_m}{}^{'}(f_m, g_m)$ entails $P_{r_m}(f_m, g_m)$. By inductive hypothesis this is the case iff $s : C \in \mathcal{F}_Q$, $s' : C' \in \mathcal{F}_Q$, $s\,f_1\,z_1 \in \mathcal{F}_Q$, …, $s\,f_m\,z_m \in \mathcal{F}_Q$, $s'\,g_1\,z_1^{'} \in \mathcal{F}_Q$, $s'\,g_m\,z_m^{'} \in \mathcal{F}_Q$, $(z_1, z_1^{'}) : P_{r_1}{}^{'} \in \mathcal{F}_Q$, …, $(z_m, z_m^{'}) : P_{r_m}{}^{'} \in \mathcal{F}_Q$. Since $s : \Theta(C, C', \{\langle f_1, P_{r_1}, g_1\rangle, \ldots, \langle f_m, P_{r_m}, g_m\rangle\}) \in \mathcal{G}_\mathcal{V}$ we have $s : \Theta(C, C', \{\langle f_1, P_{r_1}, g_1\rangle, \ldots, \langle f_m, P_{r_m}, g_m\rangle\}) \in \mathcal{F}_Q$ since otherwise rule $\mathbf{C_5}$ would be applicable. ■

Proof (Proposition 8)

The Proof follows from the following arguments.

The size of Q is finite. Since $\mathcal{S}$ is acyclic and the size of Q is finite, the number of direct successors of an individual s is finite. When one of the generating rules $\mathbf{D_2}, \mathbf{D_3}, \mathbf{D_4}, \mathbf{D_7}$ is applied to a constraint of the form $s : C$, the number of variables or concrete individual names that are generated is less or equal to the size of C, and if a constraint of the form $y : C'$ is generated then C' is always a strict sub-expression of C. All rules but $\rightarrow_\forall$ ($\mathbf{S_2}$) are not applied twice on the same constraint. The rule $\rightarrow_\forall$ is never applied to an individual s more than the number of direct successors of s.
Consider the rules that alter the goal. As the size of V is finite the number of application of the rule $\mathbf{G_1}$ is finite. As the chain leading from $\{x' : Q\}$ to $\mathcal{F}_Q$ is finite, it follows that $\mathcal{F}_Q$ contains a finite number of constraints, and then the number of application of the rule $\mathbf{G_2}$ is finite. ■

Proof (Theorem 2)

If $\mathcal{F}_Q$ contains a clash, then $\mathcal{F}_Q$ is unsatisfiable. As $x' : Q$ is in $\mathcal{F}_Q$, according to the Proposition 5, $x' : Q$ is unsatisfiable. This means that Q is unsatisfiable and an unsatisfiable concept is subsumed by any concept.

We have seen that $Q \mathrel{\dot{\preceq}_{\mathcal{S}}} V$ iff $x' : Q \models_{\mathcal{S}} x' : V$. (Proposition 4)

" $\Rightarrow$ "
Let $\mathcal{F}_Q$ be a clash-free constraint system and $x' : Q \models_{\mathcal{S}} x' : V$. According to the Corollary 2, we have $\mathcal{F}_Q \models_{\mathcal{S}} o : V$. Let $\mathcal{I}_{\mathcal{F}_Q}$ be the canonical interpretation of $\mathcal{F}_Q$. It follows from the Proposition 6 that $\mathcal{I}_{\mathcal{F}_Q}$ is a model of $\mathcal{F}_Q$. In this case it satisfies $o : V$. We have supposed $o : V \in \mathcal{G}_{\mathcal{V}}$. In this case, according to the Proposition 7 we have $o : V$ in $\mathcal{F}_Q$.

" $\Leftarrow$ "
If $o : V$ is in $\mathcal{F}_Q$, then $\mathcal{F}_Q \models_{\mathcal{S}} o : V$. According to the Corollary 2 we have $x' : Q \models_{\mathcal{S}} x' : V$. It follows from the Proposition 4 that $Q \dot{\preceq}_{\mathcal{S}} V$.

■