A Logic-Based Approach to Multimedia Interpretation

Vom Promotionsausschuss der
Technischen Universität Hamburg-Harburg
zur Erlangung des akademischen Grades
Doktor der Naturwissenschaften (Dr. rer. nat.)
genehmigte Dissertation

von

Atila Kaya

aus Izmir, Türkei

2011

Reviewers:
Prof. Dr. Ralf Möller
Prof. Dr. Bernd Neumann
Prof. Dr. Rolf-Rainer Grigat

Day of the defense:
28.02.2011

Abstract

The availability of metadata about the semantics of information in multimedia documents is crucial for building semantic applications that offer convenient access to relevant information and services. In this work, we present a novel approach for the automatic generation of rich semantic metadata based on surface-level information. For the extraction of the required surface-level information state-of-the-art analysis tools are used. The approach exploits a logic-based formalism as the foundation for knowledge representation and reasoning. To develop a declarative approach, we formalize a multimedia interpretation algorithm that exploits formal inference services offered by a state-of-the-art reasoning engine. Furthermore, we present the semantic interpretation engine, a software system that implements the logic-based multimedia interpretation approach, and test it through experimental studies. We use the results of our tests to evaluate the fitness of our logic-based approach in practice. Finally, we conclude this work by highlighting promising areas for future work.

To my dear parents and wife

Sevgili anneme, babama ve eşime

Acknowledgements

This thesis is the result of five years work in the Institute for Software Systems (STS) research group at the Hamburg University of Technology (TUHH). I am grateful to my advisor Prof. Dr. Ralf Möller for giving me the opportunity to conduct such exciting research and mentoring me. I would also like to thank Prof. Dr. Bernd Neumann and Prof. Dr. Rolf-Rainer Grigat for reviewing this work.

I would like to express my gratitude to all my colleagues at the STS research group: Sofia Espinosa, Sylvia Melzer, Alissa Kaplunova, Tobias Näth, Kamil Sokolski, Maurice Rosenfeld, Oliver Gries, Anahita Nafissi, Dr. Hans-Werner Sehring, Olaf Bauer, Rainer Marrone, Sebastian Wandelt, Volker Menrad and Gustav Munkby. Special thanks go to Dr. Patrick Hupe and Dr. Michael Wessel, who always supported and encouraged me.

I am also indebted to STS staff Hartmut Gau, Ulrike Hantschmann, Thomas Rahmlow, Thomas Sidow for their excellent administrative and technical support.

Finally, I would like to thank my parents Tükez and Dursun, and my wife Justyna for their love, care and continuous support.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation for this Research

The development of methods and technologies to realize convenient access to information is one of the everlasting challenges in computer science. In recent years, with the exponential growth of the number of documents in the World Wide Web as well as in proprietary enterprise and personal repositories, information retrieval has evolved into a major research field in computer science, which directly affects both private and business life.

Nowadays, major web search engines index billions of web pages according to sophisticated algorithms, which exploit mainly textual information from these web pages, and some additional information such as hyperlinks between web pages. In the enterprise scenario, content management systems are often used to support the creation of large amount of documents. In most enterprise-scale information systems at least some means of textual search over document repositories is provided.

Independent of the application scale, all information systems have to analyze and index documents, in order to provide for successful information retrieval. Despite major improvements in the past that empowered the advent of very successful information systems today, two major problems still need to be addressed:

- The majority of current information systems exploit textual information, which is directly identifiable at the surface of a document. Even though this kind of surface-level information can be extracted very successfully in practice, the lack of knowledge about more abstract information prevents not only better retrieval but also the development of more valuable, intelligent services that rely on deep-level semantic information. Nowadays it is widely-accepted that humans prefer to search for documents using abstract instead of surface-level information, in particular in large document repositories.

- Most search functionality provided today relies solely on the analysis of textual data. However, in recent years, not only the amount of documents has changed but also the content of these documents. Facilitated by improvements in hardware and software technology, most documents created today include rich media content such as visual and auditory information. Ignoring information from any other modality than text unnecessarily reduces the performance of information retrieval systems, especially in document repositories with rich multimedia content.

Therefore, it is essential to reveal as much as possible deep-level semantic information about the content of a multimedia document, in addition to considering information in modalities other than text. This will pave the way for the development of semantic applications that support more convenient and more successful information retrieval.

1.2 Research Objectives

The aim of this work is to investigate and develop methods for the automatic generation of rich semantic metadata, which describe the semantics of the information in multimedia documents. By providing a means for the automatic generation of semantic metadata, this work aims to pave the way for the development of semantic applications.

In the past, formal languages have been studied extensively for knowledge representation problems, and they provide appropriate means for representing semantic metadata. Semantic metadata, consisting of surface- and deep-level information, describe a multimedia document, and thus, can be seen as an *interpretation* of the multimedia document. Consequently, the task of computing interpretations of multimedia documents can be called *multimedia interpretation*. A major objective of this work is the development of a declarative, logic-based approach for the multimedia interpretation problem.

Another important objective of this work is to show that the proposed approach can be realized in practice as a software component, which exploits state-of-the-art inference engines and can be integrated with other software systems. We focus on building a stable software system rather than a research prototype. Therefore, the fitness of the software system should be examined by applying it to practical problems and evaluating its performance in terms of runtime, scalability and quality measures.

1.3 Contributions

The major contributions of this thesis are as follows:

- Following the previous line of work on media interpretation, an appropriate knowledge representation and reasoning formalism is identified as the foundation of the multimedia interpretation task. To this end, a logic-based formalism, namely Description Logics (DLs) augmented with rules, is chosen.

- A logic-based multimedia interpretation approach based on formal inference services is developed. The approach exploits ontologies and rules as domain-specific background knowledge, and can be applied to an application domain without the definition of new algorithms, but solely through the formalization of appropriate background knowledge.

- To deal with multimedia documents, which contain information in multiple modalities, a hybrid solution is proposed. Instead of enhancing modality-specific analysis tools with the ability to interpret surface-level information, the hybrid solution integrates existing analysis tools into a coherent framework. In the hybrid solution, analysis tools solely focus on the extraction of surface-level information, and a so-called semantic interpretation engine is responsible for the interpretation of the surface-level information.

- The semantic interpretation engine, an implementation of the proposed multimedia interpretation approach that incorporates state-of-the-art reasoning engines, is presented. The semantic interpretation engine is responsible for the interpretation and fusion tasks. The surface-level information, which serves as input for these tasks, is extracted by state-of-the-art analysis tools.

- The semantic interpretation engine is evaluated in a practical scenario in terms of runtime performance and scalability. Additionally, the quality of semantic metadata generated by the semantic interpretation engine is examined in an experimental study and evaluated in terms of precision and recall.

- From a more general perspective, this work shows that a logic-based approach for multimedia interpretation cannot only be formalized but also realized in practice in form of a logic-based semantic interpretation engine that automates the generation of high-quality semantic metadata about multimedia documents.

1.4 Dissemination Activities

This thesis is the most comprehensive and up-to-date presentation of our work. However, many parts of this work have been published in various conferences and workshops in order to disseminate contributions at different stages. In the following, these dissemination activities are listed in categories:

Book Chapters

- S. Espinosa, A. Kaya, R. Möller. Logical Formalization of Multimedia Interpretation. *In G. Paliouras, C. D. Spyropoulos, G. Tsatsaronis, editors, Knowledge-Driven Multimedia Information Extraction and Ontology Evolution, Springer LNCS Series, To appear in 2011*

Journal Articles

- S. Castano, S. Espinosa, A. Ferrara, V. Karkaletsis, A. Kaya, R. Möller, S. Montanelli, G. Petasis and M. Wessel. Multimedia Interpretation for Dynamic Ontology Evolution. *Journal of Logic and Computation, Oxford University Press, Advance Access published on September 30, 2008. doi:10.1093/logcom/exn049*

Conference Papers

- S. Espinosa, A. Kaya and R. Möller. On Ontology Based Abduction for Text Interpretation. *In Proceedings of 9th International Conference on Intelligent Text*

Processing and Computational Linguistics (CICLing-2008), number 4919 in Lecture Notes in Computer Science, pages 194-2005, Haifa, Israel, February 2008.

- S. Espinosa, A. Kaya, S. Melzer, R. Möller and M. Wessel. Towards a Media Interpretation Framework for the Semantic Web. *In Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI'07)*, number 4919 in Lecture Notes in Computer Science, pages 374-380, Washington, DC, USA, November 2007.

Workshop Papers

- S. Espinosa, A. Kaya and R. Möller. The BOEMIE Semantic Browser: A Semantic Application Exploiting Rich Semantic Metadata. *In Proceedings of the Applications of Semantic Technologies Workshop (AST-2009)*, Lübeck, Germany, October 2009.

- S. Espinosa, A. Kaya, and R. Möller. Formalizing Multimedia Interpretation based on Abduction over Description Logic ABoxes. *In Proceedings of International Workshop on Description Logics (DL2009)*, Oxford, UK, July 2009.

- S. Espinosa, A. Kaya, and R. Möller. Ontology and Rule Design Patterns for Multimedia Interpretation. *In Proceedings of the BOEMIE Workshop*, Koblenz, Germany, December 2008.

- S. Espinosa, A. Kaya, S. Melzer, R. Möller and M. Wessel. Multimedia Interpretation as Abduction. *In Proceedings of the International Workshop on Description Logics DL-2007*, Brixen-Bressanone, Italy, June, 2007.

- S. Castano, S. Espinosa, A. Ferrara, V. Karkaletsis, A. Kaya, S. Melzer, R. Möller, S. Montanelli and G. Petasis. Ontology Dynamics with Multimedia Information: The BOEMIE Evolution Methodology. *In Proceedings of the ESWC International Workshop on Ontology Dynamics (IWOD 07)*, Innsbruck, Austria, June 2007.

- A. Kaplunova, A. Kaya and R. Möller. Experiences with Load Balancing and Caching for Semantic Web Applications. In I. Horrocks and U. Sattler and F.

Wolter, editors, *Proceedings of International Workshop on Description Logics (DL'06)*, The Lake District, UK, May 2006.

- J. Galinski, A. Kaya and R. Möller. Development of a Server to Support the formal Semantic Web Query Language OWL-QL. *In Proceedings of the International Workshop on Description Logics (DL'05)*, Edinburgh, Scotland, July 2005.
- A. Kaya and K. Selzer. Design and Implementation of a Benchmark Testing Infrastructure for the DL System Racer. *In Proceedings of the Workshop on Application of Description Logics (ADL'04)*, Ulm, Germany, September 2004.

Technical Reports

- A. Kaplunova, A. Kaya and R. Möller. First Experiences with Load Balancing and Caching for Semantic Web Applications. Institute for Software Systems (STS), Hamburg University of Technology, Hamburg, Germany, 2006.

Project Deliverables

- S. Perantonis, R. Möller, S. Petridis, N. Tsapatsoulis, D. Kosmopoulos, M. Anthimopoulos, B. Gatos, E. Iosif, G. Petasis, V. Karkaletsis, G. Stoilos, W. Hesseler, K. Biatov, M. Wessel, A. Kaya and K. Sokolski. 2.9 Semantics Extraction from Fused Multimedia Content. The BOEMIE Consortium, BOEMIE Project Deliverable, Version 1.0 Final, March 2009.
- T. Tikwinski, C. Rosche, G. Paliouras, A. Ferrara, A. Kaya and V. Papastathis. 5.4 Specification of the Architecture. The BOEMIE Consortium, BOEMIE Project Deliverable, Version 1.0 Final, April 2007.
- K. Dalakleidi, S. Dasiopoulou, E. Giannakidou, A. Kaya, V. K. Papastathis, G. Petasis and V. Tzouvaras. 3.2 Domain Ontologies - Version 1. The BOEMIE Consortium, BOEMIE Project Deliverable, Version 2.0 Final, February 2007.
- S. Castano, K. Dalakleidi, S. Dasiopoulou, S. Espinosa, A. Ferrara, G. N. Hess, V. Karkaletsis, A. Kaya, S. Melzer, R. Möller, S. Montanelli and G. Petasis. 4.1 Methodology and Architecture for Multimedia Ontology Evolution. The BOEMIE Consortium, BOEMIE Project Deliverable, Version 1.0 Final, December 2006.

- S. Petridis, N. Tsapatsoulis, D. Kosmopoulos, V. Gatos, P. Fragou G. Petasis, V. Karkaletsis, W. Hesseler, K. Baitov, S. Espinosa, S. Melzer, A. Kaya and S. Perantonis. 2.6 Semantics Extraction from Fused Multimedia Content. The BOEMIE Consortium, BOEMIE Project Deliverable, Version 1.0 Final, February 2008.

1.5 Outline of the Dissertation

The primary goal of this thesis is the development of a declarative, logic-based approach to multimedia interpretation aiming at the automatic generation of rich semantic metadata about multimedia documents.

In Chapter 2 we set the context of this work by introducing multimedia interpretation, its applications and related research fields in Section 2.1. We present pioneering work on image interpretation that has a close connection with multimedia interpretation and are built on logical foundations in Section 2.2. In Section 2.3 we analyze the work presented in Section 2.2 to identify remaining key problems to be solved for a logic-based multimedia interpretation approach.

The goal of Chapter 3 is to logically engineer a multimedia interpretation system that is based on formal inference services, and can be implemented as part of a practical application. In Section 3.1 we select an appropriate formalism, DLs augmented with rules, for our logic-based approach and present necessary preliminaries. Having put an appropriate formalism forward, we introduce a multimedia interpretation system, and present the underlying process including analysis, interpretation and fusion steps in Section 3.2. In Section 3.3 we formalize ABox abduction in DLs as a non-standard retrieval inference service. After the presentation of relevant work on abduction, we also present an algorithm for ABox abduction and appropriate criteria for selecting preferred explanations. In Section 3.4 we present an interpretation algorithm that exploits abduction as the key inference service to compute modality-specific interpretations. Most multimedia documents such as web pages contain information in multiple modalities. Therefore modality-specific interpretations have to be fused to obtain interpretations of multimedia documents. An algorithm for the fusion of modality-specific interpretations of web pages is discussed in Section 3.5.

The engineering of a multimedia interpretation system in Chapter 3 is followed by a case study in Chapter 4. In Section 4.1 we briefly introduce a research project, in which the logic-based approach developed in this thesis plays a central role. This clarifies the practical use of our logic-based approach as part of a large application-oriented research project in a real-world context. The semantic interpretation engine, a software system that implements our logic-based approach, is the topic of Section 4.2. In particular, we focus on architecture and implementation of the semantic interpretation engine. In Section 4.3 the stepwise interpretation of a sample web page provides a case study in how interpretations are computed based on analysis results and background knowledge. Additionally, appropriate strategies for the interpretation process are studied in this section on the basis of examples.

A major goal of this thesis is to show that by selecting an appropriate knowledge formalism, a declarative approach to multimedia interpretation can be derived and realized as a software system, which exploits state-of-the-art reasoning engines. Therefore, as an important contribution of this thesis, we evaluate the interpretations that have been computed by the semantic interpretation engine in Chapter 5. In Section 5.1 we analyze the runtime performance and scalability of the semantic interpretation engine through an experimental study. We present another experimental study in Section 5.2, in which the quality of interpretation results are evaluated in terms of the widely-used metrics recall and precision.

In Chapter 6 we conclude this thesis by summarizing the major achievements of the work. Furthermore, in the light of insights gained throughout this thesis, we present promising directions for future work.

Chapter 2

Logical Formalization of Multimedia Interpretation

Nowadays, many documents in private and enterprise repositories as well as on the web are multimedia documents that contain not only textual but also visual and auditory information. Despite this fact, retrieval techniques that rely only on information from textual sources are still widely used due to the success of current software systems, in particular with respect to stability and scalability. However, to further increase the precision and recall of multimedia retrieval, the exploitation of information from all modalities is indispensable in order to derive high-level descriptions of multimedia content. These descriptions, also called deep-level semantic annotations, play a crucial role in facilitating multimedia retrieval. There is a general consensus that manual annotation of multimedia documents is a tedious and expensive task which must be automated in order to obtain annotations for large document repositories. Multimedia interpretation is defined here as the process of producing deep-level semantic annotations based on low-level media analysis processes and domain-specific conceptual data models with formal, logical semantics.

The primary goal of this chapter is to present logical foundations and formalizations of multimedia interpretation. In order to illustrate what purposes the outcome of the interpretation process should fulfill, we start with an introduction to applications of multimedia interpretation in Section 2.1. In Section 2.1, we also present related research fields, and further characterize the input of the interpretation process as a prerequisite. In Section 2.2 we present pioneering work on image interpretation that have a close connection with multimedia interpretation and are built on logical foundations. In Section 2.3 we analyze the formal image interpretation approaches presented in Section 2.2 in order to identify key problems remaining in developing a logic-based multimedia interpretation approach.

2.1 Applications and Related Research Fields

In the last decade, *information retrieval*, a traditional research field in computer science, has become the underlying basis of daily information access. In particular, the broad use of the World Wide Web and web search engines has accelerated this development. In essence, information retrieval is about providing access to information and covers different aspects such as gathering, indexing and searching of documents. Manning et al. [MRS08] provide the following definition:

> Information retrieval is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers).

The term unstructured data refers to any data that is less structured than data used in relational databases. Modern information systems often support information retrieval in the form of *Boolean retrieval* of documents, where documents are indexed by keywords (or terms). The Boolean retrieval model is a model for information retrieval in which queries are Boolean expressions composed of terms and operators such as *and*, *or*, and *not*. In this model, each document is indexed a priori with respect to keywords. In order to derive keywords or other relational descriptions, techniques from information extraction are applied.

Multimedia retrieval, an emerging research field closely related to information retrieval, investigates the retrieval of documents that contain media content represented using multiple modalities such as text, images, video and audio. Nowadays, the majority of commercial information systems such as web search engines and content management systems often rely only on textual information to support multimedia retrieval and ignore information from other modalities. The success of the long research tradition in text analysis and text-based information retrieval as well as the experience in building practical systems are the main reasons for this. For the analysis and retrieval of textual information, efficient and widely-used systems are available, whereas analysis in other modalities is still an open challenge and retrieval of information from multimedia repositories using high-level description of desired content is primarily an academic discipline [JB08].

However, most of the documents in use today, e.g. web pages, are multimedia documents. They usually contain information in textual and visual modalities, and ignoring information from modalities other than text, the quality of search results is not as high as it could be. Thus, multimedia retrieval has to be improved to include information from all modalities. By leveraging multimedia retrieval, existing information systems such as content management systems and information portals can be enabled to support more convenient services for end-users. In addition, advanced multimedia retrieval can provide for intelligent access to information, which is the key requirement for the realization of upcoming semantic information systems such as the Semantic Web.

As indicated above, multimedia retrieval relies on high-level descriptions of multimedia documents. Descriptions usually are provided in the form of annotations, which are attached (or linked) to multimedia documents. The term *annotations*, also called metadata, denotes information about data of any sort in any media. For example, keywords, which are used to support the retrieval of textual documents can be considered as *syntactic annotations* of these documents. Access to data annotated with keywords is implemented using a syntactic match operation. Improved access to multimedia requires *semantic annotations* of multimedia documents. Although there is no common understanding of what semantic annotations about multimedia encompass, they can be considered as formulas of a logical language with formal semantics. For example, semantic annotations of an image might describe the objects observable in the image as well as the relationships between these objects in terms of a high-level vocabulary. Due to the formal semantics, implicit information can be derived by reasoning systems.

Semantic annotations might also involve the description of more abstract information such as, for instance, events, which are not directly observable in a multimedia document. Semantic annotations involving abstract information are called *deep semantic annotations*. Deep semantic annotations can be obtained in a process called multimedia interpretation through the interpretation of directly observable information with respect to some domain-specific background knowledge.

The identification of directly observable information in different modalities, also called *surface-level information*, has been studied in the past for at least two decades. In natural language processing, *information extraction* is one of the major tasks that aims to automatically extract surface-level information, e.g. entities, relations and event names, from a certain domain. Evaluations have shown that state-of-the-art information extraction systems are very powerful language analysis tools that can recognize names and noun groups with an accuracy higher than 90% [CY99]. Different systems exploit various machine-learning techniques such as k-nearest neighbors or Hidden Markov Models for solving real-world problems in certain domains [AHB$^+$93]. However, information extraction is a more restricted problem than general language understanding. In fact, the language analysis employed in these systems provides for simple, reliable language analysis but not full syntactic language analysis. Therefore, when it comes to extracting more abstract information such as events (or aggregate entities) that require a deep understanding of the domain, information extraction systems are reported not to perform well in general [Gri03, pp. 545].

In computer vision, *object recognition* aims to find objects in an image or video sequence. Even though object recognition has been successfully applied in specific domains, e.g., for finding faces in images [VJ01], general object recognition is still an unsolved problem. In most systems, object recognition follows segmentation, where images are partitioned into regions, i.e. sets of pixels. Each of the pixels in a region are similar w.r.t. some feature such as color, intensity or texture [SHB07]. However, when used alone, global features like color histograms or shape analysis are not appropriate for general purpose object recognition in images [JB08]. Therefore, a wide range of local features, such as Harris corners [HS88], Shape Context [BMP02] and Scale Invariant Feature Transform (SIFT) [Low04], have been proposed. Nowadays, local features are successfully used for solving practical problems. For example, SIFT has been applied to the problem of robot localization in unknown environments in robotics [SLL02]. Mikolajczyk and Schmid present a comprehensive evaluation of various local features in [MS05].

Recently, Leibe and Schiele presented an approach that considers object recognition and segmentation as intertwined processes and uses top-down knowledge for guiding the segmentation process [LS03]. The authors reported on experimental results that show the capacity of the approach to categorize and segment diverse categories such as cars and cows. Even though the identification of observable information in image and video sequences in specific domains can be achieved with state-of-the-art computer vision systems, there is a consensus that object and event recognition in the general domain is beyond the capabilities of current technology [KLSG03].

We conclude that in text, image and video modalities surface-level information can be extracted successfully in certain domains, however when it comes to the extraction of surface-level information in general or to the extraction of more abstract information such as events and aggregate entities, the performance of existing systems are not satisfactory.

Semantic annotations of a multimedia document are used for different purposes over the lifetime of the multimedia document and, in most cases all possible usage scenarios cannot be foreseen at the time of producing semantic annotations. Having deep semantic annotations at hand, information systems can exploit annotations flexibly and, thus, support different application scenarios. For example, assume that deep semantic annotations of a news article have been extracted to be used in a news site specialized on athletics news. The semantic annotations are deep in the sense that they involve information such as: a Russian athlete named 'Yelena Isinbayeva' cleared 5.06, the outdoor world record in female pole vaulting, in an athletics event in Zurich on 28th of August 2009. Assume that the official website of the Zurich tourism is allowed to exploit these semantic annotations as well. Using the same annotations but a tourism-specific ontology the Zurich tourism website can present a link to the corresponding news article to users asking for the most important events that happened in Zurich in 2009. We believe that deep semantic annotations are essential to gain the necessary flexibility in the exploitation of multimedia, and to leverage multimedia retrieval.

Information extraction from text and the field of computer vision are related research fields providing the input to the interpretation process. From now on, we assume that the input to a multimedia interpretation process is available in symbolic form, which is computed by the above-mentioned processes (called analysis processes for short).

It is very well possible that media analysis can be influenced by media interpretation. But for the time being we consider analysis and interpretation as sequential steps.

Deep semantic annotations associated with a multimedia document represent an interpretation of the extracted information about the multimedia document and, therefore, the process of producing deep semantic annotations is called *multimedia interpretation*. The multimedia interpretation process produces deep semantic annotations based on directly observable information in multimedia documents through the exploitation of background knowledge.

The main goal of this thesis is to investigate and develop methods for the automation of the multimedia interpretation process to pave the way for the development of intelligent information systems. The efforts to formalize image interpretation are nowadays being revived in the context of Semantic Web and multimedia interpretation. We presuppose that formal representations are required such that reasoning can be employed. Therefore, in the following section, we present important works on image interpretation that are built on formal foundations, before discussing our logic-based multimedia interpretation approach in detail in the next chapter.

2.2 Related Work On Image Interpretation

In this section we present related work on image interpretation. Image interpretation has a close connection with multimedia interpretation. In fact, the multimedia interpretation problem, for which also modalities beyond images are relevant, can be considered as a generalization of the image interpretation problem. Although there exists a substantial number of approaches to image interpretation in the literature, most of them are not built on formal foundations. In this section we focus on approaches that exploit formal, declarative representations for image interpretation and have been implemented as software systems. Our goal is to study formal approaches to image interpretation and their implementations as software systems.

We expect the reader to be familiar with first-order logic. Furthermore, a basic understanding of standard notions from knowledge representation and reasoning such as deduction and *Closed World Assumption* (CWA) are necessary to follow the discussion in the remaining sections of this chapter. We will discuss these notions in more detail in Chapter 3, in which we present the knowledge representation formalism chosen for this work.

2.2.1 Image Interpretation Based on Model Generation

The first formal theory of image interpretation based on logics was introduced by Reiter and Mackworth [RM87]. They propose a so-called *theory of depiction and interpretation* that formalizes image-domain knowledge, scene-domain knowledge and a mapping between the image and scene domains using first-order logic [RM90]. An interpretation of an image is then defined as a logical model of a set of logical formulas. The approach is based on model generation, since its goal is to compute (or generate) all logical models of a given image.

The Mapsee System

We shortly discuss the main ideas of this approach and recapitulate the system *Mapsee*, which has been implemented for the interpretation of hand-drawn sketch maps of geographical regions [MMH87]. Given a sketch map consisting of chains[1], regions and various relations between them, the goal of the system is to compute an interpretation in terms of roads, rivers, shores, areas of land, areas of water etc.

The image-domain knowledge includes general knowledge about maps such as the taxonomy of image-domain objects, which are specified through first-order logic axioms:

$$\begin{aligned}&\forall x: \ \textit{image-object}(x) \ \leftrightarrow \ \textit{chain}(x) \vee \textit{region}(x)\\ &\forall x: \ \neg(\textit{chain}(x) \wedge \textit{region}(x))\end{aligned}$$

The first axiom states that chains and regions, so-called image primitives, are the only objects that can exist in a map, whereas the latter axiom states that an object cannot be both chain and region at the same time (disjointness of image primitives). Relations between image-domain objects are also part of the image-domain knowledge and are specified using axioms such as $tee(c, c')$ and $bound(c, r)$. For example, the axiom $tee(c, c')$ means that chain c meets chain c$'$ at a T-junction, and $bound(c, r)$ means that chain c encloses region r.

The approach assumes a map description to consist of finitely many chains and regions together with finitely many relations between the chains and regions. Therefore, the system makes the *domain closure assumption* by postulating that all image-domain

[1]Chain is the term used in the original paper for polylines.

objects are completely known. To this end, closure axioms of the following form are used:

$\forall x: \ \mathit{chain}(x) \ \leftrightarrow \ x \ = \ i_1 \vee \cdots \vee x \ = i_m$
$\forall x: \ \mathit{region}(x) \ \leftrightarrow \ x \ = \ i'_1 \vee \cdots \ \vee x \ = i'_n$
$\forall x, y: \ \mathit{tee}(x, y) \ \leftrightarrow \ (x \ = \ i_1 \wedge y \ = \ i'_1) \vee \cdots \vee \ (x \ = \ i_k \wedge y \ = \ i'_k)$
$\ldots$

where i and i' are constants representing image-domain objects.

Furthermore, the system makes the *Unique Name Assumption* (UNA) by assuming that all constants (e.g., image primitives such as chains and regions) denote different objects. Both assumptions, the domain closure assumption and the UNA, play an important role in the logical framework, as we will see later.

Scene-domain knowledge is represented by axioms for objects such as roads, rivers, shores, land and water areas. For instance, the following subsumption, coverage and disjointness axioms are used:

$\forall x: \ \mathit{scene\text{-}object}(x) \ \leftrightarrow \ \mathit{linear\text{-}scene\text{-}object}(x) \vee \mathit{area}(x)$
$\forall x: \ \mathit{linear\text{-}scene\text{-}object}(x) \ \leftrightarrow \ \mathit{road}(x) \vee \mathit{river}(x) \vee \mathit{shore}(x)$
$\forall x: \ \neg(\mathit{road}(x) \wedge \mathit{river}(x))$
$\forall x: \ \neg(\mathit{linear\text{-}scene\text{-}object}(x) \wedge \mathit{area}(x))$
$\ldots$

In addition, the scene-domain knowledge contains also specific restrictions such as, for instance, rivers do not cross each other:

$\forall x, y: \ \mathit{river}(x) \wedge \mathit{river}(y) \rightarrow \ \neg \ \mathit{cross}(x, y)$

Also, axioms that restrict the domain and range to scene objects only are used:

$\forall x, y: \ \mathit{cross}(x, y) \rightarrow \ \mathit{scene\text{-}object}(x) \ \wedge \ \mathit{scene\text{-}object}(y)$

Besides the specification of image- and scene-domain knowledge, also relations between the image- and scene-domain objects are specified. The mappings are represented by the binary predicate $\Delta(i, s)$ meaning that image object i depicts scene object s. The depiction relation only holds between image and scene objects:

$\forall i, s : \ \Delta(i, s) \rightarrow \textit{image-object}(i) \wedge \textit{scene-object}(s)$

For specifying image-scene-domain mappings, closure and disjointness axioms are provided:

$\forall x : \ \textit{image-object}(x) \vee \textit{scene-object}(x)$
$\forall x : \ \neg(\textit{image-object}(x) \wedge \textit{scene-object}(x))$

Furthermore, it is assumed that every image object i depicts a unique scene object, which is denoted by $\sigma(i)$:

$\forall i : \ \textit{image-object}(i) \rightarrow \textit{scene-object}(\sigma(i)) \wedge \Delta(i, \sigma(i)) \wedge [\forall s : \Delta(i, s) \rightarrow s = \sigma(i)]$

and every scene object is depicted by a unique image object:

$\forall s : \ \textit{scene-object}(s) \rightarrow (\exists^1_i : \ \textit{image-object}(i) \wedge \Delta(i, s))$

The notation $\exists^1_x : \alpha(x)$ means that there exists exactly one x for which $\alpha(x)$ holds. Finally, mappings between the image- and scene-objects:

$\forall i, s : \ \Delta(i, s) \wedge \textit{region}(i) \rightarrow \textit{area}(s)$
$\forall i, s : \ \Delta(i, s) \wedge \textit{chain}(i) \rightarrow \textit{linear-scene-object}(s)$

and mappings between relations of the image and scene domains are specified:

$\forall i_1, i_2, s_1, s_2 : \ \Delta(i_1, s_1) \wedge \Delta(i_2, s_2) \rightarrow \textit{tee}(i_1, i_2) \leftrightarrow \textit{joins}(s_1, s_2)$
$\forall i_1, i_2, s_1, s_2 : \ \Delta(i_1, s_1) \wedge \Delta(i_2, s_2) \rightarrow \textit{chi}(i_1, i_2) \leftrightarrow \textit{cross}(s_1, s_2)$
$\ldots$

The above-mentioned axioms state that *tee*[1] relations in the image depict *joins* relations in the scene and vice versa, whereas *chi*[2] relations in the image depict *cross* relations in the scene.

Given the specification of all relevant image-domain axioms, scene-domain axioms and mapping axioms, Reiter and Mackworth define an *interpretation* of an image as a logical model of the set of axioms and the set of facts describing a particular image.

[1]Shorthand for T-junction.
[2]Shorthand for X-junction.

The main problem here is that, in principle, a set of first-order formulas may have infinitely many models and therefore the computation of all models may become impossible. Even worse, it is undecidable in general whether a set of first-order formulas has a model at all. However, Reiter and Mackworth show that as a consequence of the assumptions made in their logical framework, it is possible to enumerate all models. In fact, under the additional CWA, extensions of all predicates can be defined, and therefore quantified formulas can be replaced with quantifier-free formulas. Consequently, first-order formulas can be reduced to propositional formulas, for which the computation of all models is possible [GN87]. Reiter and Mackworth formulate the problem of determining all models of the resulting propositional formulas as a *constraint satisfaction problem* (CSP). Although, in general, CSPs of this kind are NP-hard, and thus computationally intractable, several efficient approximation algorithms exist, which have also been used in the Mapsee system [MMH87].

2.2.2 Image Interpretation Based on Abduction

Inspired by the work of Reiter and Mackworth, Matsuyama and Hwang address the image interpretation problem but follow a different approach. According to Matsuyama and Hwang, the goal of image interpretation is to provide for explanations of the observations of an image, through the exploitation of axiomatized general knowledge about the world and the generation of a set of logical hypothesis. To this end, the authors follow the *hypothetical reasoning* approach of Poole et al. [PGA87, Poo89].

Hypothetical reasoning is the form of reasoning that enables to reason from observations to explanations through the generation of hypothesis (also known as explanations). This form of reasoning has initially been introduced by Peirce under the name *abduction* in the late 19th century [Pei78]. Abduction is often defined as a reasoning process from evidence to explanation, which is a type of reasoning required in several situations where the available information is incomplete [Ali06]. Abduction has been widely used to formalize explanation-based reasoning and plays an important role in intelligent problem solving tasks such as medical diagnosis [PGA87] and plan recognition [CG91].

The SIGMA System

In [MH90], Matsuyama and Hwang present a vision system called SIGMA, which has been developed for the interpretation of aerial images. In the SIGMA system, abduction-based image interpretation has been implemented as a recursive process, where expectations are explained through hypotheses.

Matsuyama and Hwang use aerial images of suburban areas that typically show houses and roads. First-order logic axioms are used to represent general knowledge about the application domain. For example, the fact that every house is related to exactly one street is represented as follows (for the sake of the example the relation is called *rel*):

$$\forall x : house(x) \rightarrow (\exists y : \ road(y) \wedge rel(x,y) \wedge \forall z : (road(z) \wedge rel(x,z)) \rightarrow z = y)$$

which is transformed into *clausal normal form*:

$$\neg house(x) \vee road(f(x))$$
$$\neg house(x) \vee rel(x, f(x))$$
$$\neg house(x) \vee \neg road(z) \vee \neg rel(x,z) \vee z = f(x)$$

where the existential quantification is replaced with a so-called *Skolem function*. The Skolem function replaces the existentially quantified variable by creating a new constant.

As an example, assume an aerial image depicting a house. The house is represented by the constant h_1. Given the above-mentioned axioms representing the general knowledge about the domain and information about the existence of a house in the scene, namely $house(h_1)$, the following information is entailed:

$$road(f(h_1))$$
$$rel(h_1, f(h_1))$$
$$\neg road(z) \vee \neg rel(h_1, z) \vee z = f(h_1)$$

Here, the new constant $f(h_1)$, denoted by using the Skolem function *f*, is called an *expected object*, in this example a road, and has to be identified in the image.

In contrast to Reiter and Mackworth, Matsuyama and Hwang do not assume the availability of an a priori image segmentation, and do not make the domain closure assumption and the UNA for the image domain. Constant symbols representing image-domain objects are not available in the beginning, but have to be created through an expectation-driven segmentation approach, which is part of the interpretation process. Consequently, also constant symbols representing scene objects are not available in the beginning of the interpretation process and have to be computed through hypotheses.

In the SIGMA system, objects in the scene (e.g. houses, roads) are associated with features in the image (e.g. points, lines, regions). Different classes of scene objects and spatial relations are defined through necessary conditions in terms of the image domain:

$$\forall x : road(x) \rightarrow greater(width(x), 5) \wedge less(width(x), 100) \wedge ribbon(shape(x))$$
$$\forall x, y : rel(x, y) \rightarrow parallel(axis(x), axis(y)) \wedge distance(center(x), center(y), 50)$$

Object attributes such as *width*, *shape*, *axis* or *center* are modeled through functions, predicates regarding spatial attributes such as *greater*, *less*, *ribbon*, *parallel* or *distance* are modeled as constraints. These axioms define the conditions that must hold for the objects of the scene-domain.

Assume that our sample image depicts, besides the house h_1, also a road represented by the constant r_1. After adding a new axiom to represent this information, namely $road(r_1)$, the following information is entailed:

$$\neg rel(h_1, r_1) \vee r_1 = f(h_1)$$

Notice that for spatial relations of the scene-domain such as *rel* only necessary conditions are defined but not sufficient ones. Therefore it cannot be proved logically, whether $rel(h_1, r_1)$ holds or not. To solve this problem, a special equality predicate is used in SIGMA, which reflects two important assumptions about the equality of scene objects: i) Two scene objects are considered to be identical, if they are of the same type, e.g. road, and have the same shape and position, i.e. occupy the same space. ii) If an existing scene object fulfills all conditions that an expected object has to fulfill, both objects are considered to be identical.

In our example, if r_1 fulfills all conditions that have to be fulfilled by the expected object $f(h_1)$ then as a result of the equality assumption, the *hypothesis* $r_1 = f(h_1)$ is generated, and $rel(h_1, r_1)$ is derived. In case no suitable scene object that is identical to the expected object $f(h_1)$ exists, the conditions of the expected object $f(h_1)$ are used for an expectation-driven image analysis process to identify an object in the image. In case an object is identified, a new constant symbol is introduced into the image domain, e.g. r_2, and the hypothesis $road(r_2)$ is created. Afterwards, the hypothesis $r_2 = f(h_1)$ is generated and $rel(h_1, r_2)$ is derived.

In order to guarantee termination, expected objects are not allowed to trigger the derivation of new expected objects, e.g. $g(f(r_1))$. In other words, expectations are not used to derive further expectations. Expectation generation is done solely through the exploitation of constant symbols, which can only be introduced by an expectation-driven image analysis process. After the generation of constant symbols the domain closure assumption is applied. Therefore, the set of first-order logic axioms can be transformed to a set of propositional logic axioms.

As mentioned above, in their work Matsuyama and Hwang follow the *hypothetical reasoning* approach of Poole et al. [PGA87, Poo89] where the task is to compute a set of logical hypotheses such that following conditions are fulfilled:

i) $\{axioms\} \cup \{logical\ hypothesis\} \models \{observations\}$
ii) SAT($\{axioms\} \cup \{logical\ hypothesis\}$)

Logical hypotheses are either *classification hypotheses*, ground instances of unary predicates with constant symbols, e.g. $road(r_2)$, or *equality hypotheses*, equality relations between constant symbols and ground instances of Skolem functions such as $r_2 = f(h_1)$. As discussed earlier, ground instances of predicates can be derived through the exploitation of logical hypotheses and general knowledge. For example, in the previous example $rel(h_1, r_2)$ has been derived, which represents a spatial relation in the scene. Matsuyama and Hwang call this derivation process the construction of a scene description, which can later be mapped to the image-domain to explain the observations of the image.

The second condition on interpretations requires the union of the sets of axioms and logical hypothesis to be *consistent* in order to provide a valid interpretation of an image. However, in general, the problem of checking whether a set of first-order logic formulas is consistent is undecidable. To provide for a pragmatic solution, Matsuyama and Hwang presume that the set of axioms are consistent by definition and define an application-specific consistency check for overlapping objects. Using special-purpose procedures, each set of overlapping objects is checked programmatically to find out whether the objects have incompatible attributes and, thus, cannot be identical. If this is the case for one of the sets of overlapping objects, then the set of axioms is inconsistent and consequently the interpretation is not valid.

According to Matsuyama and Hwang, during the interpretation process, inconsistencies can only arise if negations can be derived as well. Therefore, the set of axioms has to be extended with additional rules to state the disjointness of different classes of objects. For example:

$\forall x : \neg(road(x) \wedge house(x))$

In his doctoral thesis [Sch98, pp. 30], Schröder points out a second important source of inconsistencies in Matsuyama and Hwang's approach, which is not addressed by the work of Matsuyama and Hwang. He shows that in cases where an expected object cannot be identified in an image, the absence of the expected object has to be stated formally through the negation of its existence as follows:

$\neg\exists x : \textit{image-object}(x) \;\wedge\; \textit{image-position}(x, \ldots) \;\wedge\; \ldots$
$\neg\exists x : \textit{scene-object}(x) \;\wedge\; \textit{scene-position}(x, \ldots) \;\wedge\; \ldots$

Schröder proposes the definition of an application-specific consistency check that can detect an inconsistency in case axioms contradict about the existence of an object in a scene. For example, the above-mentioned axioms about the absence of an expected object together with an axiom about the existence of an expected object should raise an inconsistency. In addition, Schröder argues that the existence of scene objects that are not visible in an image always have to be negated formally with the help of axioms in order to guarantee the termination of the interpretation process. Otherwise, objects which are not visible in the image could be hypothesized and, in turn, this might lead to the hypothesis of an infinite number of objects in the worst case.

The hypothesis generation process in SIGMA computes so-called *interpretation networks*, which are networks consisting of mutually related object instances. Multiple interpretation networks can possibly be constructed for an image. In an interpretation network, multiple objects instances may be located in the same place in the scene. Such instances are called *conflicting instances*, and a so-called *in-conflict-with* relation is established between them. It should be noted that the SIGMA system applies no heuristics to select among the possible sets of networks but delivers the first computed set of networks as result.

2.2.3 Image Interpretation Based on Deduction

Other relevant work on image interpretation built on formal foundations is due to Russ et al. who follow a deduction-based approach to image interpretation. In [RMS97], Russ et al. present the VEIL system (Vision Environment Integrating Loom) that aims to improve computer vision programs by applying formal knowledge representation and deductive reasoning services provided by the Loom system [RMS97]. To this end the authors propose a layered architecture integrating vision processing, knowledge representation and reasoning. In this architecture a computer vision program operates at the pixel level using specialized data structures to deal with low-level processing, whereas the knowledge representation and reasoning system Loom uses symbolic structures to represent and reason higher-level knowledge.

The VEIL System

One of the major goals of VEIL is to enable the construction of explicit declarative vision models. This is achieved by exploiting the knowledge representation and reasoning facilities provided by the Loom system [MB87, Bri93]. The Loom system provides support for an expressive knowledge representation language in the KL-ONE family and reasoning tasks. It supports not only deductive reasoning but provides also facilities to apply production rules. The declarative specification of knowledge offers various benefits: i) It is easier to maintain than a procedurally specified program. ii) It enables the application of automatic validation and verification techniques. iii) Data is represented in a high-level specification instead of application-specific data structures, and thus can easily be shared or reused by other applications.

Similar to the Mapsee and SIGMA systems, also in the VEIL system, domain knowledge is represented in two different models. The *site model* is a geometric model of concrete image objects such as *runways*, *markings*, *buildings* and *vehicles*. The so-called *domain model* contains not only concrete objects such as *roads*, *buildings* and *vehicles* but also abstract aggregate objects such as *convoys* (groups of vehicles) and events such as *field training exercises*.

In the VEIL project, deductive reasoning is employed to classify an instance as belonging to a concept. For example, assume that a group of pixels in an image is identified as a *vehicle* instance v_1 and added to the knowledge base. Further analysis of the same group of pixels might unveil that v_1 has tracks. After the addition of this information into the knowledge base, Loom classifies v_1 as a *tracked-vehicle* instance, where the concept tracked-vehicle is defined as a subconcept of the concept vehicle. This is possible, because the concept tracked-vehicle is defined with necessary and sufficient conditions, which are all fulfilled by v_1.

Concrete objects in the domain model are linked to geometric objects in the site model. Operations on geographic information, e.g. the determination of the geographic location of an object, require spatial reasoning. To this end several functions such as *location*, *is-near* and *area* have been implemented at the site model level. These site model level functions are linked to domain level relations. For example, the domain level relation *area* is linked to the corresponding site level function. Loom allows a domain level relation to be defined as a procedural function.

The Loom system supports querying about objects and relationships in images. Besides a set of predefined queries, users can create new queries and assign names to that queries. Later, these names can be used in subsequent queries. This allows the user to dynamically extend the vocabulary defined in the domain model.

In [RPM+98], the practical application of Loom in two main areas of the VEIL project has been reported. First, the application of Loom in developing and extending an existing computer vision system for airport runway detection is presented. Second, the application of Loom for the integration of higher-level knowledge and the detection of events are discussed with examples. In the following we present these two application scenarios briefly.

The first application scenario is the detection and analysis of aerial photographs of airports. Airports are modeled as collections of runways, which are long thin ribbons with markings (smaller ribbons) in certain locations. Aerial images are analyzed by the computer vision system through standard analysis techniques such as the Canny edge detector [Can86] to produce hypotheses. A sequence of filtering and grouping operations are then applied to reduce the number of hypotheses. In the next step, hypotheses are verified using the site model of the application scenario, which is defined using Loom. For example, the site model describes markings in terms of their sizes, relative positions and positions on the runway. The domain knowledge represented using Loom is used to constrain the set of possible hypotheses. For example, descriptions of the size and location of markings are used to rule out some hypotheses generated by the computer vision system. To this end, the deductive reasoning service of Loom is used to determine the most-specific concepts that an object is an instance of.

The second application scenario is the detection of event sequences that span multiple images. The goal of this scenario is to process a sequence of images and detect events such as field training exercises. Forty images of a hypothetical armored brigade garrison and exercise area that share a common site model have been used in the experiments reported in [RPM+98].

In the VEIL context, an event is a sequence of scenes that satisfy certain criteria. A scene is represented as a set of object descriptions (called a world), which can be associated with a timestamp. Some of the criteria such as the temporal order apply across different scenes, whereas other criteria apply only within a single scene. In the event detection scenario, several objects such as vehicles and their locations are identified by a human expert. The human expert also corrects initial groupings of building and identifies groups of vehicles. Furthermore, the human expert adds corresponding objects to the site model. Once all relevant information is added to the site model, Loom's query answering service is used to automatically determine sequences of images that satisfy all conditions of an event definition.

Let us consider an example: A field training exercise is a sequence of scenes showing an armored unit in a garrison, then moving in convoy, then deployed in a training area and finally in a convoy again. In order to extract the scenes that meet the criteria of a field training exercise event, the following query is used:

```
(retrieve (?Y ?S1 ?S2 ?S3 ?S4)
      (and (within-world ?S1 (in-garrison ?Y))
           (within-world ?S2 (convoy ?Y))
           (within-world ?S3 (deployed-unit ?Y))
           (within-world ?S4 (convoy ?Y))
           (before+ ?S1 ?S2) (before+ ?S2 ?S3) (before+ ?S3 ?S4)))
```

Query terms, e.g. *in-garrison* and *deployed-unit*, are defined in the domain model. The result of the query is a set of tuples. Each tuple is a field training exercise event since it satisfies all conditions defined in the query. Each detected event is displayed as a collection of images in the graphical user interface of the VEIL system.

Ontology-based Interpretation of Road Networks

Recently, Hummel [Hum09] presented another deduction-based approach, in which *Description Logics* (DLs), the successor of KL-ONE, is used as a formal language for representing knowledge and reasoning about it. DLs are a family of knowledge representation formalisms, that according to Baader et al. [BCM$^+$03, pp. 47], represent the knowledge of an application domain by first defining the relevant concepts of the domain (in the so-called *TBox*), and then using these concepts to specify objects and individuals occurring in the domain (in the so-called *ABox*).[1]

The system developed in this work integrates a state-of-the-art computer vision system and a so-called *DL-reasoner* for deductive reasoning tasks. More precisely, in [Hum09], Hummel uses the expressive description logic $\mathcal{SHIQ}$, which is supported by the state-of-the-art DL-reasoner RacerPro [HM01]. The overall goal of the system is to facilitate an autonomous driverless car through the interpretation of road intersections. To this end, the system is provided as input with sensor data from a camera and a global positioning system (GPS) mounted on a vehicle, as well as with data from a digital map. For each road intersection the system is then requested to answer questions such as 'Which driving directions are allowed on each lane?', 'Which of the map's lanes is equivalent to the vehicle's ego lane?' (i.e., on which lane is the vehicle), etc. Answering such questions requires reasoning since regulations of roads and intersections as well as partial and non-complementary information from various sensors about the current situation of the car have to be considered together.

In her work, Hummel investigates appropriate ways for representing relevant scene information in DLs. For typical classes of scene information she proposes generic DL representations, which she refers to as *design patterns*. In particular, she presents design patterns for representing sensor data and qualitative scene geometry models in DLs. In the context of road intersection interpretation, different sensor setups are investigated as well. If a still image from a single sensor is interpreted, the UNA should be imposed such that two individuals in the ABox are always interpreted (in the sense of first-order logic) as different objects. However if data is acquired by multiple, non-complementary

[1]Syntax and semantics of Description Logics will be introduced in Section 3.1.1.

sensors, objects are detected multiple times, and hence the UNA must not hold. For the multiple sensor setup, Hummel requires the UNA to hold within data acquired by a single sensor only, which she calls the *local UNA*. She reports the local UNA to have been implemented as a procedural extension that enhances a knowledge base through the application of rules in a forward-chaining way.[1]

Furthermore, Hummel investigates image interpretation tasks with respect to their solvability through standard deductive DL inference services. These tasks are i) Object detection, i.e., the discovery of new scene objects ii) Object classification, i.e., the assignment of labels to a detected object iii) Link prediction, i.e., predicting the existence and types of relationships between objects iv) Data association, i.e., the identification of a set of measurements as referring to the same object. She shows that in order solve the object classification task with standard DL inference services, the maximum possible number of individuals in a scene have to be added a priori to the ABox, which describes the scene. A corresponding design pattern has been proposed in [Hum09]. In fact, if this design pattern is applied, the task of object detection can be reduced to the task of object classification, which can be solved using the so-called *ABox realization* DL inference service. In a nutshell, *ABox realization* is a deductive DL inference service that computes for all individuals in an ABox $\mathcal{A}$ their most-specific concept names w.r.t. a TBox $\mathcal{T}$.[2]

In contrast to object detection and object classification, the task of link prediction cannot be solved in $\mathcal{SHIQ}$, since $\mathcal{SHIQ}$ does not allow for role constructors and hence is not expressive enough. It is also shown that the data association task can be solved using the so-called *unification* DL inference service. In a nutshell, *unification reasoning* is a deductive DL inference service that checks for semantic equality between two individuals. Obviously, unification reasoning requires the UNA to be abandoned.

Hummel also presents the so-called road network ontology (RONNY), a $\mathcal{SHIQ}$ DL TBox in which the qualitative geometry and building regulations of roads and intersections are specified. Finally, she describes a case study where the logic-enhanced system solves interpretation tasks using RONNY and sensor data from a stereo vision sensor, a global positioning system, and a digital map. The performance of the system in solving object detection, object classification and data association tasks has been evaluated on

[1]Rule formalisms will be introduced in Section 3.1.2.

[2]Inference services in DLs will be formally introduced in Section 3.1.1.

a sample set of 23 diverse and complex intersections from urban and non-urban roads in Germany. In [Hum09], the system build through the integration of a DL-reasoner and a computer vision system is reported to significantly improve recognition rates of the computer vision system.

2.3 Discussion

After the presentation of three logic-based approaches to image interpretation (model generation, abduction and deduction), and their implementations as software systems, in this section, we discuss the commonalities and differences of these approaches. Based on the insights gained, we present the key idea in developing a logic-based interpretation approach, which will later be used to logically engineer a multimedia interpretation system.

In their work, Reiter and Mackworth presented a model generation-based approach to image interpretation in which first-order logic is used as the knowledge representation formalism. They provided the first formal definition of the image interpretation problem in a logical framework, which is an important contribution of their work. In this approach an interpretation of an image is defined as a logical model of a set of first-order formulas, and the goal is to compute all logical models of a given image.

The main problem of this approach is that, in general, a set of first-order formulas may have infinitely many models and hence the computation of all models may become impossible. In this approach, the domain closure assumption, the UNA and the CWA are made to overcome this problem. These assumptions can be made for the interpretation of hand-drawn sketch maps, however they are too strict for the interpretation of images and multimedia documents, since the combination of these assumptions prevents the dynamic creation of new objects during the interpretation process. Consequently, these assumptions make it impossible to create aggregates representing more abstract scene objects, which are essential for the generation of deep-level annotations.

Like Reiter and Mackworth also Matsuyama and Hwang use first-order logic to discuss an abduction-based approach to image interpretation. An important contribution of Matsuyama and Hwang's work is the insight that, in general, concrete observations about an image cannot logically follow solely from the background knowledge, which contains axioms representing general knowledge about the domain, but only if appropriate explanations are hypothesized and added to the background knowledge. The authors consider the computation of explanations as an abductive reasoning task.

Compared to Reiter and Mackworth's work, Matsuyama and Hwang address a more challenging problem because the images to be interpreted are not hand-drawn sketch maps but aerial photographs where factors such as illumination hinder object detection. Matsuyama and Hwang do not make the domain closure assumption and the UNA in the image domain. Therefore they do not assume the availability of image-domain objects at the beginning of the interpretation process, but require them to be created through a so-called expectation-driven segmentation approach during the interpretation process. Scene-domain objects are also not available in the beginning of the interpretation process and have to be computed through hypotheses.

Different from Reiter and Mackworth's model generation-based approach, Matsuyama and Hwang's abduction-based approach has been implemented in a vision system. To give a logical formulation of image interpretation based on abduction, Matsuyama and Hwang use first-order logic as the knowledge representation and reasoning formalism, however, in the implementation of the approach *frames* and *production rules* are used instead.

A deduction-based approach has been proposed and implemented by Russ et al. in which a KL-ONE like formalism (the Loom knowledge representation language) is used as the formalism for the interpretation of aerial photographs of airports. An important contribution of this approach is the proposal of a layered architecture that integrates vision processing, knowledge representation and reasoning. In this architecture, a computer vision system operates at the pixel level, whereas the Loom system is used to represent and reason about higher-level knowledge.

In the VEIL project, the deduction-based approach has been studied in two practical application scenarios. In the airport runway detection scenario, hypotheses are generated by the computer vision system, and then verified using the deductive reasoning service of Loom. Deductive reasoning in Loom is, in general, incomplete, which is a shortcoming of this approach. In the event detection scenario, Loom's query answering service is used to detect events. This is possible only after objects are identified and added by an human expert, which is another shortcoming of this approach.

Another deduction-based approach has recently been presented by Hummel. In this approach the expressive description logic $\mathcal{SHIQ}$ is used as the knowledge representation formalism, in which deductive reasoning is complete and highly-optimized reasoners exist, which is an important advantage of Hummel's approach. Like Russ et al. also Hummel integrates a computer vision system and a reasoner. Through a case study she shows that most image interpretation tasks can be solved using deductive DL inference services. However, Hummel also observes that the image interpretation problem cannot be solved through deduction only, and uses production rules to enhance the knowledge base before exploiting deductive reasoning to solve image interpretation tasks.

To summarize, all approaches are built on formal foundations, and most of them share the view that image interpretation cannot be solved through deductive reasoning only. The approaches differ in the way they compute an image interpretation: While some reduce first-order formulas to propositional formulas and consider the computation of an interpretation as a constraint satisfaction problem, others build a procedural framework or procedurally extend a deductive reasoner.

In this work, we favor an abduction-based multimedia interpretation approach, where abductive reasoning is used to compute explanations for the observations, i.e. analysis results of a multimedia document, in order to generate interpretations of the multimedia document, i.e. deep semantic annotations. For the implementation of abductive reasoning we prefer the enhancement of a state-of-the-art DL reasoner. Since our goal is the derivation of a declarative approach, we omit to enhance the DL reasoner through procedural extensions but exploit rules instead. I.e., for abductive reasoning we propose the exploitation of rules to define the space of abducibles. As we will discuss in detail in the next chapter, abduction in DLs can be considered as an extended query answering service. In a nutshell, the use of rules for the definition of the space of abducibles enables the computation of hypothesis in a more goal directed way, since only the abducibles required to answer a certain query have to be hypothesized. In addition to rules, we propose an additional mechanism for the computation of *preferred* explanations w.r.t. the application context.

In the next chapter we logically engineer a stable software system for multimedia interpretation. To this end we first formalize abductive reasoning towards the computation of preferred explanations, and then develop a multimedia interpretation approach that exploits abductive reasoning as the key inference service.

Chapter 3

Logical Engineering of a Multimedia Interpretation System

The main goal of this chapter is to logically engineer a multimedia interpretation system for the computation of deep semantic annotations of multimedia documents without human intervention. In developing a multimedia interpretation engine, we cannot design our system in a completely top-down fashion and implement it from scratch. In fact, we follow a bottom-up design to engineer a multimedia interpretation system through the integration of existing software systems such as computer vision systems and reasoners. The functionality of existing software systems is predefined and not always modifiable. As a result, the design of a multimedia interpretation system is also driven by the functionality of existing software systems that underly it.

For multimedia interpretation, i.e. for the generation of deep semantic annotations, we propose a *hybrid approach*. The approach is hybrid in the sense that it combines surface-level information extraction and multimedia interpretation. The general structure of this hybrid approach is sketched in Figure 3.1. First, surface-level information is identified in different modalities using existing state-of-the-art information extraction and object recognition techniques. Second, a multimedia interpretation process operates on the identified surface-level information, and at the same time exploits background knowledge in order to produce deep semantic annotations. The hybrid approach pursued in this work has several merits:

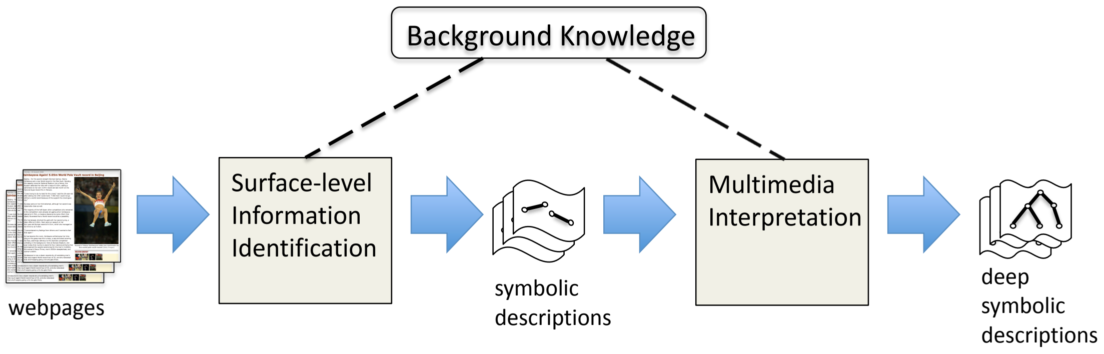

Background Knowledge
Surface-level Information Identification
symbolic descriptions
Multimedia Interpretation
deep symbolic descriptions
webpages

Figure 3.1: The hybrid approach for obtaining deep semantic annotations

- State-of-the-art natural language processing systems and computer vision systems cannot only be exploited for information extraction and object recognition as standalone software components but they can also be integrated into a coherent framework.

- The multimedia interpretation process is independent of the systems used for the identification of surface-level information. As a result, the techniques do not suppose any constraints on the multimedia interpretation process. This enables a higher degree of flexibility in choosing and developing appropriate techniques for the multimedia interpretation process.

- The background knowledge serves as a common vocabulary of all semantic annotations in a domain and can be exploited by the systems that identify surface-level information as well as the multimedia interpretation process. This empowers knowledge engineers to model the background knowledge for all modalities.

- The approach considers multimedia interpretation not merely as a data-driven, that is, bottom-up process but also as a process that incorporates top-down knowledge. In fact, the background knowledge represents top-down knowledge and is actively involved in the multimedia interpretation process, besides the surface-level information. For example, the background knowledge is used to discard some of the interpretations of a multimedia document, which are not consistent with respect to the background knowledge, or are not preferred.

- Typically, multimedia documents contain information from multiple modalities. The hybrid approach can exploit synergies that arise from fusing information from different modalities. For example, most web pages contain textual and graphical information. Detecting a person's identity in an image is a challenging task for object recognition, whereas the recognition of person names in free text is a standard task for information extraction. By appropriate fusion of the information from different modalities, the hybrid approach can provide for deep semantic annotations, which are very difficult (or even impossible) to obtain from analysis w.r.t. a single modality.

Our logic-based approach to multimedia interpretation is declarative in the sense that the interpretation process is not 'hard-coded' in program code but specified in the background knowledge. The background knowledge is not only used for representing knowledge but also serves as an executable specification such that formal inference services are exploited to automate the multimedia interpretation process.

In the remainder of this chapter, we engineer a multimedia interpretation system step by step. We begin with the identification of an appropriate formalism, Description Logics (DLs) augmented with rules, and present necessary preliminaries in Section 3.1. Having identified DLs augmented with rules as the formalism of choice, we introduce our multimedia interpretation process including the analysis, interpretation and fusion steps in Section 3.2. The formalization of ABox abduction in DLs is the topic of Section 3.3. In Section 3.4 we present an interpretation algorithm that exploits abduction as the key inference service. Finally, in Section 3.5 we present an algorithm for the fusion of modality-specific interpretations in order to obtain interpretations of multimedia documents. It should be noted that this chapter presents the general interpretation and fusion framework, whereas application specific details and hence the application of the framework are presented in Chapter 4.

3.1 Knowledge Representation Formalisms

There exist several approaches in the literature that aim to improve multimedia retrieval by assisting humans in generating semantic annotations of multimedia documents [Boß08]. In contrast to these approaches, the work presented in this thesis aims to automate the generation of semantic annotations. Furthermore, the majority of these approaches consider humans as the only beneficiaries of the semantic annotations.

Our approach focuses on the integration of semantic information from different modalities in order to gain a maximum of information about multimedia documents. In addition, our approach considers both humans and software agents as potential clients of multimedia retrieval. As a result, in our approach, the knowledge representation formalism has to serve as a foundation for defining and executing automated processes such as interpretation of surface-level information, integration of information from different modalities and semantic retrieval of information. Therefore the knowledge representation formalism plays a key role and has to be chosen carefully.

The unified modeling language (UML) and entity relationship (ER) models are widely-used representation formalisms in software engineering and database modeling respectively. In the context of semantic applications *vocabularies*, *taxonomies* and *ontologies* are established knowledge representation formalisms.

A *vocabulary*, also known as *terminology*, is a set of terms which can be associated with the entities from the domain of discourse [McG03]. A *taxonomy* is an organized vocabulary. The organization of a taxonomy is generally hierarchical. For example, SNOMED is a widely-used taxonomy that covers information from the medical domain such as diseases, findings, procedures etc. [CR80].

The term *ontology* has its origins in philosophy where an ontology is considered as a systematic account of existence and is often associated with epistemology [Gru93b]. In the context of computer science, in particular AI, an often-cited definition of an ontology is Gruber's 'specification of a conceptualization' [Gru93a]. Informally speaking, an ontology is about concepts and relationships, and is used to represent the knowledge of a domain in a declarative formalism. A formal ontology contains not only representational terms such as names for objects and relations between objects, but also axioms that constrain the interpretation of these terms. Therefore, a formal ontology provides for the semantics of a domain.

Formal ontologies are particularly suitable for the development of a declarative multimedia interpretation approach for a number of reasons:

- Ontologies enable concept-based modeling of the domain knowledge. A domain expert can model the domain knowledge in terms of concepts and relationships.

From a cognitive point of view, it is commonly believed that knowledge representation in ontologies is easier compared to knowledge representation in full first-order logic, since nowadays popular ontology languages such as DLs offer convenient, class-based, variable-free syntax for first-order logic axioms.

- Ontologies are high-level, programming-language-independent specifications. They do not rely on programming-language-specific data structures. This allows ontologies to be shared, modified and reused by various applications.
- Nowadays ontology management tools that support creation, editing, viewing and saving of ontologies are available. In addition, there exist software systems that support formal inference services on ontologies. As a result of the enduring research and development efforts, today these software tools are mature enough for employment in practice.
- The end product of most information extraction tools are annotations that represent the surface-level information identified in a multimedia document. The annotations describe a multimedia document, generally, in terms of a vocabulary. In the context of multimedia interpretation, existing information extraction tools can easily be adapted to exploit ontologies instead of vocabularies when generating semantic annotations.

We have chosen *Description Logics* (DLs) as the knowledge representation formalism in this work. DLs are a family of formal representation languages with well-understood model-theoretic semantics and computational properties. Most DLs are fragments of first-order logic, and inference in DLs used in practice is decidable. This is important since unlike other formalisms such as *Prolog*, in principle, every request to a DL system is guaranteed to terminate. DL systems have matured over years and have successfully been applied to numerous problems such as conceptual modeling, software engineering and information integration [BCM$^+$03]. In addition, DLs are the underlying basis of the Web Ontology Language OWL [PSHH03], a language recommended by the World Wide Web Consortium W3C for ontology representation in the Semantic Web [Wor09]. Compared to other knowledge representation formalisms, DLs provide for better tool support and have a larger developer and user community.

However, as we will discuss later in more detail, DLs alone do not provide the necessary expressivity for building a multimedia interpretation system, and therefore, we have chosen to augment our knowledge representation formalism through the integration of DLs with rules.

In the rest of this section we present the preliminaries of DLs and logic programming. We start with the specification of syntax and semantics of DLs, and continue with the presentation of interesting inference problems in DLs. Later, we address the limitations of DLs in building practical applications that require the augmentation of DLs with rules. We also introduce syntax and semantics of logic programming in which rule-based formalisms are grounded.

3.1.1 Introduction to Description Logics

Description Logics (DLs), a family of knowledge representation formalisms with a formally defined syntax and semantics, constitute the basis of the web ontology language OWL. The language of DLs consists of *constants*, *concepts* and *roles*. Constants denote objects, also known as individuals. Concepts denote sets of individuals, whereas roles denote relationships between pairs of individuals.

DLs have been designed to optimize the trade-off between expressive abilities and complexity of reasoning [BCM$^+$03]. There exists various DLs with different expressivity. In [SSS91], Schmidt-Schauss and Smolka introduce a naming scheme for DLs with respect to the set of constructors they allow. They introduce the description logic $\mathcal{ALC}$ as the basic language. The name $\mathcal{ALC}$ stands for Attributive concept Language with Complement. In the following we introduce syntax and semantics of the description logic $\mathcal{ALCQ}(\mathcal{D})$, which extends $\mathcal{ALC}$ by additional representation means.

Syntax and Semantics

The basic building blocks of DLs are concepts, which can be considered as variable-free logical formulas. Concepts can be atomic or complex. Complex concepts are built using concept constructors and atomic concepts. For example, consider the following expression:

$Person \sqcap \exists hasRanking.Ranking$

In this example from the athletics domain *Person* and *Ranking* are atomic concepts, *hasRanking* is a so-called atomic role, and the expression $\exists hasRanking.Ranking$ is a complex concept. The sign $\sqcap$ denotes conjunction and is a concept constructor.

In this work, we focus on the description logic $\mathcal{ALCQ}(\mathcal{D})$, Attributive Language with Complement, Qualified number restrictions and concrete Domains. In the following we use A and R for atomic concepts and atomic roles, respectively. Descriptions of complex concepts C, D in $\mathcal{ALCQ}(\mathcal{D})$ are defined inductively using concept constructors shown in Table 3.1.

Syntax	Constructor
A	atomic concept
$C \sqcap D$	conjunction
$C \sqcup D$	disjunction
$\neg C$	negation
$\exists R.C$	qualified existential restriction
$\forall R.C$	value restriction
$\exists_{\leq n} R.C$	qualified maximum restriction
$\exists_{\geq n} R.C$	qualified minimum restriction

Table 3.1: Constructors for building complex concepts in $\mathcal{ALCQ}(\mathcal{D})$

In addition, we introduce the concepts *top* ($\top$) and *bottom* ($\bot$) as abbreviations of $A \sqcup \neg A$ and $A \sqcap \neg A$.

In DLs, concepts are given a set-theoretic interpretation. Each concept denotes a subset of the domain objects. The set of domain objects denoted by a concept is the extension or the semantics of that concept. More formally, an interpretation $\mathcal{I}$ is a tuple $(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ where $\Delta^{\mathcal{I}}$ is a non-empty set of domain objects, and $\cdot^{\mathcal{I}}$ is an interpretation function. The interpretation function $\cdot^{\mathcal{I}}$ assigns to every atomic concept A a subset of $\Delta^{\mathcal{I}}$ such that $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, and to every atomic role R a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ such that $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. For complex concept descriptions, the interpretation function is extended as shown in Figure 3.2, where $\sharp M$ denotes the cardinality of the set M. For top and bottom concepts it holds that $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$ and $\bot^{\mathcal{I}} = \{\}$, respectively.

A model of a concept C is an interpretation that assigns a non-empty set $C^{\mathcal{I}}$ to the concept C. Given an interpretation $\mathcal{I}$, the set $C^{\mathcal{I}}$ is the extension of the concept C, and

$$
\begin{array}{rcl}
(C \sqcap D)^{\mathcal{I}} & = & C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
(C \sqcup D)^{\mathcal{I}} & = & C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
(\neg C)^{\mathcal{I}} & = & \Delta^{\mathcal{I}} \backslash C^{\mathcal{I}} \\
(\exists R.C)^{\mathcal{I}} & = & \{x \mid \text{there exists y} \in \Delta^{\mathcal{I}},\ (x,y) \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\} \\
(\forall R.C)^{\mathcal{I}} & = & \{x \mid \text{for all y} \in \Delta^{\mathcal{I}}, \text{ if } (x,y) \in R^{\mathcal{I}} \text{ then } y \in C^{\mathcal{I}}\} \\
(\exists_{\leq n} R.C)^{\mathcal{I}} & = & \{x \mid \sharp\{y \mid (x,y) \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\} \leq n\} \\
(\exists_{\geq n} R.C)^{\mathcal{I}} & = & \{x \mid \sharp\{y \mid (x,y) \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\} \geq n\}
\end{array}
$$

Figure 3.2: Interpretation of complex concept descriptions

the set $R^{\mathcal{I}}$ is the extension of the role R.

In many practical scenarios, besides abstract individuals also concrete datatypes and values are required. For example, in the athletics domain we may want to describe the concept *JuniorAthlete* to represent athletes who are younger than 18 years. The use of natural numbers in this example requires so-called *concrete domains.*

According to Baader and Hanscke [BH91], concrete domains are formally defined as follows: A concrete domain $\mathcal{D}$ is a pair $(\Delta^{\mathcal{D}}, \Phi^{\mathcal{D}})$, where $\Delta^{\mathcal{D}}$ is the domain of concrete objects, and $\Phi^{\mathcal{D}}$ is a set of predicate names. The sets $\Delta^{\mathcal{D}}$ and $\Delta^{\mathcal{I}}$ are disjoint. A predicate P of arity n is denoted as P^n. The interpretation function $\cdot^{\mathcal{I}}$ is extended to map each predicate name P^n from $\Phi^{\mathcal{D}}$ to a subset $(P^n)^{\mathcal{I}}$ of $(\Delta^D)^n$. A concrete domain is called *admissible* if and only if the set of predicate names $\Phi^{\mathcal{D}}$ is closed under negation and contains a name $\top_D$ for $\Delta^{\mathcal{D}}$, and the first-order satisfiability problem $P_1^{n_1}(x_{11}, \ldots, x_{1n_1}) \wedge \ldots \wedge P_k^{n_m}(x_{k1}, \ldots, x_{kn_m})$ is decidable, where x_{jk} is a variable.

The interpretation function is extended in such a way that *concrete domain attributes* are interpreted as partial functions from $\Delta^{\mathcal{I}}$ to $\Delta^{\mathcal{D}}$. If AT_1 to AT_n are concrete domain attributes and P^n is an n-ary concrete domain predicate, then the expression $\exists(AT_1 \ldots AT_n).P^n$ is called the *concrete domain predicate exists constructor*. The semantics of this constructor is given by $(\exists(AT_1 \ldots AT_n).P^n)^{\mathcal{I}}=\{x \mid$ there exists $c_1, \ldots, c_n \in \Delta^{\mathcal{D}}$ such that $(x, c_1) \in AT_1^{\mathcal{I}}, \ldots, (x, c_n) \in AT_n^{\mathcal{I}}$ and $(c_1, \ldots, c_n) \in (P^n)^{\mathcal{I}}\}$.

A DL knowledge base or ontology $\mathcal{O}$ is defined as a tuple consisting of a *TBox* $\mathcal{T}$ and an *ABox* $\mathcal{A}$: $\mathcal{O} = (\mathcal{T}, \mathcal{A})$. Each DL knowledge base has a terminological (TBox) and an assertional (ABox) part. A TBox represents intensional knowledge about the domain in terms of a finite set of terminological axioms. An ABox represents extensional knowledge about specific domain objects in terms of a finite set of assertional axioms.

In a TBox, subsumption and equality relations between concepts can be defined using so-called terminological axioms. A *generalized concept inclusion* axiom (or *GCI* in short) is a terminological axiom of the form:

$$C \sqsubseteq D$$

where C and D are concepts. An interpretation $\mathcal{I}$ satisfies a GCI axiom $C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. An interpretation $\mathcal{I}$ is a model of a TBox $\mathcal{T}$ if it satisfies all axioms in $\mathcal{T}$.

For example, if *Athlete* and *Person* are concepts from the athletics domain, the GCI axiom $\mathit{Athlete} \sqsubseteq \mathit{Person}$ states that the concept *Athlete* is a specialization of the concept *Person*, and thus every domain object in the interpretation of the concept *Athlete* must also be in the interpretation of the concept *Person*. In DL terminology, the concept *Person* is said to subsume the concept *Athlete*. *Person* is called the subsuming concept or subsumer, and *Athlete* the subsumed concept or subsumee.

$C \equiv D$ is another terminological axiom, also known as *concept definition*, used as an abbreviation for two GCI axioms, namely $C \sqsubseteq D$ and $D \sqsubseteq C$. It states that the interpretations of the concepts C and D contain the same domain objects. For example, the concept *JuniorAthlete* can be defined as

$$\mathit{JuniorAthlete} \equiv \mathit{Athlete} \sqcap \exists(\mathit{hasAge}).\leq_{18}$$

where $\leq_{18}(x)$ is a unary concrete domain predicate and *hasAge* is a concrete domain attribute.

In an ABox, knowledge about individuals and their properties can be asserted using assertional axioms as shown in Table 3.2 where i and j are individuals and $cd \in \Delta^{\mathcal{D}}$ (it is assumed that $cd^{\mathcal{I}} = cd$ for all $cd \in \Delta^{\mathcal{D}}$ hence the name *concrete* domain).

Syntax	Axiom Name
$A(i)$	concept assertion
$R(i, j)$	role assertion
$AT(i, cd)$	attribute assertion
$\mathit{same\text{-}as}(i, j)$	same-as assertion

Table 3.2: Assertional axioms

An interpretation $\mathcal{I}$ satisfies the concept assertion $A(i)$ if $i^{\mathcal{I}} \in A^{\mathcal{I}}$. An interpretation $\mathcal{I}$ satisfies the role assertion $R(i,j)$ if $(i^{\mathcal{I}}, j^{\mathcal{I}}) \in R^{\mathcal{I}}$. An interpretation $\mathcal{I}$ satisfies the attribute assertion $AT(i, cd)$ if $(i^{\mathcal{I}}, cd) \in AT^{\mathcal{I}}$. An interpretation $\mathcal{I}$ satisfies the same-as assertion $\textit{same-as}(i,j)$ if $i^{\mathcal{I}} = j^{\mathcal{I}}$. An interpretation that satisfies an assertion is called a model of the assertion.

An interpretation $\mathcal{I}$ is a model of an ABox $\mathcal{A}$ with respect to a TBox $\mathcal{T}$ if it is a model of $\mathcal{T}$ and satisfies all assertions in $\mathcal{A}$. We say that the individual j is an R-filler of the individual i, if $(i^{\mathcal{I}}, j^{\mathcal{I}}) \in R^{\mathcal{I}}$ holds in all models of $\mathcal{O}$.

For example, assume an image depicting a high jump trial, where a person's body is adjacent to a horizontal bar. This athletics scene can be represented as an ABox that contains the following assertions:

$\textit{PersonBody}(\textit{pb}_1)$
$\textit{HorizontalBar}(\textit{hb}_1)$
$\textit{adjacent}(\textit{pb}_1, \textit{hb}_1)$

where *PersonBody* and *HorizontalBar* are concepts, and *adjacent* is a role. The first two axioms are concept assertions stating that the corresponding domain objects for the individuals $\textit{pb}_1$ and $\textit{hb}_1$ belong to the interpretation of the concepts *PersonBody* and *HorizontalBar*, respectively. The third axiom states that $\textit{hb}_1$ is a filler of the role *adjacent* for $\textit{pb}_1$.

Inference Services in DLs

In many practical applications users are not only interested in storing knowledge in a DL knowledge base but also want to reason about the knowledge, i.e. exploit the knowledge. In several cases, users might want to query the knowledge base, e.g. they might want to know whether the stored knowledge is contradictory. In DLs a number of tasks for exploiting knowledge can be defined as formal decision problems. To solve these problems, practical DL systems, so-called *DL reasoners*, implement inference algorithms and offer them as so-called inference services. Standard DL inference services deal with the following tasks, which are described in detail:

Knowledge base consistency testing: Testing the consistency of knowledge bases is a standard inference service. An ABox $\mathcal{A}$ is consistent w.r.t. a TBox $\mathcal{T}$ if there exists a model $\mathcal{I}$ for $\mathcal{A}$ which is also a model of $\mathcal{T}$. A knowledge base $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ is consistent if there exists a model for $\mathcal{A}$ which is also a model for $\mathcal{T}$. Knowledge bases that are not consistent are called *inconsistent*.

Subsumption testing: Given two concepts C and D, the task of subsumption testing is to check whether $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ holds in all models of $\mathcal{T}$. In this case, it is said that D subsumes C, or alternatively, C is subsumed by D. An important reasoning task is the computation of 'direct' subsumption relationships between every pair of concept names (atomic concepts) in a given TBox in order to obtain the subsumption hierarchy between concept names, also known as taxonomy, in a knowledge base. For a given concept C, the most-specific concept names in $\mathcal{T}$ that subsume C are called the *parents*. The *children* of the concept C are the most-general concept names that are subsumed by C. The computation of the parents and children of every concept name in a TBox is called *TBox classification*.

Instance checking: Instance checking deals with finding whether an individual is an instance of a certain concept. An individual i is an instance of a concept C with respect to a knowledge base $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ if $i^{\mathcal{I}} \in C^{\mathcal{I}}$ for all models of $\mathcal{T}$ and $\mathcal{A}$. The most-specific concept names in a TBox $\mathcal{T}$ that an individual is an instance of are called the *direct types* of the individual with respect to the knowledge base. The computation of the direct types of all individuals in $\mathcal{A}$ w.r.t. $\mathcal{T}$ is called *ABox realization*.

Concept satisfiability testing: The concept satisfiability testing inference service answers the question whether a concept defined in a knowledge base can have instances. A concept C is satisfiable w.r.t. a TBox $\mathcal{T}$ if there exists a model $\mathcal{I}$ of C such that $C^{\mathcal{I}} \neq \{\}$. Checking the consistency of all concept names in a TBox (possibly without computing the subsumption hierarchy of concept names) is called *TBox coherence check*.

In addition to standard inference services, practical applications require also additional inference services. For example, *ABox entailment* and *instance retrieval* inference services are important inferences for many practical applications. Therefore, we introduce them briefly.

ABox entailment: The ABox entailment inference service answers the question whether an ABox, i.e. a set of assertions, is entailed by a knowledge base. An ABox $\mathcal{A}'$ is *entailed* by (or logically follows from) a knowledge base $\mathcal{O} = (\mathcal{T}, \mathcal{A})$, denoted with $\mathcal{T} \cup \mathcal{A} \models \mathcal{A}'$, if all models of $\mathcal{T}$ and $\mathcal{A}$ are also models of $\mathcal{A}'$.

Instance retrieval: Another important inference service is *instance retrieval*, which is used to retrieve all individuals that are instances of a given concept C or all pairs of individuals, which are related via a given role R.

The instance retrieval inference service supports the retrieval of all individuals for a concept or role name from a knowledge base. State-of-the-art DL reasoners offer an additional retrieval service by supporting so-called *conjunctive queries*. *Conjunctive queries* provide for an expressive query language for DLs with capabilities that go beyond instance retrieval.

Conjunctive queries: A conjunctive query

$\{(\underline{H}) \mid Q_1(\underline{V_1}), \ldots, Q_n(\underline{V_n})\}$

consists of a head $\underline{H}$ and a body $Q_1(\underline{V_1}), \ldots, Q_n(\underline{V_n})$ where $\underline{H}$ is a sequence of variables, $\underline{V_1}$, $\underline{V_n}$ are sequences of variables and individuals, and the body $Q_1(\underline{V_1}), \ldots, Q_n(\underline{V_n})$ denotes a conjunction of query atoms. Query atoms can be concept query atoms of the form $C(X)$ or role query atoms of the form $R(X, Y)$ where C is a concept name, R is a role name, and X and Y are variables. The variables in the head are called *distinguished* or *answer* variables, and define the format of the query result. The answer variables have to be bound to satisfying individuals. Every answer variable must appear at least once in the body of the query. The variables that appear only in the body are called *non-distinguished* variables. Unlike answer variables, non-distinguished variables are considered as existentially quantified, i.e. they do not have to be bound to individuals, rather, it suffices that in all models a satisfying domain object *exists*.

Informally speaking, the answer to a conjunctive query is a set of tuples representing bindings for the distinguished variables. If all variables in the query are replaced with the corresponding individuals from the query answer, every query atom in the body is true in each model of the knowledge base.

Currently, most DL reasoners support only so-called *grounded* conjunctive queries, which differ from the *standard* conjunctive queries introduced above by requiring that also non-distinguished variables have to be bound to individuals. Thus, from a formal point of view there is no difference between distinguished and non-distinguished variables in grounded conjunctive queries. For grounded conjunctive queries the standard semantics can be obtained (only) for so-called tree-shaped queries by using existential restrictions in query atoms (see [GHLS07], [HSTT00], [WM06] for a detailed discussion). In this work, we focus on (unions of) grounded conjunctive queries, which are supported by contemporary DL reasoners such as KAON2 [HMS04], Pellet [SP06] and RacerPro [HM01].

To describe the semantics of grounded conjunctive queries, the notion of a *variable substitution* σ is defined as a function which maps variables to individuals from a knowledge base $\mathcal{O}$, and every individual from $\mathcal{O}$ to itself. More formally, let $Vars = \{X, Y, \ldots\}$ be a set of variables, where variables are denoted by capital letters and $Inds = \{i, j, \ldots\}$ be a set of individuals, where individuals are denoted by lowercase letters. The *variable substitution* function σ assigns to every variable in $Vars$ an individual from $Inds$, and maps every individual in $Inds$ to itself. For example, $\sigma(X) = i$, $\sigma(Y) = j$, $\sigma(i) = i$.

Note that the variable substitution function σ has to be defined for individuals as well, since σ will later be applied to sequences containing both variables and individuals. σ is extended to sequences in the obvious way, i.e. by element-wise application: $\sigma(\underline{V_1}) = \sigma(\langle X \rangle)$ is defined as $\langle \sigma(X) \rangle$, and $\sigma(\underline{V_2}) = \sigma(\langle X, Y \rangle)$ is defined as $\langle \sigma(X), \sigma(Y) \rangle$.

A variable substitution σ is called *admissible* for a conjunctive query $\{(\underline{H}) \mid Q_1(\underline{V_1}), \ldots, Q_n(\underline{V_n})\}$ if σ is defined for all elements in V_1, …, V_n. Answering a grounded conjunctive query with respect to a knowledge base $\mathcal{O}$ then means finding all admissible variable substitutions such that $\mathcal{O} \models \{Q_1(\sigma(\underline{V_1})), \ldots, Q_n(\sigma(\underline{V_n}))\}$ holds. Here the query is grounded first, meaning that the variable substitution σ is applied first, before checking whether $\mathcal{O} \models \{Q_1(\sigma(\underline{V_1})), \ldots, Q_n(\sigma(\underline{V_n}))\}$ holds. Note that a grounded query atom, e.g. $Q_i(\sigma(\underline{V_i}))$, is in fact an ABox assertion, and $\mathcal{O} \models Q_i(\sigma(\underline{V_i}))$ if and only if $\mathcal{O} \models \{Q_i(\sigma(\underline{V_i}))\}$ (ABox entailment has been defined before). The *answer* of a conjunctive query is given as the set of tuples: $\{\sigma(\underline{H}) \mid \mathcal{O} \models Q_1(\sigma(\underline{V_1})), \ldots, Q_n(\sigma(\underline{V_n}))\}$. If the head of a conjunctive query is empty, i.e. $\underline{H}$ is of length zero, then the conjunctive query is called a *Boolean conjunctive query*. If an admissible variable substitution exists such that $\mathcal{O} \models \{Q_1(\sigma(\underline{V_1})), \ldots, Q_n(\sigma(\underline{V_n}))\}$ holds, then the answer to the Boolean conjunctive query is *true*, and *false* otherwise.

To give an example, we consider the ABox

$\mathit{HighJump}(\mathit{hj_1})$
$\mathit{SportsTrial}(\mathit{st_1})$
$\mathit{HighJumper}(\mathit{h_1})$
$\mathit{Athlete}(\mathit{a_1})$
$\mathit{hasParticipant}(\mathit{hj_1}, \mathit{h_1})$
$\mathit{hasParticipant}(\mathit{st_1}, \mathit{a_1})$

and the TBox

$\mathit{HighJump} \sqsubseteq \mathit{SportsTrial}$
$\mathit{HighJumper} \sqsubseteq \mathit{Athlete}$

The conjunctive query

$\{(X, Y) \mid \mathit{SportsTrial}(X), \mathit{hasParticipant}(X, Y), \mathit{Athlete}(Y)\}$

is used to retrieve all sports trials and the participating athletes from this knowledge base. The answer to this query is as follows:

$\{(\mathit{hj_1}, \mathit{h_1}), (\mathit{st_1}, \mathit{a_1})\}$

As this simple example demonstrates, query answering in DL systems requires the consideration of the axioms in a knowledge base. In this example the subsumption relations are considered to answer the query. Due to the fact that disjunctive information can be represented in DLs as well, query answering becomes even more involved, since multiple models have to be considered for query answering and some kind of case analysis is required. To illustrate the later point, assume that the ABox above contains also the following assertions:

$\mathit{Jumper}(\mathit{j_1})$
$\mathit{SportsTrial}(\mathit{st_2})$
$\mathit{hasParticipant}(\mathit{j_1}, \mathit{st_2})$

and the TBox above contains also the following axioms:

$PoleVaulter \sqsubseteq Athlete$
$Jumper \equiv (HighJumper \sqcup PoleVaulter)$

Now, in order to recognize that the pair (st_2, j_1) is also an answer to the query, the system has to perform case analysis. In one model it holds that j_1 is a *HighJumper* and thus an *Athlete*, whereas in the other model j_1 is a *PoleVaulter* and thus an *Athlete*. Hence in all models it holds that j_1 is an *Athlete*.

Unlike DL systems, standard relational database systems do not perform case analysis and consider only one model for query answering. In fact, a database instance represents exactly one interpretation, whereas an ABox represents many interpretations (its models). As shown by Reiter in [Rei84], it is equivalent to consider a database instance as a set of ground facts (in the first-order logic sense). In principle, this set of ground facts has multiple models. However, due to the absence of negative and disjunctive information it is sufficient to consider only the minimal Herbrand model for query answering. Since the database contains only positive information and its model is minimal, the corresponding negative information is maximized. This means that the modeled positive information is assumed to be complete. This is referred to as the *Closed World Assumption* (CWA). As a consequence of the CWA, the information not contained in the database is assumed to be false.

Database systems employ the CWA by means of a non-monotonic inference rule called *negation-as-failure*. In this inference rule, failure to prove some fact p automatically proves $\neg p$.

In contrast to database systems, DL systems employ the *Open World Assumption* (OWA), which means that the information about the domain may be incomplete. This is why query answering in DLs is a computationally much harder task compared to query answering in database systems.

DL systems support a larger class of logical axioms than database systems. For example, in DL systems *general truths* can be expressed as universally quantified sentences. For instance, the general truth that all participants of a sports trial are athletes is expressed with the GCI $SportsTrial \sqsubseteq \forall hasParticipant.Athlete$. Furthermore, in DL systems the existence of unnamed domain objects can be implied by the knowledge base. Unnamed domain objects are domain objects for which no individual exists in the ABox. For example, the GCI $\mathit{SportsTrial} \sqsubseteq \exists \mathit{hasParticipant}.\mathit{Athlete}$ states that each sports trial has an athlete as a participant. Notice that the existence of an individual in the ABox representing this domain object, namely an athlete, is not required.

Database systems also make the UNA, which means that different individuals are necessarily mapped to different domain objects. Contrary to database systems, DL systems usually do not make the UNA. In some DL reasoners, such as RacerPro, UNA is optional. Without the UNA, different individuals can be mapped to the same domain object. In this case, the equality of two individuals can be stated explicitly as follows: $\mathit{same\text{-}as}(\mathit{athlete}_1, \mathit{athlete}_2)$.

Merits and Shortcomings of Description Logics

The main arguments for using description logics as the logic-based formalism for multimedia interpretation can be listed as follows:

Expressive Language: DLs are based on first-order logics. In general, DLs can be considered as a restricted first-order logic language over unary and binary predicates with a special, variable-free syntax. However, there exist also description logics, such as $\mathcal{DLR}$, that allow for n-ary relations [CGL08]. Most DLs are decidable fragments of first-order logic with equality. DLs have been designed with a special focus on the trade-off between expressivity, decidability and sometimes complexity of reasoning. The computational properties of DLs have been studied extensively [BCM+03].

Standardization: DLs constitute the underlying basis of OWL, a language recommended by the W3C for ontology representation in the Semantic Web. Therefore several standards exist for interacting with DL knowledge bases. For example OWL-QL [FHH04] and SPARQL-DL [SP07] are languages for querying OWL knowledge bases. DIG [TBK+06] and its successor OWLLink [TLN+08] are protocols for communication with DL reasoners. They facilitate clients not only to query ontologies but also to configure reasoners and load ontologies. OWL API [HBN07] is a Java interface and implementation for OWL that can be exploited to develop semantic applications that require reasoning in OWL ontologies.

Tool Support: Besides several standards also a number of stable software systems exist for DLs. Most importantly, several DL reasoners with support for various inference services are available. KAON2, Pellet and RacerPro are examples of contemporary DL reasoners, which are used for building practical applications. In addition to DL reasoners also systems for supporting ontology design are available. Protégé-OWL [KMR04] and SWOOP [KPH05] are examples of software tools that assist ontology designers in modeling and editing domain ontologies in OWL. These tools also implement standard interfaces for interacting with DL reasoners. For example, using Protégé an ontology designer can check for the consistency of a knowledge base being designed. In this case, Protege can interact with DL reasoners, e.g. RacerPro, through the OWL-API.

Concept-Based Knowledge Representation: In cognitive psychology, semantic memory is a model of human memory organization [And76, pp. 380]. Driven by work on human memory and language processing, Quillian initiated work on semantic networks [Qui68]. Later, Minsky introduced *frames*, a variation of semantic networks where knowledge describing a particular *concept* is organized as a *frame* [Min75]. A frame contains properties (called *attributes*) and *relations* to other frames. In the same spirit as the seminal semantic network language KL-ONE, also its successor DLs support the concept-based knowledge representation paradigm. It is often argued that this paradigm is particularly suitable for modeling conceptual knowledge about a domain, since it offers a user-friendly, variable-free syntax, and the concept-based (frame-based) representation is appealing from a cognitive point of view. Different from other formalisms such as rule formalisms, in concept-based knowledge representation all axioms that belong to a certain cognitive concept are kept together.

Modeling of Incomplete Information: The OWA employed by DLs allows for the modeling of incomplete information, which is an important feature for multimedia interpretation where the available information is in most cases noisy and incomplete. Using DLs we can state not only information about known objects in a multimedia document but also general truths that are always valid in the domain of interest. For instance, in the athletics domain, we can state the general truth that all pole vault trial scenes include a pole object, which may or may not be visible in an image of the scene. Assume, for example, an image depicting a pole vault trial where a pole object is not visible. Under the OWA, the knowledge about the particular pole vault image is not in contradiction with the general truth about pole vault trial scenes, since the knowledge about the particular image may be incomplete.

Besides several merits, description logics have also some shortcomings, which have been revealed when building practical applications. For example, in [MHRS06], five important shortcomings of DLs are discussed, which can easily be addressed through the integration of DLs and logic programming:

Higher Relational Expressivity: Even though very expressible description logics exist, the relational expressivity of DLs, and therefore OWL, are not sufficient in some cases. For example, many practical applications require modeling of triangular structures. However, in DLs only tree-like structures can be modeled in the TBox, although very limited kinds of triangular structures can be modeled using the so-called *complex role-inclusion axioms* of the recently released OWL2 standard [OWL09].

For example, assume that in our example athletics ontology we model the concept $Person$ using DLs. However, we cannot model the fact that an instance of the concept $Person$ always has instances of the concepts $Body$ and $Face$ as parts and at the same time, these $Body$ and $Face$ instances must be adjacent to each other. We cannot model the concept $Person$ in the TBox because this requires expressing a 'triangle of relations' between the concepts $Person$, $Body$ and $Face$. Figure 3.3 shows a graphical representation of this example.

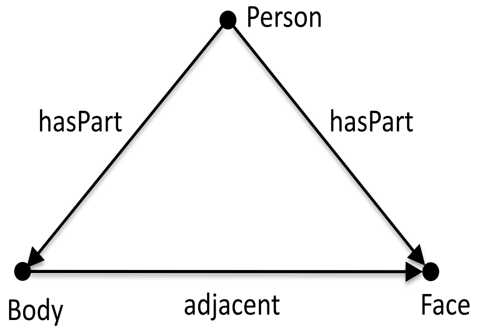


Figure 3.3: A graphical representation of the concept definition $Person$, which requires modeling of a triangular structure

It should be noted that expressive DLs support features such as transitive roles allowing for 'shortcuts' between branches of a tree. For example, in Figure 3.4 the individuals i_1 and i_2 and the individuals i_2 and i_3 are related to each other through the role R, which is depicted as continuous edges. If the role R is defined as transitive, then a 'shortcut', depicted as a dashed edge in the graph, is implied in any model of the knowledge base and, therefore, the relation R is inferred between i_1 and i_3.

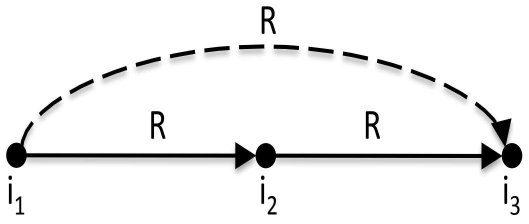


Figure 3.4: A graphical representation of an ABox with an inferred role assertion (dashed) caused by the transitive role R

The use of transitive roles is just a useful workaround for certain situations but does not solve the problem of modeling triangular structures in general. In practice, there are several situations where this workaround cannot be applied. For example, a particular role used to model a concept definition of the application domain may be non-transitive by definition. Figure 3.3 shows another example where the workaround cannot be applied since the definition of the concept *Person* requires roles that are different from each other.

Relations of Arbitrary Arity: Standard DLs only allow for modeling of roles that denote binary relationships between objects. However, when modeling ontologies for practical applications, relationships with an arity greater than two are often encountered. For example, we cannot model a ternary relation *performs* in our athletics ontology to represent the fact that an *athlete* has a certain *performance* in a particular *sports trial*, where *athlete*, *performance* and *sports trial* are concepts in the ontology. Note that some description logics such as $\mathcal{DLR}$ allow for n-ary relations [CGL08]. Even though some of the inference problems in $\mathcal{DLR}$ can be reduced to inference problems in standard DLs [HSTT99], currently, there exists no implemented DL reasoner that supports reasoning in $\mathcal{DLR}$.

Integrity constraints: Informally speaking, an *integrity constraint* is a formula that constrains the information which may be stored in a relational database. The validity of the constraints must be checked every time updates are made to the database [LST87]. Integrity constraints, which are widely used in modeling relational database schemas and object-oriented systems, are not supported by standard DLs [MHRS06]. However some proprietary and system-specific solutions exists, e.g. integrity constraints can be expressed as certain rules in RacerPro. Also certain nonstandard DLs such as autoepistemic DLs [DNR97] allow for the formulation of integrity constraints. Unlike the database realm none of these solutions are standardized for the time being.

The Unified Modeling Language (UML) is the most widely-used modeling language in software engineering [Fow04]. For example, the UML class diagram depicted in Figure 3.5 shows a conceptual model for a simple object-oriented application involving the classes *Employee* and *Company*, and the uni-directional association *worksFor*. In UML, associations are integrity constraints which are used to model the relationships between classes of objects.

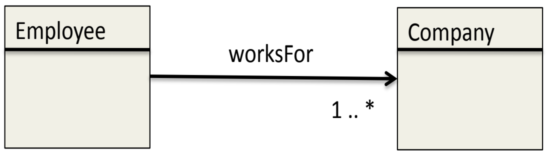


Figure 3.5: An example UML class diagram

The uni-directional association *worksFor* in Figure 3.5 means that for every instance of the *Employee* class at least one instance of the *Company* class in the *worksFor* relation must be known. The multiplicity value 1..* indicates that an employee can work for several companies. Roughly speaking, the diagram in Figure 3.5 can be translated into natural language as: ‘Each employee works for one or more companies and for each employee the company (or companies) he works for must be known’. Assume that we want to develop an object-oriented application according to the class diagram in Figure 3.5, and our application contains a single instance of the class *Employee* and no instances of the class *Company*. In this case the integrity constraint, and therefore the class diagram, is violated since our application does not contain any company instances in the *worksFor* relation with the employee instance.

Figure 3.6 shows an attempt to model the UML diagram from Figure 3.5 using standards DLs. The axiom in the TBox $\mathcal{T}$ defines a constraint on the concept *Employee* meaning: 'Each employee works for a company'. However, in the case of a DL knowledge base, where information is incomplete, this constraint means that each employee works for a company but we do not necessarily know for which one. Therefore, an ABox $\mathcal{A} = \{Employee(Michael)\}$ is consistent w.r.t. $\mathcal{T}$ in Figure 3.6, whereas $\mathcal{A}$ would violate the UML diagram from Figure 3.5, since the company for which Michael works is not known.

$$Employee \sqsubseteq \exists worksFor.Company$$

Figure 3.6: An example TBox $\mathcal{T}$

It can be concluded that the axiom in $\mathcal{T}$ is too weak to serve as a conceptual model for the object-oriented example application discussed above. In principle, autoepistemic DLs can be used to represent the integrity constraint required in this example.

Closed World Reasoning: Many applications require the formalization of closed world reasoning over DL knowledge bases, which is not addressed in standard DLs [Ros06]. For example, in a multimedia interpretation process, under certain conditions, it may be required to assume that the surface-level information identification (see Figure 3.1) has delivered complete information about the objects in a document and we might be interested in the consequences of applying closed world reasoning on the available information.

Modeling of Exceptions: In many practical applications modeling of exceptions is essential, due to the fact that the real-world is full of exceptions. However using standard DLs we cannot express exceptions, or more generally, non-monotonic knowledge. Note that besides the standard DLs we discuss in this work that allow modeling of crisp knowledge, there exist also other DLs that are designed to deal with uncertainty and vagueness. For example, probabilistic [GL02] and possibilistic [QPJ07] extensions of

DLs have been proposed in the literature.

The shortcomings of standard DLs can be addressed through the integration of description logic with logic programming. Before discussing details of our integration approach in the next chapter, we proceed with a brief introduction to logic programming.

3.1.2 Introduction to Logic Programming

Logic programming uses the language of logic to express both data and programs. Similar to first-order logic, the language of a logic program allows for constant, function and predicate symbols. Atomic formulas (also known as *atomic sentences* or *atoms* in short) have the form $p(t_1, \ldots t_n)$, where t's are terms and p is a predicate symbol of arbitrary arity. A *literal* is either an atomic formula or the negation of an atomic formula. A *clause* is a logic formula of the form:

$L_1 \vee \cdots \vee L_n$

where each L_i is a literal. A *Horn clause* (also known as *Horn rule*) is a clause that contains at most one positive literal. A *definite clause* (also known as *definite rule*) is a clause that contains exactly one positive literal:

$A_0 \vee \neg A_1 \vee \cdots \vee \neg A_n$

Following the notational convention proposed in [NM95], definite clauses can be written as follows:

$A_0 \leftarrow A_1, \ldots, A_n$

where $n \geq 0$ and $A_0, \ldots, A_n$ are atomic formulas. All variables occurring in a formula are universally quantified over the whole formula. The atomic formula A_0 is called the *head* of the clause whereas the atomic formulas $A_1, \ldots, A_n$ are called the *body* of the clause. Formulas and clauses are called *ground* if they contain no variables. A *fact* is a definite clause with an empty body, i.e. $n = 0$. Usually, the head of a fact is a ground atomic formula. A Horn clause with an empty head, i.e. where A_0 is absent, is called a *goal clause.* A *definite program* is a finite set of definite clauses. A program is also called a rule set. A rule set is called *recursive* if the body of one rule directly of indirectly depends on the head of another rule, and *non-recursive* otherwise.

To give an example, let Π be a definite program containing the following rules:

$q(x) \leftarrow p(x)$
$r(x) \leftarrow q(x)$
$p(i) \leftarrow$
$r(j) \leftarrow$

A definite program with variables can be considered as a shorthand for the set of all ground instances of its rules, i.e., for the result of substituting variable-free terms for variables in the rules of the program in all possible ways (this process is often referred to as *grounding*). Therefore Π is a shorthand for the following set of ground instances of its rules:

$q(i) \leftarrow p(i)$
$r(i) \leftarrow q(i)$
$q(j) \leftarrow p(j)$
$r(j) \leftarrow q(j)$

The set of all ground atoms in the language of a definite program Π is called the Herbrand base of Π and denoted by HB. Note that the language of a definite program is the set of constant, function and predicate symbols that occur in the definite program. In our example Π has the following Herbrand base:

$HB = \{p(i), q(i), r(i), p(j), q(j), r(j)\}$

A *Herbrand model* of a definite program Π uses HB as the interpretation domain and is a standard first-order model of Π. There exists a unique minimal model for each definite program Π, which is the *least Herbrand model* of Π. To specify the least Herbrand model of Π in our example, it suffices to give the interpretation function $\cdot^{\mathcal{I}}$ as follows:

$p^{\mathcal{I}} = \{p(i)\}$
$q^{\mathcal{I}} = \{p(i)\}$
$r^{\mathcal{I}} = \{p(i), r(j)\}$

It can easily be observed that every Herbrand model must be a superset of this least Herbrand model. Note that, in general, the Herbrand base HB may be infinite, e.g. if Π contains function symbols. In this work, we consider function-free definite clauses only and follow a Datalog-like approach.

Datalog, a prominent query and rule language used in deductive databases and production systems, supports only definite clauses with no function symbols. In addition, Datalog requires all variables that appear in the head of a rule to appear also in the body of the same rule. Systems supporting Datalog often employ *forward-chaining*, also known as bottom-up inference. Here the name *forward-chaining* indicates that rules are processed *forward*, i.e., in the sense of the logical implication sign, from body (premise) to head (conclusion). *Prolog* is a widely-used logic programming system, and unlike Datalog Prolog supports also definite clauses with function symbols. Prolog uses resolution based inference algorithms, which work in a *backward-chaining* way, also known as top-down or goal-directed inference. Backward-chaining inference based on the SLD resolution does not always guarantee termination since inference with definite clauses with function symbols is in general undecidable [Kow74, Kow79a, Kow79b], whereas termination is guaranteed for the fixed-point based inference algorithms employed for Datalog [Ull85].

In general, there are several decidable rule formalisms for which decidability is achieved through the application of some kind of restrictions, also known as safety conditions, on the rules [Ros05]. In Datalog, any program Π must satisfy the following safety conditions:

- Each fact in Π is ground.
- Each variable that appears in the head of a rule in Π must also appear in the body of the same rule.

These safety conditions guarantee that the set of all facts that can be derived from a Datalog program Π is finite [CGT89].

In the context of Datalog, it is usually distinguished between two sets of clauses: a set of ground facts, called the *Extensional Database* (EDB), is stored in a relational database, and a Datalog program Π, called the *Intentional Database* (IDB), is given. The predicates that appear in the EDB are called *EDB-predicates*. EDB-predicates may appear in Π as well, but only in clause bodies. The predicates that appear in Π but not in the EDB are called *IDB-predicates*. As a consequence, the head predicate of each clause in Π is an IDB-predicate.

Assume that a Datalog rule $A_0 \leftarrow A_1, \ldots, A_n$ and a set of ground facts $\{F_1, \ldots, F_n\}$ are given. If a substitution θ which replaces variables with constants exists such that for each $1 \leq i \leq n$ it holds that $\theta(A_i)$=F_i, i.e. the premises of the rule are satisfied, then we can infer the fact $\theta(A_0)$, also known as the conclusion. Informally speaking, we say that a rule is *applied* to a set of ground facts or a knowledge base (KB). Notice that the inferred fact may either be a new fact or it may already be contained in the KB. We say a set of rules Π is applied to a KB in a forward-chaining way, if for every rule in Π whose premises are satisfied the conclusion of the rule is added to the KB, and this process is repeated until a fixed point is reached such that no new facts can be added to the KB.

As an example, consider the following set of ground facts stored as tuples in a relational database:

$$\mathit{EDB} = \{\mathit{parent}(\mathit{mary}, \mathit{john}), \mathit{parent}(\mathit{john}, \mathit{michael})\}$$

and the following Datalog program Π consisting of the two rules:

$$\mathit{ancestor}(i, j) \leftarrow \mathit{parent}(i, j)$$
$$\mathit{ancestor}(i, j) \leftarrow \mathit{parent}(i, k), \mathit{ancestor}(k, j)$$

that defines the ancestor relationship. As a consequence of *applying* the rules in Π to EDB, the following tuples are added to the KB:

$$\{\mathit{ancestory}(\mathit{mary}, \mathit{john}), \mathit{ancestor}(\mathit{john}, \mathit{michael}), \mathit{ancestor}(\mathit{mary}, \mathit{michael})\}$$

Notice that after the addition of these tuples to the KB, a fixed point is reached where no new facts can be inferred.

In this work we use Datalog-like rules to extend DLs with logic programming. For multimedia interpretation we exploit a background knowledge consisting of a DL TBox and a set of rules. The rules used are Datalog-like in the sense that they are function-free definite clauses, and each variable that appears in the head of a rule must also appear in the body of the same rule. However, we apply an additional restriction on the rules which requires the predicate of each rule atom to exist in the signature of the DL TBox.

In this work DLs are extended with logic programming through the application of a set of rules to a DL knowledge base. To apply a rule r to a DL knowledge base $\mathcal{O}$ consisting of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$, we first define $\theta(A_i)$ as a function that maps variables from a rule atom A_i to individuals in $\mathcal{A}$. Subsequently, we define the function *apply-rule* as follows:

```
**Algorithm apply-rule($\mathcal{T}, \mathcal{A}, r$)**
**Output**: a set of ABox assertions
$r = A_0 \leftarrow A_1, \ldots, A_n$
if there exists a substitution $\theta$ such that $\forall i \in 1, \ldots, n : \mathcal{T} \cup \mathcal{A} \models \theta(A_i)$ then
    return $\{\theta(A_0)\}$
else
    return $\{\}$
end
```

Algorithm 1: The apply-rule algorithm

To apply a set of rules $\mathcal{R}$ to $\mathcal{O}$ in a forward-chaining way, we define the function *apply-rules* as follows:

```
**Algorithm apply-rules($\mathcal{T}, \mathcal{A}, \mathcal{R}$)**
**Output**: a set of ABox assertions
$\mathcal{C} := \{\}$
$\mathcal{A}_0 := \{\}$
$\mathcal{A}_1 := \mathcal{A}$
$i := 1$
while $\mathcal{A}_i \neq \mathcal{A}_{i-1}$ do
    if there exists $r \in \mathcal{R}$ such that apply-rule($\mathcal{T}, \mathcal{A}_i, r$) $\neq \{\}$ and
    apply-rule($\mathcal{T}, \mathcal{A}_i, r$) $\not\subseteq \mathcal{A}_i$ then
        $\mathcal{C} := \mathcal{C} \cup$ apply-rule($\mathcal{T}, \mathcal{A}_i, r$)
        $\mathcal{A}_{i+1} := \mathcal{A}_i \cup$ apply-rule($\mathcal{T}, \mathcal{A}_i, r$)
    else
        $\mathcal{A}_{i+1} := \mathcal{A}_i$
    end
    $i := i + 1$
end
return $\mathcal{C}$
```

Algorithm 2: The apply-rules algorithm

We say that a DL knowledge base $\mathcal{O}$ entails the ABox assertion $P(\underline{I})$ under the Datalog semantics:

$$\mathcal{O} \models_{\mathcal{R}} P(\underline{I})$$
$$\textit{iff}$$
$$\mathcal{T} \cup \mathcal{A} \cup \textit{apply-rules}(\mathcal{T}, \mathcal{A}, \mathcal{R}) \models P(\underline{I})$$

where P is a concept or role name from $\mathcal{T}$ and $\underline{I}$ is a sequence of individuals from $\mathcal{A}$.

As we have discussed before, logic programming is often seen as a way to overcome several shortcomings of DLs [LR98]. Logic programming has some important differences to DLs: In logic programming the arity of predicates is not restricted whereas standard DLs allow for unary and binary predicates only. Furthermore, different from description logics, rules allow for modeling of triangular structures. Coming back to the example in Figure 3.3, the following rule can be used to model the concept *Person* that requires a 'triangle of relations' between the concepts *Person*, *Body* and *Face*.

$$\textit{Person}(z) \leftarrow \textit{hasPart}(z, x), \textit{Body}(x), \textit{hasPart}(z, y), \textit{Face}(y), \textit{adjacent}(x, y)$$

3.2 Overview of a Multimedia Interpretation System

As discussed in Section 2.1 in detail, we consider multimedia interpretation as a process for the computation of deep semantic annotations of multimedia documents. The multimedia interpretation process produces deep semantic annotations based on surface-level information in multimedia documents through the exploitation of background knowledge. In this context, surface-level information are surface-level semantic annotations of multimedia documents. Therefore the multimedia interpretation process takes semantic annotations as input, enriches them and provides deep semantic annotations, i.e. deep-level and surface-level information, as output.

In Section 3.1, we have chosen a logic-based formalism for multimedia interpretation, namely DLs extended with rules. This formalism is appropriate for multimedia interpretation for several reasons. As discussed in Section 3.1, the relational expressivity of standard DLs is often not sufficient for multimedia interpretation. This problem can be solved using rules. Furthermore, as we will discuss in this section, there is another important motivation for extending DLs with rules. As presented in Section 2, multimedia interpretation is not a purely deductive process but may require abduction.

Since existing DL reasoners do not support abduction, we propose rules as a way to define the space of abducibles and to achieve abductive inference.

In the formalism we have chosen for multimedia interpretation, the background knowledge consists of a TBox and a set of rules. In our approach, a TBox contains the domain knowledge and is called the *domain ontology*. We do not permit the use of individual names, so-called *nominals* in the TBox. In addition, a set of rules contain the knowledge about how to interpret multimedia documents, which contain information about the domain. Therefore we call these rules *interpretation* rules.

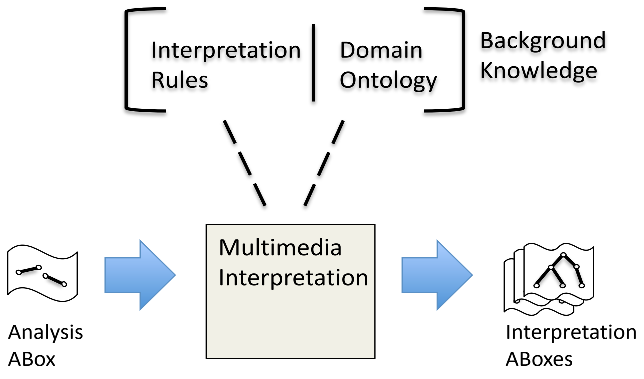


Figure 3.7: The multimedia interpretation process. Input: analysis ABox, Output: interpretation ABox(es), The background knowledge: Domain ontology and interpretation rules

Semantic annotations of a document can be represented as an ABox, i.e. as a set of assertions. Initially, semantic annotations are the result of modality-specific analysis processes and therefore we call an ABox that represents the results of an analysis process an *analysis ABox*. Multimedia interpretation is a process based on reasoning about analysis ABoxes with respect to background knowledge, the outcome being one or many (extended) ABoxes. The goal of the multimedia interpretation process is to compute ABoxes that represent deep semantic annotations. As discussed in Section 2.1, deep semantic annotations involve more abstract information, e.g. events, than the information found in analysis ABoxes.

Each ABox produced by the multimedia interpretation process represents a possible interpretation of a multimedia document and, therefore, we call such ABoxes *interpretation ABoxes*. Figure 3.7 illustrates the multimedia interpretation process where available background knowledge is comprised of the domain ontology and interpretation rules (compare with Figure 3.1). An analysis ABox contains instance and role assertions about surface-level objects. For example, an analysis ABox A_1, representing the outcome of image analysis for a sports image, might contain the following assertions:

$$A_1 = \{\mathit{Body}(i_1), \mathit{Face}(i_2), \mathit{HorizontalBar}(i_3), \mathit{adjacent}(i_1, i_2), \mathit{near}(i_1, i_3)\}$$

Besides the surface-level information from the corresponding analysis ABox, an interpretation ABox describes usually also some *deep-level information*. Hence, if no interpretations can be computed for a document, e.g. due to lack of background knowledge, the interpretation ABox of the document describes no deep-level information, and is the same as the analysis ABox. Furthermore, roughly speaking, the information in an interpretation ABox should not contradict with the background knowledge. This means that an interpretation ABox should be consistent w.r.t. the TBox.

In [NM06] and [MN08], Möller and Neumann propose the use of DLs for the representation of *aggregates* that can be used by reasoning services as building blocks for a scene interpretation process. In these works, DLs serve as a basis for formal scene interpretation processes, yet a concrete implementation of these processes is missing.

Inspired by these works, we consider *deep-level objects* as aggregates that are built on surface-level objects such that aggregates contain surface-level objects as their parts. Therefore, given surface-level objects and relations among these objects, the task of the interpretation process is to hypothesize aggregates and parthood relations between aggregates and surface-level objects. In this context, a hypothesis is a set of assertions and explains the existence of particular surface-level objects or constellations of surface-level objects in the sense that, if the hypothesis is added to the available background knowledge then a particular surface-level object or a particular constellation of surface-level objects is entailed. This is a major difference of our approach to other approaches, which aim to attach keywords (or terms) to documents in order to support the Boolean retrieval model. For example, the keyword sports trial can be attached to an image depicting a typical sports trial. Later, this image can be retrieved when searching for the keyword sports trial. However, it remains opaque why this document is believed to be related to a sports trial. In our approach the goal is to compute interpretations, which are rich symbolic structures representing deep-level information about documents. Parthood relations between aggregates and surface-level objects play an important role in here, because they set aggregates in relation with other surface-level objects and, therefore, explain why an aggregate is believed to exist in the interpretation of a document. For an image depicting a sports trial, our approach aims to compute an interpretation where a sports trial object is in relation with surface-level objects such as sports equipments or persons. Therefore the interpretation computed by our approach can be queried more conveniently. For example, a user can query for sports trial images in which a horizontal bar and a pillar are shown.

An interpretation of a document can be computed by generating hypotheses that explain observations. In the case of multimedia interpretation, observations are surface-level objects and relations among surface-level objects, i.e. surface-level information. Figure 3.8 illustrates an interpretation consisting of observations, surface-level objects and relations among surface-level objects, and explanations, aggregates and relations between aggregates and surface-level objects.

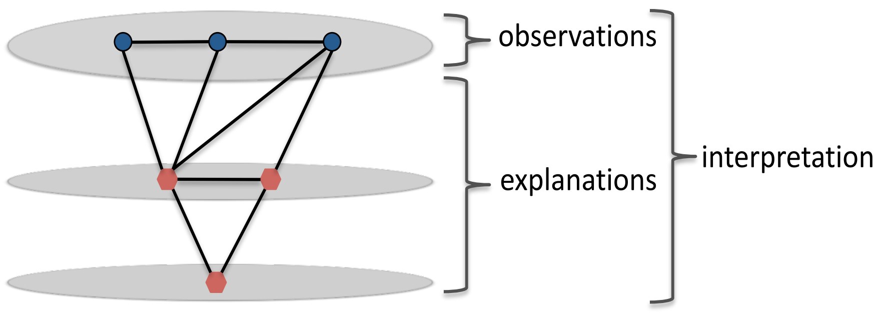


Figure 3.8: Interpretation of a document consisting of observations and their explanations

It should be noted that aggregates can have different levels of abstraction (see Figure 3.8). Some aggregates may directly be built from surface-level objects whereas others may be more abstract and build on other aggregates. The selection of the necessary level of abstraction for aggregates is an application-specific decision. In application scenarios in which very deep semantic annotations, and hence very abstract aggregates in the interpretations, are required, more abstract aggregate concepts have to be modeled in the background knowledge.

The multimedia interpretation process is required to compute rich relational structures, i.e. objects that are related to each other, and thus as many abstract aggregates, and possibly relations among them, as possible. Consequently, the multimedia interpretation process cannot terminate after explaining the surface-level objects and their relations. In fact, the computation of explanations for surface-level objects and their relations should be the first iteration of an iterative process. In subsequent iterations the explanations computed by the previous iteration should be considered as observations, and should be explained as well. If none of the observations can be explained in an iteration, then the process should terminate. In Section 3.4, along with other details of the interpretation process, we will show that the termination of the process can be guaranteed if certain conditions are fulfilled. The surface-level information together with the computed explanations constitute the interpretation of the document, i.e. the outcome of the interpretation process.

As a result of noise and the inherent ambiguity in real-world data, it is realistic to expect that many documents can be interpreted in multiple valid ways. In some cases, even a single observation may be explained in many different ways. For example, this is true if precise information about distinctive features of an observation is missing or if the background knowledge does not contain enough knowledge about the domain. Hence, the multimedia interpretation process must include a mechanism to restrict the number of explanations to those, which are more 'probable' than others, and maybe to compute 'preferred' explanations only. Even if the interpretation process is capable of computing 'preferred' explanations, there can still be situations, e.g. because of the ambiguity of available information, where the computation of multiple explanations cannot be avoided. Therefore the multimedia interpretation process should be able to compute multiple interpretations for a document, and the embedding context should be set up to handle multiple interpretations.

In order to deal with multimedia documents that are multi-modal information resources, information in any modality needs to be analyzed and interpreted. Existing analysis tools are specialized in extracting information from single modalities such as the text or visual modality. Therefore, in our approach multimedia documents are partitioned into their *segments* prior to analysis. A segment is a part of a multimedia document and contains information in one modality. For example, a web page containing an image, its caption and a text paragraph has three segments.

Modality-specific analysis and interpretation processes are then applied to these segments to obtain interpretation ABoxes, which are also modality-specific. At this

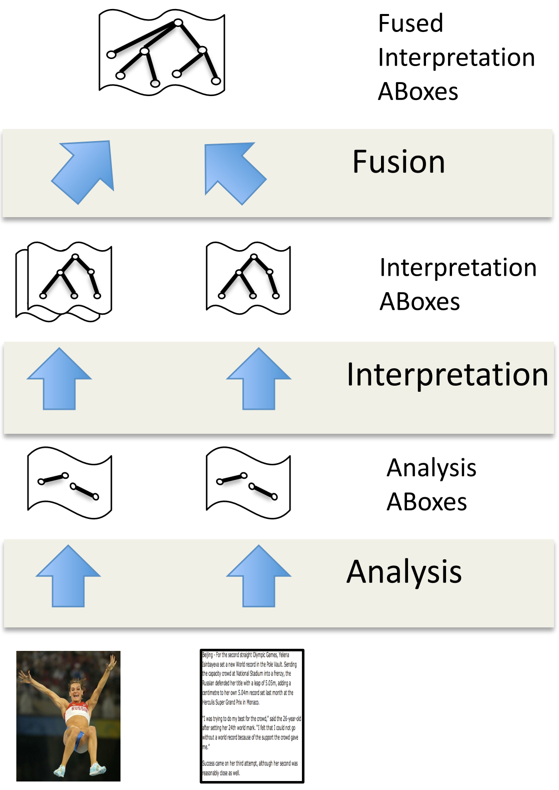


Figure 3.9: The multimedia interpretation approach including processing steps for analysis, interpretation and fusion

stage it is necessary to merge the modality-specific interpretation results in order to obtain deep semantic annotations of the whole web page. This is why we enhance our hybrid approach to include another processing step called *fusion* that follows the interpretation step. Figure 3.9 illustrates the enhanced hybrid approach to multimedia interpretation including processing steps for analysis, interpretation and fusion. Informally speaking, the goal of the fusion process is to take various interpretation results as input and provide for a *fused interpretation ABox* with respect to the initial compositional structure of the multimedia document and the background knowledge.

In the literature, the merging of information from multiple sources, possibly with different conceptual or contextual representations, is referred to as *information fusion* or *information integration*. Information integration, also known as *data integration* has been extensively studied in the context of database systems [Ull97]. Information integration is often defined as the problem of combining data residing at different resources and providing users with a unified view on this data [Len02]. The use of formal ontologies in our approach assures that the information sources we want to fuse, namely modality-specific interpretation ABoxes, have the same conceptual and contextual representation. In our context, the fusion problem differs from the general problem of information fusion, and is more similar to the *sensor data fusion* problem which is a specialization of the information fusion problem. In sensor data fusion the goal is to fuse sensor data or data derived from sensor data from different sources into a single model of some aspect of the world such that the resulting information is more appropriate than the information from one of these sources only [CY90].

We consider interpretation results from different modalities as information derived from various modality-specific sensor data. In addition, we define the fusion process with special emphasis on the peculiarities of multimedia documents. Most multimedia documents are composed of segments that contain information in one or many modalities. The information in a segment of the document usually refers to information in another segment, and different segments complement each other. For example, a web page from a news web site typically contains texts about several events and images of most important people involved. Furthermore images are usually captioned with textual information. Captioning images with text has two major advantages: First, readers can easily obtain additional textual information about persons or things shown in an image. Second, the text captioning an image serves as a link between the image and the rest of the multimedia document such that readers can easily find out paragraphs in the textual part of the multimedia document that refer to this image. Therefore we design a fusion process with respect to two important peculiarities of multimedia documents of interest:

- To fuse the interpretations of the segments of a multimedia document appropriately, the structural composition of the multimedia document has to be considered such that the references between the segments are retained.
- The information contained in segments of a multimedia document are in a sense redundant and complementary.

The fusion process should merge modality-specific interpretations of a multimedia document with respect to the structural composition of the document such that one or many fused interpretation ABoxes for the whole multimedia document is obtained. In addition, it should exploit the inherent redundancy of information in multimedia documents and make equalities between aggregates with different names but describing the same abstract entity in the real-world explicit.

Fusion of interpretation ABoxes can be achieved by merging corresponding ABoxes into a fused interpretation ABox, and adding so-called *same-as assertions* into the fused interpretation ABox if certain application-dependent conditions are fulfilled. In Section 3.5, we present an algorithm that implements fusion in our multimedia interpretation system.

This way, fusion offers two major benefits:

- Fusion facilitates the improvement of modality-specific interpretation results. During fusion, uncertainty and imprecision involved in interpretation results of a modality can be disambiguated using the additional information obtained from other modalities and the background knowledge.

- Fusion not only empowers answering of queries which require the combination of information from different modalities, but also enables the use of more intuitive queries. For instance, assume that a user wants to retrieve images of *Yelena Isinbayeva* from a multimedia repository. This requires the identification of the string *Yelena Isinbayeva* as a person name in image captions. The existence of same-as assertions between aggregates in fused interpretation ABoxes enables answering of queries that are more intuitive. In this simple example, the user can directly ask for images that depict *Yelena Isinbayeva*, instead of asking for images that are captioned with text about *Yelena Isinbayeva.*

To recapitulate, after setting up the appropriate formalism we have discussed the process underlying our multimedia interpretation system including the analysis, interpretation and fusion steps. We proceed with the formalization of ABox abduction in DLs, which is essential for explanation generation.

3.3 Formalizing ABox Abduction

As discussed in Chapter 2 in detail, explanation generation is one of the key problems in image interpretation. If a declarative solution is searched, many works share the view that abduction is the appropriate solution to the explanation generation problem. The main goal of this section is to formalize explanation generation as an abductive reasoning task which can be realized in DLs through ABox abduction.

In works that exploit logic-based formalisms, abduction is often formalized as

$$\Sigma \cup \Delta \models \gamma \tag{3.1}$$

where Σ is the background knowledge, γ an observation, and Δ a set of formulas, also known as an explanation, that together with the background knowledge entails the observation. Given the background knowledge Σ and an observation γ, the goal of abduction is to find explanations (Δs).

Formula 3.1 defines no restrictions on explanations and hence allows for explanations that are inconsistent or trivial. Furthermore, in general, for every γ there exists infinitely many explanations, since we can obtain an infinite number of explanations by adding new formulas to an explanation. Therefore, additional restrictions have to be defined on explanations to limit the number of explanations, and constrain abductive reasoning to compute only such explanations that fulfill certain criteria. In the literature, e.g. [Pau93, AL97], the most-commonly used restrictions on explanations are:

Consistency:	$\Sigma \cup \Delta \not\models \bot$
Minimality:	Δ is a 'minimal explanation' for γ
Relevance:	$\Delta \not\models \gamma$
Explanatoriness:	$\Sigma \not\models \gamma$, $\Delta \not\models \gamma$

Consistency is the most fundamental restriction on explanations, because if an explanation is inconsistent w.r.t. the background knowledge Σ then not only the observation γ but also any other formula is entailed. Informally speaking, an inconsistent explanation Δ (w.r.t. the background knowledge Σ) can 'explain' anything. An explanation Δ is *minimal* if for all other non-equivalent explanations Δ' that are comparable w.r.t. $\models$ it holds that $\Sigma \cup \Delta' \models \Delta$. Notice that an infinite number of explanations can be computed by adding new formulas to a minimal explanation. In [EKS06], an explanation Δ is called *relevant* if γ does not follow from Δ alone. Furthermore, an explanation is called *explanatory* if it is guaranteed that γ follows only from the union of Δ and Σ, but not from Δ or Σ separately.

In Formula 3.1, if DLs are used as the underlying representation and reasoning formalism, Σ is a DL knowledge base consisting of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$: $\Sigma = (\mathcal{T}, \mathcal{A})$.[1] γ is a concept or role assertion, and Δ is a finite set of concept and role assertions. Given an observation and a DL knowledge base, the goal of ABox abduction is to compute explanations, Δs, that fulfill certain criteria.

We consider ABox abduction as a non-standard retrieval inference service in DLs, where the task is to find out what should be added to the DL knowledge base in order to answer a query. Different from the standard instance retrieval service, where answers are found by exploiting the knowledge base, ABox abduction requires the hypothesization of a set of concept and role assertions, a so-called explanation, such that the query can be answered. To implement ABox abduction as a practical inference service, a mechanism for constraining the search space for explanations, i.e. for specifying the space of abducibles, is required. Rules provide for the necessary formal language for specifying the space of abducibles. Therefore, the enhancement of DLs with logic programming is a promising solution for the realization of ABox abduction as a non-standard retrieval inference service. We proposed this idea for the first time in [EKM$^+$07a], and later presented its application in building a multimedia interpretation approach in [EKM$^+$07b, EKM08a, CEF$^+$07].

In our approach, an algorithm implements ABox abduction in DLs. The consistency of an explanation (Δ) generated by the abduction inference service is guaranteed, since every computed explanation is checked for consistency, and inconsistent explanations are discarded. Furthermore, we do not require explanations to be minimal, since the abduction inference service always computes a finite number of explanations. However, we require explanations to fulfill some context-dependent criteria in order to reduce the number of explanations, and to acquire preferred explanations only. Before a detailed discussion of our ABox abduction algorithm, we present relevant work from the literature that exploits abduction for the explanation generation problem.

3.3.1 Related Work on Abduction

The explanation generation problem has been addressed by several works in the literature. All these works share the view that deductive inference is not sufficient for explaining observations. They consider abduction as the appropriate form of reasoning

[1]In equations such as $\Sigma \cup \Delta \models \bot$ we read $(\mathcal{T}, \mathcal{A})$ as $\mathcal{T} \cup \mathcal{A}$

for explanation generation. In the following we present some of these works, which study abduction in different formalisms and application domains.

Hobbs et al.: One of the earliest approaches to apply abductive inference in practice is due to Hobbs et al. [HSME88], who consider abductive inference as inference to the best explanation and apply abduction to interpret texts. The approach has been investigated in the TACITUS project [HM87] for the interpretation of casualty reports, messages about breakdown in machinery, and other texts. Abductive inference is applied in solving so-called *local pragmatics* problems such as reference resolution, compound nominal interpretation, syntactic and lexical ambiguity resolution, and metonymy resolution.

The work presented in [HSME88] is significantly different from other approaches that provide for a shallow text analysis, because it aims to interpret sentences in discourse by providing the best explanation of why the sentences would be true. To this end, a knowledge base of domain knowledge has been built, and procedures for using the knowledge base for the interpretation of discourse have been developed.

To interpret a sentence, the sentence is first transformed into the so-called *logical form* [HM87]. Hobbs et al.'s work exploits first-order logic as the formalism for interpreting texts. The logical form of a sentence is a first-order expression, and is produced through syntactic analysis and semantic translation of the sentence.

For example, consider the sentence 'The Boston office called'. This sentence poses three local pragmatics problems: i) The reference of 'the Boston office' has to be resolved. ii) The metonymy to '(Some person at) the Boston office called' has to be resolved. iii) The implicit relation between Boston and the office has to be resolved. The logical form of this sentence is as follows:

$$\exists x, y, z, e : \mathit{call}'(e, x) \wedge \mathit{person}(x) \wedge \mathit{rel}(x, y) \wedge \mathit{office}(y) \wedge \mathit{Boston}(z) \wedge \mathit{nn}(z, y)$$

where e is a calling event carried out by x, who is a person. The primed predicate $\mathit{call}'(e, x)$ means that e is the eventuality of *call* being true of x. x is related to the office y, and may be the subject of the sentence. The office y is in some unspecified relation nn to z, which is Boston. Furthermore, assume that the knowledge base contains the facts:

$$\{\mathit{Boston}(B_1),\ \mathit{office}(O_1),\ \mathit{in}(O_1, B_1),\ \mathit{person}(J_1),\ \mathit{work\text{-}for}(J_1, O_1)\}$$

the implication:

$\forall y, z : \mathit{in}(y, z) \rightarrow \mathit{nn}(z, y)$

which states that if y is in z, then z and y are in a possible compound nominal relation, and the implication:

$\forall x, y : \mathit{work\text{-}for}(x, y) \rightarrow \mathit{rel}(x, y)$

which states that if x works for y, then x and y are in a relation named rel.

To interpret a sentence, its logical form has to be proved abductively with respect to the knowledge base. In our example, $\mathit{call}'(e, x)$ is a constraint imposed by the predicate on its arguments. All conjuncts found in the logical form except $\mathit{call}'(e, x)$, which has to be assumed, can be derived from the facts in the knowledge base, and constitute a proof. The proof resolves 'the Boston office' to O_1, determines in as the implicit relation in the compound nominal 'Boston office', and expands the metonymy to 'J_1 who works for the Boston office called'. As Hobbs et al. indicate, solving local pragmatics problems is equivalent to proving the logical form and the constraints.

In [HSAM90], Hobbs et al. point out that in abduction there may be many possible explanations for an observation, and thus, criteria are needed to choose among potential explanations. Besides the obvious criterion that requires the consistency of an explanation, they propose *simplicity*, *consilience* and *specificity* as further criteria. Informally speaking, a preferred explanation should be as small as possible (simplicity), and explain as many observations as possible (consilience). Furthermore the explanation with the most-specific assumptions should be preferred (specificity), since it is most informative. It should be noted that an explanation may not fulfill all three criteria equivalently well. Consequently, to find out the preferred explanation among many possible explanations, explanations have to be evaluated with respect to all three criteria and compared with each other. For example, assume that for the observation 'An alarm sounded' two possible explanations are given, namely 'Something must have happened' and 'There must be a fire'. If no information about the type of alarm is available, we prefer the explanation 'Something must have happened' to the more specific explanation 'There must be a fire', because the first explanation hypothesizes less and thus is simpler. However, if we know that the alarm is a fire alarm, then the latter explanation is preferred, since it is most-specific and thus more informative.

For abductive inference the authors present a process, which they call an *abduction scheme*. The abduction scheme has three major features in order to apply the above-mentioned criteria: i) Goal expressions are assumable at varying costs. ii) Assumptions can be made at various levels of specificity. iii) The natural redundancy of texts can be exploited to obtain simpler explanations.

To interpret the logical form of a sentence, the abduction scheme proceeds as follows: First, every conjunct in the logical form of a sentence is assigned a cost. Note that if a conjunct in the logical form cannot be proved or derived in the logical sense from the axioms in the knowledge base, then it has to be assumed. The cost assigned to each conjunct is called *assumability cost*, and represents the relative cost of assuming that conjunct when computing an explanation. For example, compound nominal relations are given a very high cost value, since assuming them means failing to solve the interpretation problem.

Second, the assumability cost is passed back to the atoms in the body of Horn rules by assigning weights to the atoms. For example, in the Horn clause:

$$Q(x) \leftarrow P_1^{w_1}(x) \wedge P_2^{w_2}(x)$$

w_1 and w_2 are weights. According to the abduction scheme, if the cost of assuming Q is c, then the cost of assuming P_1 is w_1c, and the cost of assuming P_2 is w_2c. The assignment of cost and weight values to atoms enables the calculation of overall costs for possible explanations, and thus, the preference of one explanation over another. This is why the authors call their approach ‘weighted abduction’.

Third, *factoring* is supported. Factoring means that goal expressions may be unified, in which case the resulting expression is given the smaller of the costs of the input expressions. For example, given the goal expression:

$$\exists x, y : q(x) \wedge q(y)$$

where the costs of $q(x)$ and $q(y)$ are \$20 and \$10, respectively, factoring assumes x and y to be identical, and generates the following expression:

$$\exists x : q(x)$$

where $q(x)$ costs \$10. Factoring provides for merging of redundancies and thus leads to minimal explanations. It should also be noted that in the abduction scheme, each time an assumption is made, it is checked for consistency with respect to the knowledge base.

Hobbs et al. also present the idea of employing a type hierarchy with disjointness relations among types as a heuristics to eliminate a large number of assumptions that are likely to result in an inconsistent explanation. Therefore the implementation of the approach includes a module which defines the types that various predicate-argument positions can take on, and the likely disjointness relations among types. Even though this heuristic leads to a substantial speed-up of the interpretation process, it is also reported to cause the process to fail for certain rarely occurring sentences [HSAM90].

To recapitulate, in [HSME88] Hobbs et al. present interesting work towards the formalization of text interpretation as abduction, and address important natural language problems. However, empirical investigations of the behavior and performance of the approach with very large knowledge bases are missing.

Charniak and Goldman: In [CG91], Charniak and Goldman propose the use of probabilistic abduction for plan recognition. Plan recognition is the problem of constructing possible plans to explain a set of observations and then selecting one or more of the plans as the best explanation. Observations, typically, are actions performed by an agent. In this view, plan recognition is the problem of inferring an agent's plans from observations. The authors consider plan recognition and text understanding as a particular case of the abduction problem, where the goal is to reason from actions (effects) to plans (causes).

In [CG91], Charniak and Goldman present a probabilistic framework. The goal of the framework is to apply plan recognition for text understanding. I.e., understanding the meanings of stories by understanding the way the actions of characters in the story serve purposes in their plans. The framework enables the construction of possible plans, and the selection of the best plan among possible plans. To this end, the framework exploits *Bayesian networks* for selecting the plan, which has the highest degree of support from evidence.

The model of the plan recognition framework contains a knowledge base of facts about the world, which is expressed in a first-order notation. The language used in the framework consists of i) Terms denoting actions and physical objects. ii) Terms denoting sets of actions and objects. iii) Functions from terms to terms to indicate roles. iv) The predicates ==, *inst* and *isa*. For example, *bus2* denotes an object, *bus-* denotes a set of objects, and *agent-of* denotes a function. The statement (*agent-of drive3*) denotes the agent of the action *drive3*. The predicate *inst* denotes set membership: (*inst bus2 bus-*). The predicate *isa* denotes subsumption between two sets: (*isa bus- vehicle-*). Finally, the predicate == denotes the so-called *better name relation*, and is used for noun-phrase references. For instance, the statement (== *it20 jack2*) means that *jack*2 is a better name for *it*20.

In [CG91], the authors also introduce a software system called *Wimp3* that has been developed to experiment with the probabilistic framework. Wimp3 contains a parser that reads words from a given text one at a time. The parser deals with linguistic problems such as reference and ambiguity resolution, and produces statements in a first-order notation. The statements describe the words in the story, and the syntactic relations between the words.

The output of the parser is used by the so-called *network construction component*. The network construction component contains a set of rules that are applied in a forward-chaining way to construct a set of Bayesian networks. In Figure 3.10, a simplified version of one of the rules used for Bayesian network construction is shown.

```
(→   (inst ?i ?f) :label ?A
     (→ ← (role-inst ?f ?slot ?sfrm)
           (Exists (x :name ?sfrm)
           (and (inst ?x ?sfrm) :label ?B
                 (== '(?slot ?x) ?i) :label ?C)))
     :prob ((?B → ?A) ((t t 1))))
```

Figure 3.10: A rule used by the Wimp3 system for network construction

The rule in Figure 3.10 can be translated into natural language as follows [Gol90]:

> If we see a new action (*inst ?i ?f*), and this object could fill ?*slot* in a frame of type ?*sfrm* [(*role-inst ?f ?slot ?sfrm*)], then hypothesize that this action is explained by filling ?*slot* of an entity of type ?*sfrm* (this entity is the ?*sfrm*-explanation of ?*i*).

Given the simple sentence 'Jack went to a liquor store.', Wimp3 constructs the Bayesian network shown in Figure 3.11. Notice that conditional probability tables are not shown in Figure 3.11 for the sake of brevity.

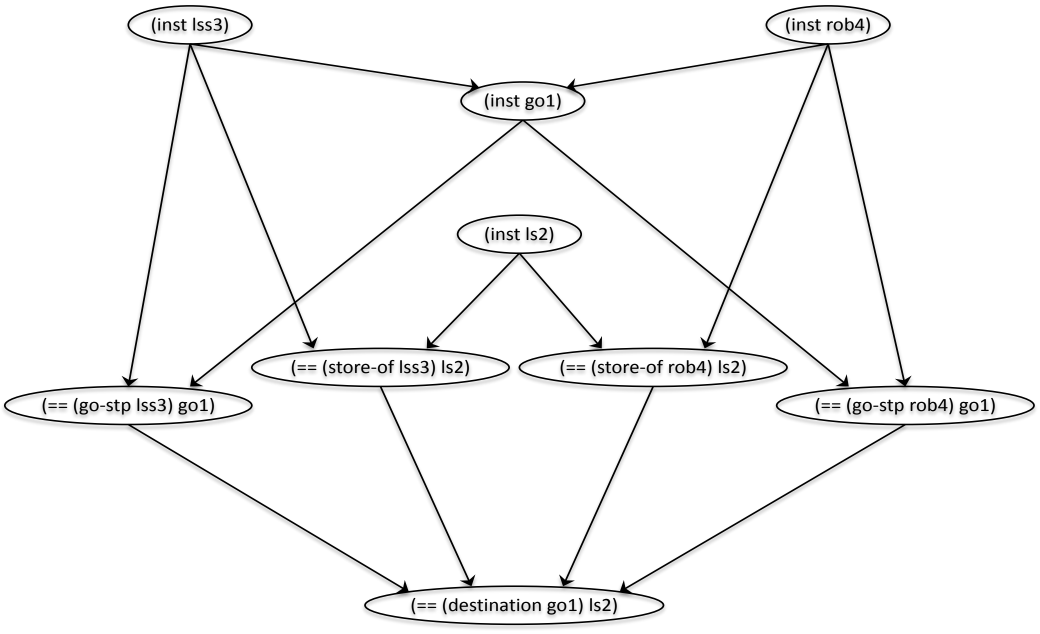


Figure 3.11: The Bayesian network constructed for plan recognition

The Bayesian network represents two plans (explanations), namely liquor store shopping ($lss3$) and robbery ($rob4$), which compete to explain the observation that someone went ($go1$) to a liquor store ($ls2$). Since shopping is a much more common event than robbery, the prior priority assigned to the shopping node is much higher than the one assigned to the robbery node. In this example, the evidence (== ($destination$ ($go1$) $ls2$) supports both plans roughly in proportion to their prior properties, because there is no other evidence. As a result, the liquor store shopping plan has a higher posterior probability, and is selected as the best plan.

Probabilistic reasoning in Bayesian networks is, in general, an NP-hard problem [Coo90]. Therefore it is essential to construct a compact Bayesian networks. A Bayesian network is compact if it is sparse. In a sparse Bayesian network most of the nodes have only a few parent nodes. However, not all domains have a sparse structure, and even for domains with a sparse structure constructing a sparse Bayesian network is not a trivial problem.

In [CG91], the authors also report scalability problems of the probabilistic framework. Tests conducted using Wimp3 are reported to show that the growth in the size of the knowledge base results in the generation of very large Bayesian networks, what the authors consider as the main drawback of their approach. In view of the advancements in computer hardware since then, we think that new empirical investigations of the approach are required.

Kakas et al.: In [RAKD06], Kakas et al. present an abductive logic programming approach for assisting clinicians in the selection of drugs for patients infected with the Human Immunodeficiency Virus (HIV). Since, for the time being, no cure for HIV exists, the goal of drug treatment is to postpone the collapse of immune function of the patient known as the Acquired Immune Deficiency Syndrome (AIDS). Even though potent drugs are available for slowing down the progression of the disease, the treatment of HIV is complicated by the propensity of the virus to accrue mutations that cause resistance to known drugs. Clinical trials of patients have been analyzed by experts and the results, so-called *genotypic drug resistance interpretation rules*, are maintained in drug resistance databases such as the Stanford HIVDB [Sha03]. In a nutshell, the genotypic drug resistance interpretation rules show which mutations cause which drug resistance regarding HIV treatment.

Different from existing HIV resistance test interpretation algorithms, the approach presented by Kakas et al. uses these rules not in a forward-chaining way to predict likely drug resistances implied by observed mutations, but it uses them backwards to explain the observed drug resistance of a patient in terms of likely mutations. Based on the information about likely mutations a patient may or may not have, the approach predicts for which other drugs the patient may be resistant as well. Given this information, clinicians can design *salvage therapies* for patients, whose first- or second-line treatments are failing, without the use of *resistance tests*. This is an important advantage of Kakas et al.'s approach compared to the use of existing resistance tests, which are expensive and require expert interpretation. In addition, existing resistance tests are ineffective in some cases, e.g. for patients with low level of viruses in their bloods ($\leq$ 500-900 copies/ml).

The authors consider abduction as a form of reasoning, which, given a goal or observations G and a logic program T, returns explanations (Δs) that logically entail the goal with respect to the logic program T. Each explanation is a set of ground atoms whose predicates are specified as *abducible.* Informally speaking, the abducibles A are predicates whose extents are incompletely specified by the program T. The abducibles may be subject to further integrity constraints IC.

Formally, an ALP theory is a triple (T, A, IC) which consists of a logic program T, a set of abducibles A, and a set of logical formulas IC which represent integrity constraints. A goal or observation G is a set of literals. An explanation Δ of G with respect to (T, A, IC) is a set of ground atoms with predicates specified in A such that the following holds:

i) $T \cup \Delta \models_{LP} G$
ii) $T \cup \Delta \models_{LP} IC$

Here $\models_{LP}$ denotes standard entailment relation of logic programming under the stable model semantics. In this abductive framework, integrity constraints IC have two roles: i) To augment the basic model in T with any partial information on the abducible predicates ii) To impose additional requirements on the hypotheses Δ.

In [RAKD06], the authors describe a software system called *in-Silico Sequencing System* (iS3) that employs Abductive Logic Programming (ALP) [KD93, KKT98] for the computation of explanations, and thus for the realization of abduction. The iS3 system is applied in the HIV clinical management context. Given a patient's treatment history, iS3 computes first sets of viral mutations the patient may be carrying, and then the drugs the patient is most likely resistant to. The objective of iS3 is to assist clinicians in defining a future treatment plan for HIV patients.

In this scenario, the logic program T used by iS3 consists of genotypic drug resistance interpretation rules. These rules, which are extracted from the drug resistance database maintained by the French AIDS research agency [BVDR+03], have the form:

$resistant\ (P, T, D) \leftarrow mutation\ (P, T, M)$

where atoms of the form $resistant(P, T, D)$ denote that patient P is resistant to drug D at time T, and atoms of the form $mutation(P, T, M)$ denote that patient P carries mutation M at time T. For example, the rule

$resistant\ (P, T, \text{‘zidovudine’}) \leftarrow mutation\ (P, T, \text{‘151M’})$

states that the viral mutation '151M' causes resistance to the drug named 'zidovudine'.

Furthermore, the set of abducibles A contains only a single abducible, namely the predicate *mutation*. The set of integrity constraints IC contains a single integrity constraint to model the fact that if a patient P carries a mutation M at time $T1$, then he or she must carry that mutation at all later times $T2$:

$$\mathit{false} \leftarrow \mathit{mutation}\ (P, T1, M),\ T2 \geq T1,\ \mathit{not\ mutation}\ (P, T2, M)$$

Given a patient's clinical history, the system abductively infers the mutations the patient may be carrying. In a second step, it exploits genotypic drug resistance interpretation rules to deduce for which drugs the patient may be resistant.

A remarkable characteristic of the work presented by Kakas et al. is its assumption that it is not always possible to select the best explanation among various possible explanations, since the application domain is intrinsically uncertain, and thus, one can never be absolutely sure which mutations a patient is carrying. Different from the approaches of Charniak and Goldman [CG91] and Hobbs et al. [HSME88], where abduction is applied to infer the best explanation, Kakas et al.ś approach deals with multiple alternative explanations. In order to predict for which drugs a patient may be resistant, the iS3 system ranks each mutation according to the number and size of explanations it appears in, and each drug according to the number of explanations that imply its resistance. To avoid the explosion of possible explanations for a drug resistance, iS3 computes solely minimal abductive explanations for abductive inference. A minimal explanation is an explanation from which no smaller explanations can be obtained by removing atoms [KKT98].

The work of Kakas et al. shows us that it is not always possible to consider abduction as inference to the best explanation. In application domains with intrinsic uncertainty, an abductive framework has to deal with multiple alternative explanations. In that case, it is essential to apply appropriate criteria for reducing the number of possible explanations.

Shanahan: Other relevant work on abduction is due to Shanahan [Sha05], who presents a formal theory of robot perception as a form of abduction. The work extends the basic abductive framework presented in [Sha96], where robot perception is given a logical characterization by defining it as an abductive task. For a given stream of low-level sensor data Γ, the task of abduction is to find one or more explanations (Δs) that are consistent with the background theory Σ such that the stream of low-level sensor data Γ logically follows from the union of the background theory Σ and an explanation Δ:

$$\Sigma \cup \Delta \models \Gamma$$

The stream of low-level sensor data Γ consists of a set of observation sentences. The abductive framework tries to explain sensor data by hypothesizing the existence, locations, and shapes of objects. Therefore an explanation Δ contains a consistent logical description of the locations and shapes of hypothesized objects in the scene (or environment). The background theory Σ describes how robot's interactions with the world impact on its sensors. Typically there are many Δs that explain the given sensor data, thus a preference relation is required to order them.

[Sha05] extends the basic abductive framework presented in [Sha96] in several important practical aspects of robot perception such as *top-down information flow*, *active perception* and *sensor fusion.* The incorporation of *top-down information flow* is achieved through the use of *expectations.* An explanation, together with the background theory Σ, entails not only the observations in Γ but also a number of other observation sentences, called expectations, that might not exist in the original sensor data Γ. Each explanation can be confirmed or revoked by checking whether its expectations are fulfilled. To this end low-level sensor data can be reanalyzed, e.g. using a different edge detection algorithm, or additional sensor data can be gathered, probably in a different modality. As a result, the flow of high-level information influences low-level image processing.

Shanahan further extends the basic abductive framework to encompass *active perception.* Motivated by human perception, which is active (e.g. human look by adjusting their eyes or turning their heads to an object), active vision and perception have been studied in computer vision community. In active vision, an observer is considered active if engaged in activities to control the geometric parameters of some sensory apparatus [AWB87]. In [Bal91], a camera control system mounted on a movable robotic arm has been used to study active perception. Shanahan widens the concept of active perception to include any form of action that leads to the acquisition of new information via a robot's sensors. In Shanahan's view, active perception includes not only low-level actions such as adjusting the threshold of an edge detection routine or rotating a robot's camera, but also high-level actions such as moving a robot to explore its environment.

Another extension of the basic abductive framework aims to address the problem of *sensor data fusion* where potentially conflicting data from multiple sources has to be fused into a single model of some aspect of the world [CY90]. In the robotics context, the problem of sensor fusion is relevant even for very simple robots with a single sensor since the same sensor data can be processed using different algorithms or can be extracted at different times. Therefore, Shanahan argues for a careful design of the background theory Σ with respect to conflicting data. To this end, Σ should include noise and/or abnormality conditions in terms of formulas that can explain away unexpected sensor data as noise and/or dismiss the absence of expected sensor data as abnormal. The preference relation over Δs should be defined in such a way that the explanation with the fewest noise or abnormality terms is preferred.

Shanahan considers the definition of a preference relation over possible explanations an important measure against the explosion in the number of possible explanations for observations. In [Sha05], the preference relation is required to respect both the explanatory value and expectations of an explanation. In a nutshell, the idea is to prefer explanations that explain more sensor data against competing explanations by assigning a higher explanatory value to them. In addition, other things being equal, an explanation with fewer unfulfilled expectations should be preferred.

Assume that the set of all possible explanations $\Delta_1, \ldots, \Delta_n$ is computed to explain a stream of sensor data Γ. Under the assumption that the set of explanations is mutually exclusive, i.e. one and only one of the explanations is true, the explanatory value of an explanation Δ_k is equal to the posterior probability of Δ_k. Let R be the set of all possible explanations apart from Δ_k: $R = \{\Delta_1, \ldots, \Delta_{k-1}, \Delta_{k+1}, \ldots, \Delta_n\}$. Following Bayes' rule, the posterior probability of Δ_k is:

$$\mathbf{P}(\Delta_k \mid \Delta_k \oplus R) = \frac{\mathbf{P}(\Delta_k \oplus R \mid \Delta_k)\,\mathbf{P}(\Delta_k)}{\mathbf{P}(\Delta_k \oplus R)}$$

In this equation $\mathbf{P}(\Delta_k)$ stands for the prior probability of Δ_k. The $\oplus$ (exclusive or) operator represents the assumption that one and only one of the terms on the right-hand side of the $\mid$ sign is true. Notice that $\mathbf{P}(\Delta_k \oplus R \mid \Delta_k)$ can be removed from the equation, because $\mathbf{P}(\Delta_k \oplus R \mid \Delta_k) = 1$, i.e. $\mathbf{P}(\Delta_k \oplus R)$ is always true if $\mathbf{P}(\Delta_k)$ is true. Furthermore, $\mathbf{P}(\Delta_k \oplus R)$ can be replaced with $\mathbf{P}(\Delta_k) + \mathbf{P}(R)$, since according to the probability law $\mathbf{P}(\Delta_k \vee R) = \mathbf{P}(\Delta_k) + \mathbf{P}(R)$ - $\mathbf{P}(\Delta_k \wedge R)$, and $\mathbf{P}(\Delta_k \wedge R) = \{\}$. Consequently, the equation simplifies to:

$$\mathbf{P}(\Delta_k \mid \Delta_k \oplus R) = \frac{\mathbf{P}(\Delta_k)}{\mathbf{P}(\Delta_k) + \mathbf{P}(R)} \tag{3.2}$$

Equation 3.2 defines the probability of $\mathbf{P}(\Delta_k)$ to be true, given that either Δ_k or R is true. The prior probability of R can be calculated as follows (see the definition of R above):

$$\mathbf{P}(R) = \left[\sum_{i=1}^{n} \mathbf{P}(\Delta_i)\right] - \mathbf{P}(\Delta_k) \tag{3.3}$$

Notice that R is the set of all possible explanations apart from Δ_k, so we need to subtract $\mathbf{P}(\Delta_k)$ from the sum of the probabilities of all explanations.

The prior probability of each hypothesis (aka explanation) Δ_i consisting of the atoms $\Psi_1, \ldots \Psi_m$ is calculated using the following equation:

$$\mathbf{P}(\Delta_i) = \prod_{j=1}^{m} \mathbf{P}(\Psi_j) \tag{3.4}$$

Equation 3.4 is defined under the very strict assumption that the probabilities of each atom Ψ_j constituting an explanation Δ_i are independent from each other.

Using Equations 3.2, 3.3 and 3.4 the posterior probability of each explanation, and thus the explanatory value of each explanation can be calculated. Furthermore, Equation 3.4 also enables the incorporation of expectations in the calculation of the explanatory value. To this end, noise terms can be assigned with low probabilities and then, according to Equation 3.4, an explanation including a term with a low probability gets a lower explanatory value. Therefore, Equation 3.4 enables the realization of the idea to punish explanations that explain away some sensor data as noise or have unfulfilled expectations. Note that there may be situations such as the malfunctioning of a sensor where the data delivered by that sensor is erroneous, and thus explanations that explain away the erroneous sensor data as noise should not be punished but rewarded. In Shanahan's framework this can be achieved by assigning high probability values to noise terms, since we expect the observations made by the malfunctioning sensor to be wrong.

In [Sha05], Shanahan presents not only a formal theory of robot perception as abduction but also discusses the application of the theoretical framework in various robotics contexts. To this end, he describes a number of experiments that took place in various environments. He evaluates the abductive account of perception using real robots that have different capabilities with respect to movement and the number of sensors [Sha00]. To realize his abductive framework in practice, e.g. in the experiments reported in [Sha00] and [Sha05], Shanahan exploits the *event calculus*. The event calculus was introduced initially by Kowalski and Sergot [KS86] as a logic programming formalism for representing events and their effects. Shanahan uses a dialect of the event calculus introduced by Kowalski and Sergot as the formalism for representing and reasoning about robot actions and their effects [Sha99]. The ontology of the event calculus in [Sha05] contains events (or actions), time points and so-called *fluents*. Fluents are any kind of parameters, e.g. the location of a robot, whose values change over time.

To sum up, Shanahan considers robot perception as an abductive process for processing a set of low-level sensor data. He presents an abductive framework that implements the abductive process which consists of four steps: i) Establish a set of hypotheses, i.e. compute a set of explanations using abduction. ii) Compute the explanatory value of each hypothesis, and select the best hypothesis with respect to the explanatory values. iii) Determine the expectations of each hypothesis selected in step ii, and reanalyze the sensor data to confirm or deny these expectations. iv) Recompute the explanatory value of the selected hypothesis with respect to the reanalysis done in step iii, and reorder the hypotheses.

It should be emphasized here that in Shanahan's approach the computation of the explanatory value of an explanation is based on the assumption that the probabilities of atoms Ψ_j constituting an explanation Δ_i are independent from each other (see Equation 3.4). Even though Shanahan reports his approach to deliver satisfactory experimental results in the robotics context, this is a very strong simplification, that, in general, cannot hold in other contexts.

Elsenbroich et al.: The work by Elsenbroich et al. [EKS06] presents an interesting investigation of various forms of abductive inference tasks in expressive DLs. Among others, they identify TBox and ABox abduction as important abductive reasoning tasks in DLs. They consider *TBox abduction* as an abductive reasoning task that can be used to repair unwanted non-subsumptions. I.e. in cases where an ontology engineer wants to have a subsumption such as $C \sqsubseteq D$, which does not follow from the TBox T, TBox abduction provides a finite set of TBox assertions S as a solution such that:

$T \cup S \models C \sqsubseteq D$

In [EKS06], the authors also consider *ABox abduction* in DLs as a new query answering service, that allows for abductive retrieval of instances of concepts or roles that would entail a desired ABox assertion. Given the TBox T and an ABox assertion γ that is desired to be entailed, a solution to the ABox abduction problem is any finite set of ABox assertions S such that

$T \cup S \models \gamma$

In addition to the formalization of abductive reasoning tasks in DLs, Elsenbroich et al. also address the problem of selecting 'good' solutions. Abductive problems, in general, can have infinitely many solutions and therefore it is important to identify criteria for preferring one solution over another. Therefore, the authors discuss different preference criteria such as syntactic notions to prefer solutions of a specific form of minimal length or semantic notions such as maximality w.r.t. subsumption. Finally, the authors address the problem of finding solutions by citing relevant work from literature. They classify these works into three groups according to the technique used for finding solutions: Those exploiting abductive logic programming [KD93], those using resolution for finding abductive explanations [Pau93], and those aiming to develop tableaux calculi for realizing abductive inference in propositional logic [AL97] or first-order logic [MP93]. Elsenbroich et al. favor the integration of abductive reasoning techniques into existing tableau algorithms for DLs but they do not provide for a concrete solution in [EKS06].

Putting it together, abduction is the form of reasoning from observations to explanations (or hypothesis) and has been widely used to formalize the computation of explanations. Besides theoretical studies there exists also several practical works in the literature, which have implemented abductive frameworks for solving real-world problems using different formalisms such as first-order logic or Horn-clause logic. The majority of these works share the view that in many application domains it is not always possible to compute a single explanation as the best explanation among multiple alternative explanations. In many cases abduction has to compute several possible explanations, and a measure to rank these explanations is necessary. Given such a measure, a preference relation over possible explanations can be defined. This enables abductive frameworks to cope with the possible explosion in the number of explanations by focusing on more likely explanations.

Except Elsenbroich et al.ś work, which is an investigation of abduction in DLs, all other works presented in this section not only discuss an abductive framework that exploits a formal language, but also present a software system. These software systems can be considered as *proof of concept*, i.e. as prototypes used to demonstrate that the ideas are workable, and can be realized as software systems. Therefore, in our view, the next research challenge is the development of a generic abduction framework, which can be applied to various application domains, and its implementation as a stable software system. To this end, we believe that existing DL reasoners which already demonstrated some scalability and stability should be reused, and enhanced if necessary. The proposal by Elsenbroich et al. to consider ABox abduction as an extended query answering service is a promising idea, since abduction can be performed in a more goal directed way, i.e. only the abducibles required to answer a certain query have to be hypothesized. Furthermore, existing DL reasoners incorporate powerful query engines, which offer at least so-called grounded conjunctive queries, that can be exploited. It is obvious that the idea proposed by Elsenbroich et al. can be extended to the case of conjunctive queries, but it is not clear yet how the more general rule-based abduction can be realized within the existing DL reasoners. These research issues are addressed in the next sections of this thesis.

Considering all these facts, we think that the incorporation of existing DL reasoners, and the enhancement of their capabilities is a promising way to build a stable software system for multimedia interpretation. Finally, evaluation strategies are required to rigorously explore the performance and scalability of the software system.

3.3.2 The ABox Abduction Algorithm

As the discussion of relevant work has shown, abductive reasoning is the appropriate form of reasoning for interpretation. In our multimedia interpretation approach, existing DL reasoning mechanisms and rules are combined in a coherent framework. In this framework, abduction constitutes the basis for the interpretation process that aims to compute deep-level annotations for a multimedia document. As mentioned before, the generation of explanations as part of a multimedia interpretation process requires hypothesization. To this end, rules are used to define the space of abducibles.

Notice that as discussed in Section 3.2, the multimedia interpretation process should not stop after explaining the surface-level objects and their relations. In fact, the computation of explanations for surface-level objects and their relations is only the first step of an iterative multimedia interpretation process. In subsequent iterations the explanations computed by the previous iteration should be explained as well. This is plausible since requiring explanations for explanations forces the multimedia interpretation process to compute rich relational structures, and thus deep-level annotations are obtained.

Before the presentation of the interpretation process, we proceed with the ABox abduction algorithm which is used by the interpretation process. As discussed in Section 3.3, abduction is formalized as

$$\Sigma \cup \Delta \models \gamma$$

where the background knowledge Σ, and an observation γ are given, and explanations (Δs) are to be computed. If DLs are used as the underlying formalism, Σ is a knowledge base that consists of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$, γ is an ABox assertion, and Δ is an ABox. In our approach, we extend the background knowledge Σ to contain also a non-recursive rule set $\mathcal{R}$: $\Sigma = (\mathcal{T}, \mathcal{A}, \mathcal{R})$. As discussed in Section 3.1.2, all atoms that appear in rules from $\mathcal{R}$ are required to be DL atoms. In other words, every atom in rules from $\mathcal{R}$ is either a concept or a role name in $\mathcal{T}$.

More precisely, γ is an ABox assertion of the form $P(\underline{I})$, where P is a concept or role name from $\mathcal{T}$, and $\underline{I}$ is a sequence of individuals from $\mathcal{A}$. Consequently, the abduction formula becomes

$$\Sigma \cup \Delta \models_{\mathcal{R}} P(\underline{I})$$

In our framework, abduction is implemented as a retrieval inference service in DLs, and thus requires a Boolean query as input. Therefore, we introduce the function *transform*, which given an ABox assertion $P(\underline{I})$ returns a Boolean query $\{() \mid P(\underline{I})\}$. Unfortunately existing DL reasoners do not support Boolean query answering w.r.t. a set of rules in Σ, i.e. only $\mathcal{T} \cup \mathcal{A}$ is considered. However, given that the rules are non-recursive, it is possible to *expand* the Boolean query $\{() \mid P(\underline{I})\}$ w.r.t. the rules in $\mathcal{R}$, and thus effectively eliminate the rules from Σ. The result of applying the function *expand*, which will be presented in more detail later, to a Boolean query is, in general, a set of Boolean conjunctive queries. Given a Boolean query, the goal of the abductive retrieval service is to acquire (or hypothesize) what has to be added to the ABox, i.e. explanations, in order to answer the query positively.

To give a rough overview, the ABox abduction algorithm performs three major subtasks in the following order:

1. Expand the Boolean query w.r.t. a set of rules to obtain a set of Boolean conjunctive queries.

2. Instantiate query variables to generate explanations.

3. Filter out explanations that are not preferred.

In the remainder of this section, we present the first two subtasks of the ABox abduction algorithm, and then the complete ABox abduction algorithm. The third subtask will be discussed in the next section together with an overview of existing literature on selecting preferred explanations.

Query Expansion: The goal of the query expansion task is the semantics-preserving transformation of a Boolean query into a set of Boolean conjunctive queries w.r.t. a set of rules $\mathcal{R}$ such that the rules are eliminated from the background knowledge Σ. For query expansion we define the function *expand* as follows:

```
Algorithm expand($\{() \mid P(\underline{I})\}, \mathcal{R}$)
Output: a set of Boolean conjunctive queries
$S_0 := \{\}$
$S_1 := \{\{P(\underline{I})\}\}$
$i := 1$
while $S_i \neq S_{i-1}$ do
    if there exists $A_j \in S_i$ such that $expand'(A_j, \mathcal{R}) \neq \{\}$ then
        $S_{i+1} := (S_i \setminus A_j) \cup expand'(A_j, \mathcal{R})$
    else
        $S_{i+1} := S_i$
    end
    $i := i + 1$
end
return $\{\ \{() \mid \beta_1, \ldots, \beta_n\} \mid \{\beta_1, \ldots, \beta_n\} \in S_i\}$
```

Algorithm 3: The query expansion algorithm *expand*

where S_i denotes a set of sets of query atoms, A_j denotes a set of query atoms and β_i denotes a query atom. Algorithm 4 depicts the auxiliary function *expand*$'$:

```
Algorithm expand'($\{\alpha_1, \ldots, \alpha_n\}, \mathcal{R}$)
Output: a set of sets of query atoms
if there exists $\alpha_i$ and $(\alpha_i' \leftarrow \beta_1', \ldots, \beta_n') \in \mathcal{R}$ such that $\alpha_i = P(\underline{V})$ and
$\alpha_i' = P(\underline{W})$ of same arity then
    return $\{A' \mid A'' \in expand''(\alpha_i, \mathcal{R}),\ A' = (A \setminus \alpha_i) \cup A''\}$
else
    return $\{\}$
end
```

Algorithm 4: The algorithm for the auxiliary function *expand*$'$

α_i is a query atom of the form $\alpha_i = P(\underline{V})$ where P is a concept or role name, and hence $\underline{V}$ is a sequence of length one or two, respectively. The sequence $\underline{V}$ may contain individuals and variables as elements. The sequence $\underline{W}$ contains variables as elements, and has the same length as $\underline{V}$. Algorithm 5 shows the auxiliary function *expand*$''$:

```
Algorithm expand$''(\alpha, \mathcal{R})$
Output: a set of sets of query atoms
$B := \{\, \{\beta_1, \ldots, \beta_n\} \mid (\alpha' \leftarrow \beta'_1, \ldots, \beta'_n) \in \mathcal{R},$
                          $\Theta_1(\alpha, \alpha', \beta'_1) = \beta_1,$
                          $\vdots$
                          $\Theta_n(\alpha, \alpha', \beta'_n) = \beta_n\}$
return $B$
```

Algorithm 5: The algorithm for the auxiliary function *expand*$''$

Finally, we define the function Θ that rewrites atoms by renaming variables based on the atoms $\alpha = P(x_1, \ldots, x_n)$ and $\alpha' = P(y_1, \ldots, y_n)$. Note that y_i is a variable, and x_i is a variable or an individual. Based on α and α', an arbitrary variable z is renamed as follows:

$$\Theta_i(P(x_1, \ldots, x_n), P(y_1, \ldots, y_n), z) := \begin{cases} x_i & \text{if } z = y_i \\ z & \text{otherwise} \end{cases}$$

The function Θ is extended inductively for atoms as follows:

$$\Theta_i(\alpha, \alpha', P(z_1, \ldots, z_n)) := P(\Theta_i(\alpha, \alpha', z_1), \ldots, \Theta_i(\alpha, \alpha', z_n))$$

Variable Instantiation: Recall from Section 3.1.1 that answering a grounded conjunctive query w.r.t. a DL knowledge base $\mathcal{O}$ means finding admissible variable substitutions such that $\mathcal{O} \models \{Q_1(\sigma(\underline{V_1})), \ldots, Q_n(\sigma(\underline{V_n}))\}$ holds. To this end, the variable substitution function σ is applied to map variables from the query to individuals from an ABox $\mathcal{A}$.

In the context of abduction, to generate an explanation from a Boolean conjunctive query, the variables from the query have to be substituted with individuals as well. However, different from standard conjunctive query answering, abduction requires *hypothesization*, and hence query variables should be *instantiated* either with individuals from the ABox $\mathcal{A}$, or with individuals that do not exist in $\mathcal{A}$. To distinguish between existing and hypothesized individuals, we introduce the non-empty set $ExtInds_A$ which contains all individuals from $\mathcal{A}$, and the set $NewInds$ which contains a set of *hypothesized* individuals, also known as *new individuals*. The set $Vars_{BQ}$ contains all variables from a Boolean conjunctive query BQ. The sets $NewInds$ and $Vars_{BQ}$ have the same cardinality. Therefore we denote $NewInds$ as $NewInds_{BQ}$.

We define the variable instantiation function $\Phi_{BQ,\mathcal{A}}$ in a nondeterministic way as follows:

$$\Phi_{BQ,\mathcal{A}}(X,\Omega) := \begin{cases} i \in NewInds_{BQ} & \text{if } X \in Vars_{BQ}, \Omega = \text{use new individuals} \\ i \in NewInds_{BQ} \cup ExtInds_{A} & \text{if } X \in Vars_{BQ}, \Omega = \text{reuse existing individuals} \\ X & \text{if } X \in ExtInds_{A} \end{cases}$$

where X is either an individual from the set $ExtInds_A$ or a variable. If X is an individual, then applying $\Phi_{BQ,\mathcal{A}}$ results in X. If X is variable, the value of the parameter Ω defines the strategy in instantiating X. Depending on the value of the strategy parameter Ω, $\Phi_{BQ,\mathcal{A}}$ instantiates a variable with a new or an existing individual.

The variable substitution function $\Phi_{BQ,\mathcal{A}}$ is extended to sequences of variables and individuals in the obvious way, i.e. by element-wise application: If $\underline{V_1} = \langle X \rangle$ then $\Phi_{BQ,\mathcal{A}}(\underline{V_1}, \Omega)$ is defined as $\langle \Phi_{BQ,\mathcal{A}}(X,\Omega) \rangle$, and if $\underline{V_2} = \langle X, Y \rangle$ then $\Phi_{BQ,\mathcal{A}}(\underline{V_2}, \Omega)$ is defined as $\langle \Phi_{BQ,\mathcal{A}}(X,\Omega), \Phi_{BQ,\mathcal{A}}(Y,\Omega) \rangle$.

The complete ABox abduction algorithm: Having discussed the subtask *expand* for eliminating the rules in the abduction formula by expanding a Boolean query into a set of Boolean conjunctive queries, and the subtask *variable instantiation* for generating explanations based on the set of Boolean conjunctive queries, we proceed with the presentation of the complete ABox abduction algorithm.

```
Algorithm abduce($\mathcal{T}, \mathcal{A}, \mathcal{R}, \{() \mid P(\underline{I})\}$, $\Omega$)
Output: a set of explanations: $\Delta s = \{\Delta_1, ..., \Delta_n\}$
$\mathcal{BQ}s$ := expand($\{() \mid P(\underline{I})\}$, $\mathcal{R}$)
$\Gamma s := \{\}$
for each $BQ \in \mathcal{BQ}s$ where $BQ = \{() \mid Q_1(\underline{V_1}), \ldots, Q_n(\underline{V_n})\}$ do
    for each $\Phi_{BQ,\mathcal{A}}$ do
        $\Gamma = \{Q_1(\Phi_{BQ,\mathcal{A}}(\underline{V_1}, \Omega)), \ldots, Q_n(\Phi_{BQ,\mathcal{A}}(\underline{V_n}, \Omega))\}$
        if $\mathcal{T} \cup \mathcal{A} \cup \Gamma \not\models \bot$ then
            $\Gamma s := \Gamma s \cup \{\Gamma\}$
        end
    end
end
$\Delta s$ := select-preferred-explanations($\Gamma s$, $\mathcal{T}$, $\mathcal{A}$)
return $\Delta s$
```

Algorithm 6: The ABox abduction algorithm *abduce*

Informally speaking, the ABox abduction algorithm proceeds as follows:

1. Expand the query $\{() \mid P(\underline{I}\}\}$ w.r.t. $\mathcal{R}$ to obtain the set of Boolean conjunctive queries $\mathcal{BQ}$s (query expansion subtask).

2. Iterate as BQ over the Boolean conjunctive queries.

3. Iterate as $\Phi_{BQ,\mathcal{A}}$ over the set of possible variable instantiation functions w.r.t. BQ and $\mathcal{A}$. Note that this set if finite, and hence the termination of the loop is guaranteed.

4. Apply the current $\Phi_{BQ,\mathcal{A}}$ to BQ to instantiate its variables, yielding an ABox Γ whose non-entailed assertions constitute Δ_i. I.e., the explanation Δ_i contains only hypothesized assertions, whereas Γ may contain entailed assertions as well.

5. Check for each explanation Γ whether $\mathcal{A} \cup \Gamma$ is consistent with respect to $\mathcal{T}$. If Γ is consistent, then add it to the set of explanations Γs.

6. Remove non-preferred Γ's from Γs with respect to some criteria, determine their hypothesized assertions, and assign them to Δs.

7. Return the set of preferred explanations (Δs) for the Boolean query $\{() \mid P(\underline{I})\}$.

It should be noted here that Algorithm 6 provides for a specification of the abductive retrieval service. In a practical implementation, the algorithm can be optimized in various ways. For example, instead of selecting the preferred explanations from all explanations at the end of the algorithm, one can think of an incremental algorithm scoring and maintaining a *best so far explanation* and using it as a criterion for filtering out other explanations already at runtime.

Another possible optimization is to check for the consistency of an explanation before all variables of the corresponding Boolean conjunctive query are instantiated. In fact, partial explanations can be checked for consistency in order to detect and discard inconsistent explanations earlier, and thus, improve the performance of the abductive retrieval service.

3.3.3 Selecting Preferred Explanations

As discussed earlier, the general abduction formula in Equation 3.1 allows for an infinite number of explanations for any observation, if no further constraints are applied on explanations. Even though our abduction algorithm (shown in Algorithm 6) is guaranteed to deliver a finite number of explanations (Γs), it may still deliver too many explanations for a practical multimedia interpretation scenario. Therefore abductive reasoning is required to deliver 'preferred' explanations only (Δs).[1] In addition to the constraints on explanations already discussed in Section 3.3, namely consistency, minimality, relevance and explanatoriness, further criteria have been proposed in the literature.

In [Tha78], the cognitive psychologist Thagard investigates appropriate criteria for selecting a scientific theory as the best explanation for a phenomenon. He focuses on explanatory scientific theories such as Darwin's *theory of the evolution of species by natural selection* or Huygens' *wave theory of light*. According to Thagard, a scientific theory that best satisfies the criteria *consilience*, *simplicity* and *analogy*, is the best explanation. In a nutshell, a theory T_1 is more consilient than a theory T_2, if T_1 explains more classes of facts than T_2. In the scientific theories context, the notion of a *class of facts* has been used to organize facts for which competing scientific theories provide explanations. For example, phenomena such as reflection and polarization are two classes of facts that can be explained by the wave theory of light. A theory T_1 is simpler than a theory T_2, if T_2 assumes the existence of entities, also known as auxiliary assumptions, that T_1 does not. A theory T_1 has more explanatory value if there exists an analogy between T_1 and an established theory. For example, Darwin uses an analogy between his theory of the evolution of species by natural selection and the established theory of artificial selection to argue for the explanatory value of his theory. In artificial selection, also known as selective breeding, human preferences have a significant effect on the evolution of a particular species. Since natural selection is similar to artificial selection in a number of respects, Darwin concludes that natural selection also leads to the development of species.

Mayer and Pirri [MP96] discuss the logical nature of abduction and emphasize the preference of hypotheses as the key problem of abduction. They point out the inappropriateness of Thagard's consilience criteria for a logic-based approach, since an inconsistent theory can explain anything, and hence is maximally consilient. The authors consider preference not only as a logical issue that is context-independent

[1] Although Γ and Δ are both called explanations, a Δ is a 'preferred' explanation, and for convenience reasons Δ does not contain entailed assertions.

[MP96] and hence identify two main categories of criteria: *context-independent* and *context-dependent* criteria. In addition to context-independent criteria, the authors advocate the use of context-dependent criteria, since each domain has its own laws of preference.

Regarding context-independent criteria, consistency and minimality are the most-commonly used criteria on explanations in the literature, as discussed earlier in this section. Our ABox abduction algorithm delivers only consistent explanations, since it discards inconsistent explanations (see Algorithm 6). As discussed earlier, the minimality restriction is often used to reduce the number of explanations. An infinite number of (more-specific) explanations exists otherwise. Since the presented ABox abduction algorithm always computes a finite number of explanations, the minimality restriction is not required to reduce the number of explanations. In our approach, explanations are constructed with respect to the finite space of abducibles, which is defined explicitly in the form of a non-recursive rule set. It is thus guaranteed that the number of explanations is finite, and the ABox abduction algorithm computes all possible explanations with respect to the space of abducibles.

Regarding context-dependent criteria, we define a preference relation for multimedia interpretation inspired by the works of Thagard, Mayer and Pirri, and Hobbs et al. as follows. Assume that an ABox $\mathcal{A}$ with the results of the analysis process for a multimedia document is given. Furthermore, assume that for an assertion from $\mathcal{A}$ the set of explanations Γs has been computed according to Algorithm 6. This means that we have a set of consistent explanations Γs and would like to acquire the set of preferred explanations Δs.

We prefer explanations that involve as many ground assertions from the analysis ABox $\mathcal{A}$ (consilience) as possible, and at the same time hypothesize as few assertions as possible (simplicity). More precisely, we define the consilience value S_c and the simplicity value S_h as follows:

$$S_c(\mathcal{T}, \mathcal{A}, \Gamma) := \sharp\{\alpha \in \mathcal{A} \mid \mathcal{T} \cup \Gamma \models \alpha\}$$
$$S_h(\mathcal{T}, \mathcal{A}, \Gamma) := \sharp\{\gamma \in \Gamma \mid \mathcal{T} \cup \mathcal{A} \not\models \gamma\}$$

where $\mathcal{T}$ is a TBox and Γ an explanation. In order to have a measure of preference for explanations we define the preference score S as follows:

$$S(\mathcal{T}, \mathcal{A}, \Gamma) = S_c(\mathcal{T}, \mathcal{A}, \Gamma) - S_h(\mathcal{T}, \mathcal{A}, \Gamma) \tag{3.5}$$

The preference score reflects the assumption that, in the context of multimedia interpretation, an explanation is preferable over others if it is more consilient and simpler. Therefore, an explanation with the highest preference score S is preferred over other explanations.

According to Equation 3.5, an explanation with high consilience value is allowed to hypothesize more assertions than explanations with lower consilience values. For example, assume that two explanations, Γ_1 and Γ_2, with the following consilience and simplicity values are given: $S_{c,1}$=4, $S_{h,1}$=2 and $S_{c,2}$=2, $S_{h,2}$=1. Even though Γ_1 hypothesizes more assertions than Γ_2, Γ_1 is the preferred explanation, because according to Equation 3.5, the preference score of Γ_1 is higher than the preference score of Γ_2.

In DLs, checking an ABox assertion for entailment is a computationally expensive task. In multimedia interpretation, analysis ABoxes often contain a large number of assertions, and thus, the computation of the preference score according to Equation 3.5 requires many invocations of the entailment inference service. To overcome this problem, the consilience value S_c can be approximated such that fewer entailments checks are required. Since all entailed assertions in Γ contribute to the consilience of Γ, we define the approximated consilience value S_a as follows:

$$S_a(\mathcal{T}, \mathcal{A}, \Gamma) := \sharp\{\gamma \in \Gamma \mid \mathcal{T} \cup \mathcal{A} \models \gamma\}$$

The approximated consilience value S_a can be computed much more efficiently than the initial consilience value S_c, since only the assertions in Γ have to be checked for entailment. We slightly modify the definition of the preference score S to incorporate S_a instead of S_c:

$$S(\mathcal{T}, \mathcal{A}, \Gamma) = S_a(\mathcal{T}, \mathcal{A}, \Gamma) - S_h(\mathcal{T}, \mathcal{A}, \Gamma) \tag{3.6}$$

The use of the approximated consilience S_a in the preference score S ensures on the one hand efficient computability, and on the other hand reflects (at least partially) the original consilience value S_c. With respect to performance considerations, we decided to use the preference score S defined in Equation 3.6, and obtained very satisfactory experimental results in terms of quality and runtime performance. In Chapter 5, we present these results in detail and evaluate them.

In addition to the preference score S defined in Equation 3.6, we apply a further context-dependent criterion. In accordance with Hobbs et al. [HSAM90], among two explanations with the same preference score, we prefer the explanation with the most-specific assumptions, since it is most informative in the context of multimedia interpretation. For example, the image of an athlete vaulting with a pole can be explained as showing a sports trial or a pole vault trial, where we prefer the latter one. Notice that the preference of more specific explanations is in accordance with the consilience criteria, because more specific explanations potentially entail more assertions from $\mathcal{A}$, and thus have higher consilience values.

As a result of the variable substitution function Θ used in Algorithm 5, the algorithm may deliver explanations which are logically equivalent modulo renaming of new individuals. Consequently, the standard ABox entailment inference service is not sufficient to compare explanations for specificity. Therefore, we define the so-called *relaxed ABox entailment relation*, denoted by the $\rightsquigarrow$ symbol, in order to compare two explanations (or ABoxes in general) for specificity. Informally speaking, the new individuals in the ABox to be checked for entailment, i.e. the new individuals on the right hand side of the of the entailment sign, have to be considered as variables and not as ABox individuals, whereas the new individuals on the left hand side of the entailment sign are considered as ABox individuals.

Let us assume that the explanations Γ_1 and Γ_2 (w.r.t. the TBox $\mathcal{T}$) are to be compared under the UNA. The relationship $\Gamma_1 \rightsquigarrow \Gamma_2$ holds if there exists a total and injective mapping ψ: *new-inds-from*$(\Gamma_2) \rightarrow$ *inds-from*(Γ_1) such that $\Gamma_1 \models \psi(\Gamma_2)$ w.r.t. $\mathcal{T}$. Note that this definition is only reasonable if the UNA holds for both explanations. Given an ABox the function *new-inds-from* returns all new individuals from the ABox, where as the function *inds-from* returns all individuals (including new individuals).

The entailment problem $\Gamma_1 \models \psi(\Gamma_2)$ w.r.t. $\mathcal{T}$ can be reduced to Boolean conjunctive query answering as follows: Assume that a knowledge base consisting of $\mathcal{T}$ and Γ_1 is given: $\mathcal{O} = (\mathcal{T}, \Gamma_1)$. First, each new individual $i \in$ *new-inds-from*(Γ_2) is replaced with a distinct query variable. Subsequently, the function *transform* is applied to obtain a Boolean conjunctive query, which is then answered with respect to the knowledge base $\mathcal{O}$. If the Boolean conjunctive query is answered with true, then $\Gamma_1 \models \psi(\Gamma_2)$ w.r.t. $\mathcal{T}$

holds. Consequently, the relationship $\Gamma_1 \rightsquigarrow \Gamma_2$ holds. In this case, we say that Γ_1 is more specific than Γ_2, and prefer Γ_1 over Γ_2.

To recapitulate, it is not sufficient to obtain a finite set of explanations of an observation for practical applications such as multimedia interpretation. We have to further reduce the number of explanations by selecting the preferred explanations among all possible explanations. The ABox abduction algorithm in Algorithm 6 incorporates the *select-preferred-explanations* function in order to deliver only such explanations that are preferred with respect to the two context-dependent criteria presented in this section. We define the function for selecting preferred explanations as follows:

```
**Algorithm select-preferred-explanations**($\Gamma s$, $\mathcal{T}, \mathcal{A}$)
**Output**: a set of preferred explanations: $\Delta s = \{\Delta_1, ..., \Delta_n\}$
$\Gamma_{max} := \{\Gamma \mid \Gamma \in \Gamma s\ ,\ \nexists\ \Gamma' \in \Gamma s\text{: } S(\mathcal{T}, \mathcal{A}, \Gamma') > S(\mathcal{T}, \mathcal{A}, \Gamma)\}$
$\Gamma_{most-spec} := \{\Gamma \mid \Gamma \in \Gamma_{max},\ \nexists\ \Gamma' \in \Gamma_{max} : \Gamma' \rightsquigarrow \Gamma\}$
$\Delta s := \{\Delta \mid \Gamma \in \Gamma_{most-spec}, \Delta = \{\gamma \in \Gamma \mid \mathcal{T} \cup \mathcal{A} \not\models \gamma\}\}$
**return** $\Delta s$
```

Algorithm 7: The algorithm for selecting preferred explanations

Given the set of all possible explanations Γs, the *select-preferred-explanations* function filters out non-preferred explanations as follows: First, the preference score of each explanation in Γs is computed according to the formula in Equation 3.6, and the explanations with the highest preference score constitute the set Γ_{max}. Second, the most-specific explanations from Γ_{max} are determined using the relaxed ABox entailment inference service to obtain the set $\Gamma_{most-spec}$. Finally, all non-entailed assertions from $\Gamma_{most-spec}$ are used to constitute the set Δs, which is then returned as the result of the *select-preferred-explanations* function call.

Notice that it is not always possible to determine the more specific explanation between two explanations. Therefore, the result of the *select-preferred-explanations* function call may contain multiple explanations, and hence the abductive retrieval service may deliver more than one explanation for an observation. Our framework is designed to deal with multiple alternative explanations when computing interpretations of a multimedia document.

3.4 Abduction-Based Interpretation

After the discussion of ABox abduction as a non-standard DL retrieval inference service and the presentation of an algorithm for its implementation, we proceed with the presentation of an algorithm for multimedia interpretation, in which abduction plays a key role.

Let us assume that background knowledge about the application domain is available in the form of a TBox and a non-recursive rule set. In a nutshell, given the background knowledge and an analysis ABox representing the analysis results of a multimedia document, the goal of the interpretation process is to compute interpretations of the document, and to deliver interpretation ABoxes as result. To this end, the interpretation algorithm exploits abduction to compute explanations for assertions.

Depending on the application context, it is not always required to compute explanations for every assertion of an analysis ABox. For example, some assertions may represent analysis results that are considered as correct by default. We call such assertions *bona-fide assertions*. Other assertions of the analysis ABox require fiats, i.e. their correctness is questioned by the agent and should be explained. We call these assertions *fiat assertions*.

As discussed earlier in this chapter, it is not sufficient to only explain fiat assertions from the analysis ABox, i.e. surface-level objects and their relations, to compute an interpretation of a document. Since interpretations are required to consist of rich relational structures including abstract aggregates, after the computation of explanations for a fiat assertion, the interpretation process should identify new fiat assertions, for which explanations should be computed as well.

At first sight, it seems to be a reasonable idea to use all assertions returned by the *abduce* function as fiats for the next level. However, the rules used in this framework have some limitations (e.g. the head of each rule cannot contain more than one DL atom) and hence are not specific but rather general. Since it is not possible to define very context-specific rules, a large number of rules has to be defined, and the space of abducibles is, in general, very large. Consequently, using all assertions returned by the *abduce* function as fiats for the next level requires a lot of computation and reduces the performance of the interpretation process. This problem can be solved by better controlling the interpretation process and identifying the fiats for the next level of the process. To this end, we introduce forward-chaining rules as a mechanism for the definition of context-specific rules, and the context-specific identification of fiats for the next level. Unlike the rules used for abduction which are applied with respect to the single atom in the rule head, forward-chaining rules require a conjunction of atoms to be true in order to be applied. Therefore forward-chaining rules enable the definition of more context-specific interpretation means, and at the same time enable better control of the interpretation process through context-specific identification of fiats. As we will see later, the interpretation algorithm exploits the *apply-rules* function (introduced in Section 3.1.2) to infer context-specific information and to identify fiat assertions for the next level of the interpretation process.

Our practical experiments have shown that context-specific identification of the fiat assertions for the next level of the interpretation process is crucial for the run-time performance of a multimedia interpretation engine. In addition, the quality of the interpretation results depend on the order in which fiat assertions are explained, since the order, in general, effects the interpretations of a multimedia document and most application scenarios require the computation of certain preferred interpretations. Therefore application-specific strategies are required for selecting the next fiat assertion from a set of fiat assertions. Since the order in which fiat assertions are explained is important, a simple set of fiat assertions is not sufficient. To represent the order of the fiats, we define the set of fiat assertions *Fiats* that consists of pairs of the form (γ, *level*), where γ is a fiat assertion and *level* is a natural number. Informally speaking, a level i means that the assertion was introduced in the i-th recursive call of the interpretation algorithm. Since in the beginning of the interpretation process *Fiats* contains fiat assertions from the analysis ABox, these assertions have the level zero.

The generic interpretation algorithm *interpret* relies on the function *select-fiat*, which implements the interpretation strategy. The function *select-fiat* selects the next fiat from the set *Fiats* (as specified above) according to some strategy (e.g. depth-first or breadth-first) w.r.t. the level of the assertions. Depending on the requirements of an application context, an appropriate *select-fiat* function has to be defined.

Having investigated a large repository with multimedia documents from the athletics domain, our experiments have shown that the following *select-fiat* function is the most appropriate one in this domain, and ensures that the interpretations computed by the interpretation algorithm correspond to the interpretations preferred by humans:

```
Algorithm select-fiat($Fiats$)
Output: a triple: ($\gamma$, $level$, $\Omega$)
$min\text{-}level := min$ ({ $level$ | ($\gamma$, $level$) $\in$ $Fiats$})
$\mathcal{C} :=$ { $\gamma$ | ($\gamma$, $min\text{-}level$) $\in$ $Fiats$, $\gamma$ is a concept assertion }
$\mathcal{R} :=$ { $\gamma$ | ($\gamma$, $min\text{-}level$) $\in$ $Fiats$, $\gamma$ is a role assertion }
if $\mathcal{C} \neq \{\}$ then
    Let $\gamma \in \mathcal{C}$
    return ($\gamma$, $min\text{-}level$, use new individuals)
else
    Let $\gamma \in \mathcal{R}$
    return ($\gamma$, $min\text{-}level$, reuse existing individuals)
end
```

Algorithm 8: The algorithm for selecting the next fiat assertion, its level and the appropriate abduction strategy

It should be noted that the *select-fiat* function is only called with a non-empty set *Fiats*. Besides a fiat assertion γ and the *level*, the triple returned by the *select-fiat* function contains also Ω, the strategy parameter value to control the *abduce* function (see Algorithm 6).

Informally speaking, the *select-fiat* function proceeds as follows: After the determination of the lowest level of fiat assertions, it returns a fiat concept assertion if there is any, and a fiat role assertion otherwise. For fiat concept assertions the abduction strategy parameter gets the value *use new individuals* and for fiat role assertions the value *reuse existing individuals*. Additional motivation for the selection of the appropriate abduction strategy parameter will be given in Section 4.3.2 using practical examples from the athletics domain.

Algorithm 9 shows the multimedia interpretation algorithm *interpret*, which is the centerpiece of our multimedia interpretation approach. In order to get interpretations for a multimedia document, the recursive *interpret* algorithm is initially called with a TBox $\mathcal{T}$, a non-recursive rule set $\mathcal{R}$, an analysis ABox $\mathcal{A}$, and the set *Fiats* as input.

The set *Fiats* contains initially all fiat assertions from the analysis ABox $\mathcal{A}$. We assume that the set of fiat assertions is determined before the invocation of *interpret*. In a practical application, any assertion from an analysis ABox can be considered as a fiat assertion, if there exists a rule $r_i \in \mathcal{R}$ where the predicate of the rule atom in the head of r_i is the same as the concept or role name of the assertion.

```
Algorithm interpret($\mathcal{T}, \mathcal{R}, \mathcal{A}, Fiats$)
Output: a set of interpretation ABoxes: $\mathcal{I} = \{\mathcal{A}_1, ..., \mathcal{A}_n\}$
if $Fiats = \{\}$ then
    return $\{\mathcal{A}\}$
else
    $(\gamma, level, \Omega) :=$ select-fiat($Fiats$)
    $Fiats' := Fiats \setminus \{(\gamma, level)\}$
    $\Delta s :=$ abduce($\mathcal{T}, \mathcal{A} \setminus \gamma, \mathcal{R}, \{() \mid \gamma\}, \Omega$)
    if $\Delta s = \{\}$ then
        return interpret($\mathcal{T}, \mathcal{R}, \mathcal{A}, Fiats'$)
    else
        $\mathcal{I} := \{\}$
        for each $\Delta \in \Delta s$ do
            $\mathcal{A}' := \mathcal{A} \cup \Delta$
            $\mathcal{C} :=$ apply-rules($\mathcal{T}, \mathcal{A}', \mathcal{R}$)
            if $\mathcal{T} \cup \mathcal{A}' \cup \mathcal{C} \not\models \bot$ then
                $\mathcal{I} := \mathcal{I} \cup$ interpret($\mathcal{T}, \mathcal{R}, \mathcal{A}' \cup \mathcal{C}, Fiats' \cup \{(c, level + 1) \mid c \in \mathcal{C}\}$)
            end
        end
        return $\mathcal{I}$
    end
end
```

Algorithm 9: The interpretation algorithm *interpret*

Informally speaking, the multimedia interpretation algorithm proceeds as follows:

1. If the set *Fiats* is empty, return a (singleton) set containing the ABox $\mathcal{A}$.
2. Otherwise determine the next fiat assertion γ, its level and the appropriate abduction strategy Ω using the *select-fiat* function, and remove the pair (γ, *level*) from *Fiats*.
3. For γ compute explanations (Δs) using the *abduce* function.
4. If no explanations could be computed, then return the result of the recursive *interpret* function call for the remaining set of fiat assertions.
5. Otherwise for each explanation Δ, add Δ to the ABox and apply the rules $\mathcal{R}$ in a forward-chaining way to obtain the set of consequences $\mathcal{C}$.

6. If the ABox augmented with the consequences $\mathcal{C}$ is consistent w.r.t. $\mathcal{T}$, then add each assertion from $\mathcal{C}$ as a fiat of level *level*+1 (where *level* is the level of γ) to the set of fiat assertions, and continue the interpretation with this augmented set of fiats. Note that for the recursive *interpret* function call the ABox augmented with the consequences $\mathcal{C}$ is used. Furthermore, the results of the recursive calls for each Δ are accumulated into the set $\mathcal{I}$, in which all interpretation ABoxes are collected.

7. After the accumulation of interpretations for all Δ, $\mathcal{I}$ is returned as result.

It should be noted here that the result of the *interpret* algorithm always contains at least one interpretation ABox. Let us assume the worst case in which for none of the fiat assertions in *Fiats* an explanation could be computed. Since the algorithm removes the fiat assertions successively, the set *Fiats* eventually becomes empty, and hence the algorithm terminates returning a singleton set containing the analysis ABox as the interpretation result.

3.5 Fusion of Modality-Specific Interpretations

The multimedia interpretation algorithm presented in the previous section facilitates the computation of interpretations for analysis results from a single modality. As discussed in Section 3.2 in detail, in our approach multimedia documents such as web pages are partitioned into segments prior to analysis. Modality-specific analysis and interpretation processes are then applied to each segment to obtain modality-specific interpretation ABoxes. Notice that a single segment may have multiple interpretations. The goal of the fusion process is to obtain one or many interpretation results for the whole multimedia document.

The first step in the fusion process is to merge (take the union of) the modality-specific interpretation results of all segments of a multimedia document. The information in different segments are, in general, redundant and complementary. Therefore the fusion process is required to identify individuals from different segments describing the same real-world entity. Notice that the individuals in different modality-specific interpretation results are distinct, and hence application-specific conditions have to be defined for the identification of such individuals in order to yield an information gain.

After the identification of individuals that describe the same real-world entity, certain possible interpretations of a multimedia document can be ruled out due to inconsistencies.

To give an example, assume that an image depicting an athlete during a sports trial, and the image caption containing the person name ‘Blanka Vlasic’ is given. Assume that there exists two interpretations for the image segment, one containing a pole vaulter and the other one a high jumper. Furthermore, the caption is interpreted as containing a person instance named ‘Blanka Vlasic’. Given the additional background information that ‘Blanka Vlasic’ is a high jumper, and pole vaulters and high jumpers are disjoint, any attempt to identify the person instance from the caption segment and the pole vaulter instance as the same will yield an inconsistency. As a result, one of the fusion possibilities is ruled out by combining the information from different modalities.

To sum up, the goal of the fusion process is not only to merge modality-specific interpretation ABoxes into a fused interpretation ABox. The fusion process is also required to identify individuals in different modality-specific interpretation ABoxes that describe the same entity in the real world, and make their sameness explicit through the addition of same-as assertions to the fused interpretation ABox(es). Notice that in our framework it is guaranteed that the sets of individuals contained in interpretation ABoxes from different modalities are disjoint. It should also be noted that for fusion we drop the UNA, which is employed for interpretation.

The conditions under which two individuals from different modality-specific interpretation ABoxes should be considered as describing the same real-world entity is application-dependent. To this end, we introduce the function *query-same-as-candidates* that poses queries to the union of the modality-specific interpretation ABoxes, and returns a set of pairs of individuals, where each pair of individuals is believed to be the same. Since these queries are application-dependent, we will present them in Section 4.3.3 for the athletics domain.

If one of the segments of a document has multiple interpretations, there exist also multiple alternatives to fuse the interpretation ABoxes, and multiple alternatives for fusion lead to the computation of several interpretation ABoxes for a multimedia document. However, in a practical application scenario we would like to reduce the number of fused interpretation ABoxes for a multimedia document. Getting back to our previous example, if we know the fact that the athlete ‘Blanka Vlasic’ is a high jumper, we would like to rule out interpretations, in which she is considered, e.g., a pole vaulter. To achieve this within our logical framework, we enhance our background knowledge $\mathcal{T}$ with additional axioms such as:

$$Person \sqcap \exists hasPersonName.\exists hasValue.\text{`}Blanka\ Vlasic' \sqsubseteq HighJumper$$

Henceforth we refer to the enhanced background knowledge as $\mathcal{T}_{enc}$. Since $\mathcal{T}_{enc}$ contains also axioms such as:

$$HighJumper \sqcap PoleVaulter \sqsubseteq \bot$$

the interpretations in which 'Blanka Vlasic' is considered a pole vaulter become inconsistent and are ruled out. In Section 4.3.3, using practical examples from the athletics domain, we will show how the axioms added to $\mathcal{T}$ cause certain fusion alternatives to become inconsistent, and thus enable the fusion algorithm to rule out certain fused interpretation ABoxes.

Algorithm 10 shows the fusion algorithm *fuse*. In order to get fused interpretation ABoxes for a multimedia document, the algorithm is called with a TBox $\mathcal{T}_{enc}$, a set of modality-specific interpretation ABoxes $\{\mathcal{M}_1, \ldots, \mathcal{M}_n\}$, and a set of conjunctive queries $\mathcal{CQ}s$ as input.

```
Algorithm fuse($\mathcal{T}_{enc}, \{\mathcal{M}_1, \ldots, \mathcal{M}_n\}, \mathcal{CQ}s$)
Output: a set of fused interpretation ABoxes: $\mathcal{F} = \{\mathcal{A}_{f1}, ..., \mathcal{A}_{fn}\}$
$\mathcal{F} := \{\}$
for each $(\mathcal{A}_1, \ldots, \mathcal{A}_n) \in \mathcal{M}_1 \times \cdots \times \mathcal{M}_n$ do
    $\mathcal{A} = \bigcup_{i \in 1,\ldots n} \mathcal{A}_i$
    $\mathcal{C} :=$ query-same-as-candidates$(\mathcal{T}_{enc}, \mathcal{A}, \mathcal{CQ}s)$
    $\mathcal{SA} := \{$same-as$(i,j) \mid (i,j) \in \mathcal{C}\}$
    $\mathcal{F} := \mathcal{F} \cup$ compute-consistent-fused-interpretations $(\mathcal{T}_{enc}, \mathcal{A}, \mathcal{SA})$
end
return select-preferred-fused-interpretations $(\mathcal{T}, \mathcal{F})$
```

Algorithm 10: The fusion algorithm *fuse*

Informally speaking, the fusion algorithm proceeds as follows:

1. For each fusion alternative, i.e. the cross product of the sets of modality-specific interpretation ABoxes, accumulate the corresponding modality-specific interpretation ABoxes into an ABox $\mathcal{A}$.

2. Obtain the set of pairs of candidate individuals $\mathcal{C}$ that probably describe the same real-world entity by posing the set of application-specific conjunctive queries $\mathcal{CQ}s$ to $\mathcal{T}_{enc} \cup \mathcal{A}$.

3. Using the pairs of individuals in $\mathcal{C}$ generate the set of same-as assertions $\mathcal{SA}$.

4. Call the function *compute-consistent-fused-interpretetions* with the enhanced background knowledge, the ABox and the generated same-as assertions to obtain the set of consistent fused interpretation ABoxes.

5. After the accumulation of all consistent fused interpretation ABoxes in $\mathcal{F}$, select preferred fused interpretations by calling the *select-preferred-fused-interpretations* function, and return them as result.

As shown in Algorithm 11, the function *query-same-as-candidates* poses a set of application-specific conjunctive queries $\mathcal{CQ}s$ in order to acquire pairs of individuals that probably describe the same real-world entity.

```
Algorithm query-same-as-candidates($\mathcal{T}_{enc}, \mathcal{A}, \mathcal{CQ}s$)
Output: a set of pairs of individuals
for each $CQ \in \mathcal{CQ}s$ where $CQ = \{(X,Y) \mid Q_1(\underline{V_1}), \ldots, Q_n(\underline{V_n})\}$ do
    $\mathcal{C} := \mathcal{C} \cup \{(\sigma(X), \sigma(Y)) \mid \mathcal{T}_{enc} \cup \mathcal{A} \models \{Q_1(\sigma(\underline{V_1})), \ldots, Q_n(\sigma(\underline{V_n}))\}$
end
return $\mathcal{C}$
```

Algorithm 11: The algorithm for identifying individuals that probably describe the same real-world entity

Different from the abduction and interpretation algorithms, the fusion algorithm *fuse* contains functions which, in the terminology of object-oriented development introduced in [GHJV93], act as 'template methods' and can be replaced/overwritten with application-specific functions. For example, instead of simply ruling out inconsistent interpretations completely, one may integrate different *repair mechanisms* into the *compute-consistent-fused-interpretetions* function.

As shown in Algorithm 12, the general *compute-consistent-fused-interpretetions* function can be defined in an application-independent way such that the consistent maximal subset of $\mathcal{A}$ is returned as result.

```
Algorithm compute-consistent-fused-interpretations($\mathcal{T}_{enc}, \mathcal{A}, \mathcal{SA}$)
Output: a set of consistent fused interpretation ABoxes
$\mathcal{C} := \{\mathcal{A}' \mid \mathcal{A}' \subseteq \mathcal{A} \cup \mathcal{SA},\ \mathcal{A}' \cup \mathcal{T} \not\models \bot\}$
return $\{\mathcal{A}' \mid \nexists \mathcal{A}'' \in \mathcal{C}, \mathcal{A}' \nsubseteq \mathcal{A}''\}$
```

Algorithm 12: The algorithm for computing the consistent maximal subset of $\mathcal{A} \cup \mathcal{SA}$

For an application scenario, the function *compute-consistent-fused-interpretetions* can be overwritten in an application-specific way to enable alternative treatments of inconsistent fused interpretations. In Section 4.3.3, we will present an application-specific *compute-consistent-fused-interpretetions* function that has been tailored towards web pages and the athletics domain in order to compensate for less reliable image analysis results.

In the same spirit, the general *select-preferred-fused-interpretations* function can be overwritten in an application-specific way as well. For example, as shown in Algorithm 13, one may want to prefer those fused interpretations that contain a maximal number of same-as assertions, in order to maximize the information gain.

```
Algorithm select-preferred-fused-interpretations($\mathcal{T}, \mathcal{F}$)
Output: a set of preferred fused interpretation ABoxes
$m := max_{\mathcal{A} \in \mathcal{F}}\ \sharp\{same\text{-}as(i,j) \mid same\text{-}as(i,j) \in \mathcal{A}\}$
return $\{\mathcal{A} \mid \mathcal{A} \in \mathcal{F},\ m = \sharp\{same\text{-}as(i,j) \mid same\text{-}as(i,j) \in \mathcal{A}\}\}$
```

Algorithm 13: An algorithm for selecting preferred fused interpretations

To study the appropriateness of our approach we have developed a software system, a so-called *semantic interpretation engine*, that implements the algorithms *interpret* and *fuse*. In Chapter 4 we present the semantic interpretation engine together with a case study of how interpretations of web pages with athletics news are computed in a practical scenario.

Chapter 4

Case Studies

This work aims to develop a declarative multimedia interpretation approach by combining existing formalisms and exploiting reasoning services. In order to show the applicability of this approach in practice, a multimedia interpretation system has been logically engineered in the previous chapter. This chapter presents *the semantic interpretation engine*, a software system for the automatic generation of deep-level semantic annotations in practical application scenarios. In addition, a detailed discussion of the interpretation of a sample multimedia document provides a case study in how the semantic interpretation engine can be used to generate deep-level semantic annotations of a multimedia document.

The semantic interpretation engine is the central component of the software system developed during the BOEMIE project. The design of the semantic interpretation engine has been influenced by the BOEMIE project and the use case of the engine is derived from this project. We therefore start with a brief introduction of the BOEMIE project in Section 4.1. Afterwards, we present both the architecture and the implementation of the semantic interpretation engine in Section 4.2. Finally, in Section 4.3, we present the stepwise interpretation of a sample web page as a case study. Based on the insights gained during the case study, we identify the appropriate processing order for fiat assertions, and appropriate values for the strategy parameter of the abductive retrieval service.

4.1 The BOEMIE Project

The BOEMIE project is a research project funded by the European Union under the Information Society Technologies program (IST-FP6-027538). The name BOEMIE is an acronym for Bootstrapping Ontology Evolution with Multimedia Information Extraction.

The BOEMIE architecture has been designed to integrate state-of-the-art software tools as components into a coherent framework [TRP$^+$07]. Using domain-specific background knowledge, the framework processes a multimedia corpus as a set of data sources and produces a repository of deep-level semantic annotations for the corpus. The design of the architecture has been determined by two principal objectives of the project: to facilitate ontology evolution and to support ontology-based retrieval of multimedia documents. Ontology evolution is defined as a bootstrapping process that aims to incrementally improve the quality and performance of both multimedia retrieval and semantics extraction from multimedia content.

The software system developed during the project has been evaluated using multimedia content such as web pages and videos. Web pages with news about athletics events have been collected from different web sites such as International Association of Athletics Federations (IAAF) and USA Track & Field (USATF) web sites [Int09, USA09]. Relevant videos of athletics events have been acquired from television broadcasters. Athletic events taking place in three major European cities, namely Athens, Berlin and London, are the application domain of BOEMIE.

The DL part of the BOEMIE background knowledge consists of three domain ontologies [DDG$^+$07], the *Athletics Event Ontology* (AEO), where all concepts and relations regarding the athletics domain are modeled, the *Multimedia Content Ontology* (MCO) that has been defined to address structural aspects of multimedia content, and the *Geographic Information Ontology* (GIO) where notions for representing geographic information are modeled. Strictly speaking, these ontologies are TBoxes, because they include terminological knowledge only. However, in the Semantic Web context, the term ontologies is widely used instead of TBoxes. In the BOEMIE project we comply with the Semantic Web nomenclature and consider the term domain ontologies as a more general name for ontologies where the ABox part is empty.

In BOEMIE, the extraction and interpretation of information from multimedia documents and the evolution of background knowledge occurs in a bootstrapping process. In each bootstrapping cycle new deep-level semantic annotations of a multimedia documents corpus are generated and stored in a repository of annotations, the so-called *BOEMIE annotation repository*. The content of multimedia documents, i.e. web pages and videos, are stored in another repository called *BOEMIE multimedia repository*. Note that annotations of a multimedia document contain not only semantic information about the content of the document but also information about the compositional structure of the document and the uniform resource locators (URLs) of the document. For example, annotations of a web page include information about the segments of the web page, such as text paragraphs and images, and the URLs of the web page as well as URLs of the images in the web page. Similarly, annotations of a video include information about the existence of audio and video OCR segments and the URL of the video. Therefore, information from the annotation repository can easily be related to content stored in the multimedia repository.

The BOEMIE project also aims to integrate existing geographic information with deep-level annotations of multimedia documents. To this end, a so-called *Geographic Information System* (GIS) is exploited by the BOEMIE system. In general, geographic information systems are special information systems for capturing, storing, managing, analyzing and displaying data which are geographically referenced [LGMR91]. The GIS used in BOEMIE provides for maps and information about various geographic points of interest like stadiums, marathon routes etc. in Athens, Berlin and London. Notice that the geographic information ontology GIO is part of the BOEMIE background knowledge, and thus, is used not only by the GIS but also by other components of the BOEMIE software system. Therefore, spatial information about points of interest from GIS is related to deep-level annotations of multimedia documents. It should be noted here that spatial information from GIS is not used for the multimedia interpretation process, but to enhance deep-level annotations of multimedia documents.

BOEMIE deals with four main tasks: information extraction, interpretation, ontology evolution and multimedia retrieval. In the following we concisely present these tasks and some software components, which have been developed for these tasks.

Information extraction Information is extracted from multimedia documents such as web pages and videos. Multimedia documents are heterogeneous data sources that generally contain information in multiple modalities. Various analysis tools process data in modalities such as text, image, video or audio in order to extract surface-level information from a single modality. For example, optical character recognition techniques are used for acquiring textual information from videos. This process is known as *Video OCR*. More precisely, Video OCR is a process that detects, segments and recognizes texts in video frames [AGP07, AGPP07]. The basic analysis tools integrated into the BOEMIE system return the extracted information in proprietary formats. The output of the analysis tools are then translated to ABoxes compliant with the AEO, MCO and GIO ontologies. Therefore, the output of the information extraction task is always a modality-specific analysis ABox [PTK+08].

Interpretation Taking modality-specific analysis ABoxes as input, interpretation aims to compute deep-level semantic annotations for multimedia documents. For the interpretation task we have developed a software component called the semantic interpretation engine, which implements the interpretation and fusion algorithms presented in Chapter 3.

The semantic interpretation engine interprets analysis ABoxes, i.e. surface-level annotations, of multimedia document segments to generate modality-specific interpretation results, i.e. deep-level annotations. Afterwards, it fuses the modality-specific interpretation results to produce interpretations of whole multimedia documents, i.e. fused interpretation ABoxes [EKM+07b, EKM08a]. In Section 4.2, we will present the semantic interpretation engine in more detail and discuss the computation of fused interpretation ABoxes by using an example web page.

Ontology evolution Ontology evolution aims to adapt the BOEMIE background knowledge used for information extraction and interpretation in order to obtain better deep-level annotations of multimedia documents in subsequent bootstrapping steps [CDD+06]. For this task an evolution toolkit has been developed that supports a domain expert in a semi-automatic enhancement process where the BOEMIE background knowledge evolves.

The evolution toolkit outputs proposals for enriching the background knowledge in terms of new concept and relations in the ontology and new rules for interpretation. To achieve this goal, it analyses the fused interpretation ABoxes produced by the semantic interpretation engine using machine learning techniques. Additionally, it consults other ontologies that deal with the athletics domain using ontology matching techniques. The proposals for change are presented to the domain expert together with summarized current interpretation results such that the domain expert obtains a holistic view and can make decisions on ontology evolution. For this purpose, a web-based application called the *BOEMIE Semantic Manager* has been developed [TRP+07]. The BOEMIE Semantic Manager supports users in viewing and managing multimedia resources and the information extracted from these content by the BOEMIE system [CFML08]. Therefore it can be considered as a content management system.

Multimedia retrieval Another important task is to develop software applications that support convenient retrieval of multimedia documents. For this reason a graphical user interface, called the *BOEMIE Semantic Browser*, has been developed [EKM09]. The BOEMIE Semantic Browser is a web-based application that can be accessed using a web browser. It has two variants: One for machines with high processing power like desktop computers and notebooks. Another one developed with particular focus on mobile devices with limited capabilities, for example a mobile phone with a small display and limited processing power.

The main goal of the BOEMIE Semantic Browser is to demonstrate how deep-level annotations generated by the BOEMIE system can be used for ontology-based access to multimedia resources enriched with geographical information. More precisely, it aims to demonstrate how semantic information can be exploited to offer innovative and convenient ways of multimedia access that goes beyond the access methods offered by conventional multimedia content management systems such as keyword-based search.

A typical use case scenario of the BOEMIE Semantic Browser is the following: A user starts accessing multimedia content based on geographic information, e.g. by accessing all videos recorded at a certain geographic location in London, where London marathon takes place. A key feature of the BOEMIE Semantic Browser is its support for ontology-based navigation. Assume that a user views a high jump image using the BOEMIE Semantic Browser. The image depicts *Blanka Vlasic*, the world's top-ranked high jumper in 2007, clearing a height. On demand, the user is provided with semantic context menus and can navigate to semantically related content using ontology-based navigation proposals, e.g. other images showing *Blanka Vlasic*, other high jump images, other jumping images or the web page to which the image shown belongs to.

Furthermore, the BOEMIE Semantic Browser can be used as a tool for debugging the information that has been extracted by the BOEMIE system. For example, a region depicting a pillar in the above-mentioned image is highlighted and tagged with the label *pillar* or the words *London Marathon* in a web page are highlighted and tagged with the label *a marathon event name*. Therefore, the BOEMIE Semantic Browser can be used to examine the information extracted by the BOEMIE system and to comprehend why a particular multimedia content is displayed as relevant.

4.2 The Semantic Interpretation Engine

In the previous section, we have briefly presented the BOEMIE project in order to explain the context in which the semantic interpretation engine has been developed. In this section, we present both architecture and implementation of the semantic interpretation engine.

The semantic interpretation engine is a software system developed for the computation of modality-specific interpretations and the fusion of these interpretations in order to obtain deep-level semantic annotations of multimedia documents. In the BOEMIE project, the semantic interpretation engine is called BIWS, an acronym for BOEMIE Interpretation Web Services. However, BIWS has been designed as a generic, stand-alone software component that can also be used in other contexts and projects. Therefore we prefer to use the more general name, namely the semantic interpretation engine, in this work.

The semantic interpretation engine supports both interpretation and fusion by implementing the interpretation and fusion algorithms presented in Section 3.4 and Section 3.5, respectively. To this end, it exploits inference services provided by the state-of-the-art DL reasoner RacerPro. Besides standard inference services such as ABox consistency testing, the semantic interpretation engine utilizes the *retrieve-with-explanation* function provided by RacerPro. RacerPro's *retrieve-with-explanation* function is an implementation of the ABox abduction algorithm *abduce* (see Algorithm 6 in Section 3.3.2).

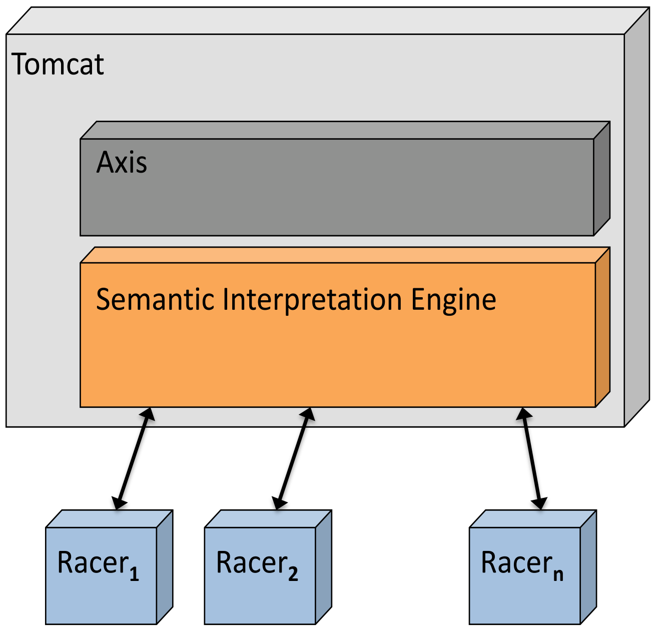


Figure 4.1: The architecture of the semantic interpretation engine, which is deployed into the Apache Tomcat servlet container. The Apache Axis is a core engine for web services. The semantic interpretation engine exploits the inference services offered by RacerPro. Each RacerPro instance is dedicated to a single modality.

Figure 4.1 illustrates the architecture and the typical deployment of the semantic interpretation engine, which has been implemented in the Java programming language as a so-called *servlet*. A servlet is a server-side Java program that can be deployed into a web server or servlet container, and accessed via a request-response programming model. A servlet can dynamically process requests and construct answers. A servlet container is responsible for managing the lifecycle of servlets, mapping a URL to a particular servlet and ensuring that the URL requester has the correct access rights. In the BOEMIE project, the semantic interpretation engine has been deployed into the Apache Tomcat servlet container [Apa09]. However, it can also be deployed into any other servlet container that complies with the servlet specification released by Sun Microsystems under the name JSR-000154 Java Servlet 2.5. Specification [Mic09].

In the BOEMIE scenario, a set of RacerPro instances is managed by the semantic interpretation engine, where for each modality a dedicated RacerPro instance is used. Most important parameters of the semantic interpretation engine, such as the set of RacerPro instances to be utilized, the domain ontologies and interpretation rule sets are declared in a configuration file to offer high flexibility. To improve scalability, the semantic interpretation can easily be configured to manage multiple RacerPro instances for each modality.

The semantic interpretation engine offers its functionality through web-based services, a.k.a. web services. This enables the development of a flexible software system where the semantic interpretation engine and other software components are loosely coupled through standardized interfaces. The web services offered by the semantic interpretation engine can be grouped into two categories: single-modality interpretation services and fusion services. In its current state, the semantic interpretation engine offers single-modality interpretation in image, text, caption, audio and videoOCR modalities. It offers fusion services for web pages and videos. In this work we only report on the interpretation of web pages consisting of image, text and caption segments.

Typically, multimedia documents such as web pages and videos contain information in multiple modalities. For example, a video is an audiovisual medium, and hence contains information both in visual and auditory modalities. A web page contains information in visual and textual modalities. In order to obtain semantic annotations of web pages and videos, the semantic interpretation engine can be used as follows: First, all segments of a multimedia document are interpreted with respect to information in a single modality using modality-specific interpretation services. Second, modality-specific interpretations of multimedia document segments are fused with respect to information about structural composition of the multimedia document using fusion services.

For example, assume that a client of the semantic interpretation engine wants to obtain semantic annotations of a web page that consists of a textual part and two images, which are captioned with text. First, the client has to call the modality-specific interpretation service for each part of the web page. More precisely, the client has to call the image interpretation service for each image, the caption interpretation service for each caption and the text interpretation service for the text separately. Each modality-specific interpretation service requires surface-level information, which have been extracted from that modality by the analysis tools, as input. The client has to provide this information in the form of an analysis ABox when calling these services. The interpretation of the web page segments are independent from each other, and thus can be called in any order. Second, the client has to call the web page fusion service that requires information about the structural composition as input. The information about the structural composition has to be provided as an ABox as well when calling the web page fusion service. In this ABox, all segments of the web page have to be represented together with information about how these segments are related to each other, e.g., which caption belongs to which image.

4.3 Interpretation of a Sample Multimedia Document

The goal of this section is to discuss the interpretation of multimedia documents in detail. In particular, we focus on the interpretation of web pages, and present the stepwise interpretation of a sample web page as a case study. However, the approach is general and can be applied to other kinds of multimedia documents as well. For example, the approach is also applicable to videos for which videoOCR and audio analysis results are available. The interpretation of video segments based on videoOCR and audio analysis results works analogously to the interpretation of web page segments. The semantic interpretation engine supports also video interpretation including the fusion of videoOCR and audio interpretation results. Due to space restrictions, in this section, we only discuss the interpretation of web pages consisting of image, text and caption segments in detail.

In the remainder of this section, at first, we use a sample web page to present the interpretation of web page segments based on modality-specific analysis results and background knowledge. Afterwards, we discuss appropriate strategies for the interpretation process including appropriate values for the strategy parameter Ω of the abductive retrieval service and the appropriate processing order for fiat assertions. Finally, we explain how modality-specific interpretations of web page segments are fused to obtain interpretations of the sample web page.

4.3.1 Modality-Specific Interpretations

Figure 4.2 shows a sample web page taken from the website of the International Association of Athletics Federations IAAF [Int09][1]. The web page consists of a text passage and an image that is captioned. The textual information in the caption complements the visual information in the image by providing additional information such as the athlete's name, performance, the city and the country where the picture was taken.

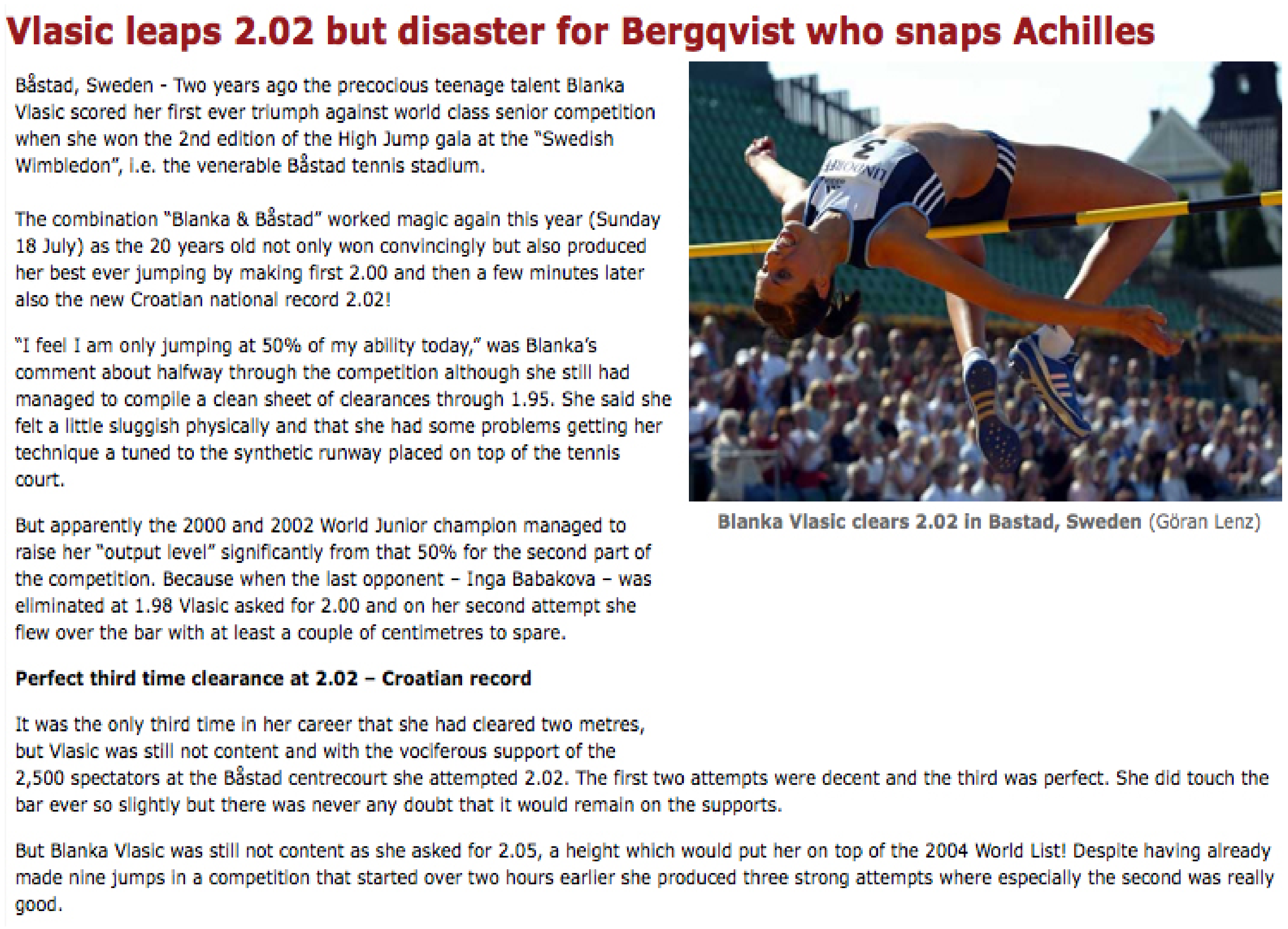
We assume that the web page in Figure 4.2 has successfully been partitioned into text, image and caption segments, and analyzed by analysis tools to obtain surface-level information in corresponding modalities. To obtain interpretations of web page segments, a client calls the corresponding interpretation web services of the semantic interpretation engine. The modality-specific interpretation web services are independent from each other, and thus, can be called in any order. In this example, we start with the interpretation of the image analysis ABox, continue with the caption analysis ABox, and conclude with the text analysis ABox.

The interpretation services offered by the semantic interpretation engine require a DL ABox in the OWL syntax as input. The Web Ontology Language OWL is a knowledge representation language that is widely used in the Semantic Web context. Various software tools such as ontology modeling tools and DL reasoners support OWL which has an XML-based syntax. In this example, however, we prefer to use a syntax inspired by the KRSS syntax [PSS93] instead of OWL, due to better readability and compactness.

The analysis results of the image depicted in Figure 4.3 are represented in the ABox *imageABox01*, which is shown in Figure 4.4. Furthermore, we assume that the TBox part of the background knowledge contains the axioms shown in Figure 4.5. The TBox $\mathcal{T}$ shown in Figure 4.5 is an excerpt from the athletics domain ontology AEO used in the BOEMIE project, and contains only the axioms that are relevant for our example.

The abbreviations *DLC* and *SLC* stand for deep-level concept and surface-level concept, respectively. All objects that are detected in an image are represented as instances of the SLC in analysis ABoxes. For example, the instances $face_1$, $body_1$ and bar_1 in Figure 4.4 are instances of the concepts $PersonFace$, $PersonBody$ and

[1]Figures containing athletics images and athletics news are reproduced here with permission granted by IAAF.



Vlasic leaps 2.02 but disaster for Bergqvist who snaps Achilles

Båstad, Sweden - Two years ago the precocious teenage talent Blanka Vlasic scored her first ever triumph against world class senior competition when she won the 2nd edition of the High Jump gala at the "Swedish Wimbledon", i.e. the venerable Båstad tennis stadium.

The combination "Blanka & Båstad" worked magic again this year (Sunday 18 July) as the 20 years old not only won convincingly but also produced her best ever jumping by making first 2.00 and then a few minutes later also the new Croatian national record 2.02!

"I feel I am only jumping at 50% of my ability today," was Blanka's comment about halfway through the competition although she still had managed to compile a clean sheet of clearances through 1.95. She said she felt a little sluggish physically and that she had some problems getting her technique a tuned to the synthetic runway placed on top of the tennis court.

Blanka Vlasic clears 2.02 in Bastad, Sweden (Göran Lenz)

But apparently the 2000 and 2002 World Junior champion managed to raise her "output level" significantly from that 50% for the second part of the competition. Because when the last opponent – Inga Babakova – was eliminated at 1.98 Vlasic asked for 2.00 and on her second attempt she flew over the bar with at least a couple of centimetres to spare.

Perfect third time clearance at 2.02 – Croatian record

It was the only third time in her career that she had cleared two metres, but Vlasic was still not content and with the vociferous support of the 2,500 spectators at the Båstad centrecourt she attempted 2.02. The first two attempts were decent and the third was perfect. She did touch the bar ever so slightly but there was never any doubt that it would remain on the supports.

But Blanka Vlasic was still not content as she asked for 2.05, a height which would put her on top of the 2004 World List! Despite having already made nine jumps in a competition that started over two hours earlier she produced three strong attempts where especially the second was really good.

Figure 4.2: A sample web page with athletics news

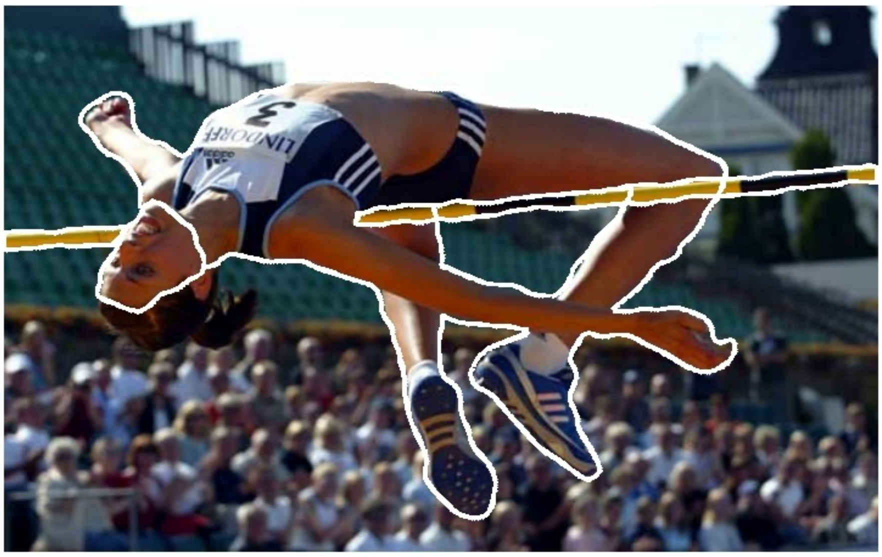


Figure 4.3: The image taken from the sample web page in Figure 4.2

$PersonFace\ (face_1)$
$PersonBody\ (body_1)$
$HorizontalBar\ (bar_1)$
$adjacent\ (face_1, body_1)$
$adjacent\ (body_1, bar_1)$

Figure 4.4: The ABox *imageABox01* representing the results of image analysis for the image in Figure 4.3

$HorizontalBar$, which are disjoint subconcepts of the SLC. All objects that are hypothesized during the interpretation process are instances of disjoint subconcepts of the DLC. The disjointness axioms are necessary to avoid 'awkward' explanations, which would otherwise be generated.

In addition to the TBox $\mathcal{T}$, the background knowledge used for image interpretation also contains a set of rules. Figure 4.6 depicts an excerpt of the image interpretation rules $\mathcal{R}_{ima}$ that are relevant for our discussion.

In Figure 4.6 the star sign (*) next to a line number indicates that the corresponding rule should be considered only if the rule set $\mathcal{R}_{ima}$ is to be applied in a forward-chaining way, whereas all other rules in $\mathcal{R}_{ima}$ are to be considered only if a query is to be

$$
\begin{array}{rcl}
\mathit{DLC} & \sqsubseteq & \neg \mathit{SLC} \\
\mathit{Person} & \sqsubseteq & \mathit{DLC} \sqcap \exists_{\leq 1}\, \mathit{hasPart}.\mathit{PersonFace} \sqcap \exists_{\leq 1}\, \mathit{hasPart}.\mathit{PersonBody} \\
\mathit{Athlete} & \sqsubseteq & \mathit{Person} \\
\mathit{Jumper} & \sqsubseteq & \mathit{Athlete} \\
\mathit{HighJumper} & \sqsubseteq & \mathit{Jumper} \\
\mathit{PoleVaulter} & \sqsubseteq & \mathit{Jumper} \\
\mathit{SportsTrial} & \sqsubseteq & \mathit{DLC} \sqcap \exists_{\leq 1}\, \mathit{hasParticipant}.\mathit{Athlete} \sqcap \neg \mathit{Person} \\
\mathit{Jumping} & \sqsubseteq & \mathit{SportsTrial} \\
\mathit{HighJump} & \sqsubseteq & \mathit{Jumping} \\
\mathit{PoleVault} & \sqsubseteq & \mathit{Jumping} \\
\mathit{HighJump} & \sqsubseteq & \exists_{\leq 1}\, \mathit{hasPart}.\mathit{HorizontalBar} \sqcap \forall\, \mathit{hasParticipant}.\mathit{HighJumper} \\
\mathit{PoleVault} & \sqsubseteq & \exists_{\leq 1}\, \mathit{hasPart}.\mathit{HorizontalBar} \sqcap \exists_{\leq 1}\, \mathit{hasPart}.\mathit{Pole} \sqcap \\
 & & \forall\, hasParticipant.PoleVaulter \sqcap \neg \mathit{HighJump} \\
\mathit{Object} & \sqsubseteq & \mathit{SLC} \sqcap \neg \mathit{OrganismPart} \\
\mathit{PersonFace} & \sqsubseteq & \mathit{OrganismPart} \\
\mathit{PersonBody} & \sqsubseteq & \mathit{OrganismPart} \sqcap \neg \mathit{PersonFace} \\
\mathit{HorizontalBar} & \sqsubseteq & \mathit{Object} \sqcap \neg \mathit{Pole}
\end{array}
$$

Figure 4.5: An excerpt of the TBox $\mathcal{T}$ for the athletics domain

$$
\begin{array}{llcl}
\mathbf{1} & \mathit{adjacent}(Y,Z) & \leftarrow & \mathit{Person}(X), \mathit{hasPart}(X,Y), \mathit{PersonFace}(Y), \\
 & & & \mathit{hasPart}(X,Z), \mathit{PersonBody}(Z) \\
\mathbf{2} & \mathit{adjacent}(Y,Z) & \leftarrow & \mathit{PoleVault}(X), \mathit{hasParticipant}(X,Y), \mathit{PoleVaulter}(Y), \\
 & & & \mathit{hasPart}(X,Z), \mathit{Pole}(Z) \\
\mathbf{3} & \mathit{adjacent}(Y,Z) & \leftarrow & \mathit{PoleVault}(X), \mathit{hasParticipant}(X,Y), \mathit{PoleVaulter}(Y), \\
 & & & \mathit{hasPart}(X,Z), \mathit{HorizontalBar}(Z) \\
\mathbf{4} & \mathit{adjacent}(Y,Z) & \leftarrow & \mathit{HighJump}(X), \mathit{hasParticipant}(X,Y), \mathit{HighJumper}(Y), \\
 & & & \mathit{hasPart}(X,Z), \mathit{HorizontalBar}(Z) \\
\mathbf{5^*} & \mathit{adjacent}(Y,Z) & \leftarrow & \mathit{Person}(Y), \mathit{hasPart}(Y,X), \mathit{adjacent}(X,Z), \mathit{Object}(Z)
\end{array}
$$

Figure 4.6: An excerpt of the image interpretation rules $\mathcal{R}_{ima}$ for the athletics domain

expanded during abduction, i.e. if $\mathcal{R}_{ima}$ is to be applied in a backward-chaining way. In fact, RacerPro's rule definition language supports the use of a parameter as part of a rule definition in order to explicitly declare whether a rule should be considered for application in a forward-chaining or backward-chaining way. This parameter is not visible in Figure 4.6, since we prefer to use a more general logic programming notation (and the star sign) instead of RacerPro's rule definition language in discussing our example. Notice that the rules for query expansion constitute a non-recursive rule set, since by definition the only recursive rule in $\mathcal{R}_{ima}$, namely the rule with the star sign, is to be considered only if $\mathcal{R}_{ima}$ is applied in a forward-chaining way.

We assume that a client who wants to obtain interpretations of the web page in Figure 4.2 calls, at first, the *interpretImage* function of the semantic interpretation engine, and provides the ABox *imageABox01* in Figure 4.4 as input. At first, the semantic interpretation engine determines the set of fiat assertions from *imageABox01*. In the current implementation, any assertion from *imageABox01* is considered as a fiat assertion, if there exists a rule $r_i \in \mathcal{R}_{ima}$ where the concept or role name of the assertion is also the predicate of a rule atom in the head of r_i. Therefore, the set *Fiats* initially contains the following assertions:

$$\mathit{Fiats} = \{(\mathit{adjacent}(\mathit{face_1}, \mathit{body_1}),\, 0),\, (\mathit{adjacent}(\mathit{body_1}, \mathit{bar_1}),\, 0)\}$$

As discussed in Section 3.4, fiat assertions from the analysis ABox have the level zero in the set *Fiats*.

The semantic interpretation engine employs the interpretation algorithm *interpret* presented in Algorithm 9 in Section 3.4. In this example, the *interpret* algorithm is initially called with the TBox $\mathcal{T}$ (Figure 4.5), the set of interpretation rules $\mathcal{R}_{ima}$ (Figure 4.6), the analysis ABox *imageABox01* (Figure 4.4) and the above-mentioned set *Fiats*.

Following the interpretation algorithm, a fiat assertion is selected from the set *Fiats*, removed from *Fiats*, and transformed into a Boolean query in order to call the abductive retrieval service. Assume that the fiat assertion $\mathit{adjacent}(\mathit{face_1}, \mathit{body_1})$ is selected first. After the removal from *Fiats*: $\mathit{Fiats}' := \mathit{Fiats} \setminus \{(\mathit{adjacent}(\mathit{face_1}, \mathit{body_1}), 0)\}$, the assertion is transformed into a Boolean query for abduction. The DL reasoner RacerPro provides the function *retrieve-with-explanation*, which is an implementation of the ABox abduction algorithm *abduce* presented in Section 3.3.2 (see Algorithm 6). The exact syntax of the RacerPro function call is as follows:

```
(retrieve-with-explanation ()
   (face1 body1 adjacent) (:reuse-old))
```

The function *retrieve-with-explanation* accepts the strategy parameter Ω that defines the strategy in instantiating variables. As discussed in Section 3.3.2, there are two possible values for Ω: 'use new individuals' and 'reuse existing individuals'. If the *retrieve-with-explanation* function is called without the optional strategy parameter, Ω has the value 'use new individuals', and thus the function prefers to hypothesize new individuals instead of reusing existing individuals when generating explanations.

The *retrieve-with-explanation* function can also be instructed to additionally generate explanations where existing individuals are reused. If the function *retrieve-with-explanation* is called with the optional parameter value *reuse-old*, which corresponds to the Ω value 'reuse existing individuals', it tries to reuse existing individuals as part of an explanation, if such individuals exist in the ABox (see Algorithm 6 in Section 3.3.2). In other words, the parameter *reuse-old* instructs the abductive retrieval service to use a certain strategy in explanation generation.[1] In the next section, we will investigate the role of the strategy parameter value in explanation generation in detail.

All rules for query expansion in Figure 4.6 have the atom *adjacent* in the head, and thus can be exploited to generate explanations for the Boolean query:

- $\Gamma_1 = \{Person(new\_ind_1),\ hasPart(new\_ind_1,\ face_1),$
  $PersonFace(face_1),\ hasPart(new\_ind_1,\ body_1),\ PersonBody(body_1)\}$
- $\Gamma_2 = \{PoleVault(new\_ind_2),\ hasParticipant(new\_ind_2,\ face_1),$
  $PoleVaulter(face_1),\ hasPart(new\_ind_2,\ body_1),\ Pole(body_1)\}$
- $\Gamma_3 = \{PoleVault(new\_ind_3),\ hasParticipant(new\_ind_3,\ face_1),$
  $PoleVaulter(face_1),\ hasPart(new\_ind_3,\ body_1),\ HorizontalBar(body_1)\}$
- $\Gamma_4 = \{HighJump(new\_ind_4),\ hasParticipant(new\_ind_4,\ face_1),$
  $HighJumper(face_1),\ hasPart(new\_ind_4,\ body_1),\ HorizontalBar(body_1)\}$

However, only Γ_1, the explanation generated using the rule at line 1 of Figure 4.6 is consistent w.r.t. $\mathcal{T}$ and $\mathcal{A}$. This is due to the disjointness axioms in $\mathcal{T}$. The *retrieve-with-explanation* function discards inconsistent explanations. Since there exists only

[1]It should be noted here that for role assertions that are transformed into a Boolean query for abduction, the semantic interpretation engine calls *retrieve-with-explanation* always with the strategy parameter value *reuse-old*.

one consistent explanation, the *retrieve-with-explanation* function computes no preference scores, and returns a single explanation, which contains the set of non-entailed assertions from Γ_1:

$$\Delta_1 = \{\mathit{Person}(\mathit{new\_ind_1}),\ \mathit{hasPart}(\mathit{new\_ind_1},\ \mathit{face_1}),\ \mathit{hasPart}(\mathit{new\_ind_1},\ \mathit{body_1})\}$$

Informally speaking, in Δ_1, the adjacency of the face and the body is explained by hypothesizing a person instance to whom they both belong to. The *retrieve-with-explanation* function generates unique names for aggregates (hypothesized instances) as needed.

Following the interpretation algorithm in Algorithm 9, the assertions from Δ_1 are added to $\mathcal{A}$: $\mathcal{A}' := \mathcal{A} \cup \Delta_1$. Figure 4.7 depicts the ABox $\mathcal{A}'$ at this stage:

$\mathit{PersonFace}\ (\mathit{face_1})$
$\mathit{PersonBody}\ (\mathit{body_1})$
$\mathit{HorizontalBar}\ (\mathit{bar_1})$
$\mathit{adjacent}\ (\mathit{face_1}, \mathit{body_1})$
$\mathit{adjacent}\ (\mathit{body_1}, \mathit{bar_1})$
$\mathit{Person}\ (\mathit{new\_ind_1})$
$\mathit{hasPart}(\mathit{new\_ind_1}, \mathit{face_1})$
$\mathit{hasPart}(\mathit{new\_ind_1}, \mathit{body_1})$

Figure 4.7: The ABox $\mathcal{A}'$ after the addition of Δ_1

In the next step, the rules in $\mathcal{R}_{ima}$ are applied in a forward-chaining way by calling the *execute-or-reexecute-all-rules* function of RacerPro which is an implementation of the *apply-rules* function introduced in Section 3.1.2 (see Algorithm 2). By definition, the rule at line 5 of Figure 4.6 is the only rule in $\mathcal{R}_{ima}$ that has to be considered by the *execute-or-reexecute-all-rules* function. The premises of this rule are proven to be true w.r.t. $\mathcal{A}'$ in Figure 4.7 such that the atom in its head, the consequence, must be true as well. Therefore, the result set of the *execute-or-reexecute-all-rules* function call is: C=$\{\mathit{adjacent}(\mathit{new\_ind_1}, \mathit{bar_1})\}$. Since $\mathcal{A}' \cup C$ is consistent w.r.t. $\mathcal{T}$, the algorithm *interpret* is called recursively with the following parameters: $\mathcal{T}$, $\mathcal{R}_{ima}$, $\mathcal{A}' \cup C$, $\mathit{Fiats}' \cup \{(c, \mathit{level} + 1) \mid c \in \mathcal{C}\}$.

In the new *interpret* function call, the set *Fiats* contains following assertions:

$Fiats = \{(adjacent(body_1, bar_1), 0), (adjacent(new\_ind_1, bar_1), 1)\}$

Next, the assertion with the lowest level from *Fiats*, namely $adjacent(body_1, bar_1)$, is selected. Analogous to the first fiat assertion, the assertion $adjacent(body_1, bar_1)$ is removed from *Fiats*, transformed to a query to call the *retrieve-with-explanation* function. However, this time the *retrieve-with-explanation* function delivers no answers, because all explanations that can be generated using the rules in $\mathcal{R}_{ima}$ are inconsistent.

Next, the interpretation algorithm is called recursively. In the new *interpret* function call, the only assertion from *Fiats*, namely $adjacent(new\_ind_1, bar_1)$ is selected. The assertion is removed from *Fiats*, and transformed into a Boolean query for abduction:

```
(retrieve-with-explanation ()
   (new_ind1 bar1 adjacent) (:reuse-old))
```

For this query only the explanations that can be generated using the rules at lines 2, 3 and 4 in Figure 4.6 are consistent w.r.t. $\mathcal{T}$ and $\mathcal{A}$. These explanations are as follows:

- $\Gamma_5 = \{PoleVault(new\_ind_5), hasParticipant(new\_ind_5, new\_ind_1), PoleVaulter(new\_ind_1), hasPart(new\_ind_5, new\_ind_6), Pole(new\_ind_6)\}$
- $\Gamma_6 = \{PoleVault(new\_ind_7), hasParticipant(new\_ind_7, new\_ind_1), PoleVaulter(new\_ind_1), hasPart(new\_ind_7, bar_1), HorizontalBar(bar_1)\}$
- $\Gamma_7 = \{HighJump(new\_ind_8), hasParticipant(new\_ind_8, new\_ind_1), HighJumper(new\_ind_1), hasPart(new\_ind_8, bar_1), HorizontalBar(bar_1)\}$

where Γ_5 is generated using the rule at line 2, Γ_6 the rule at line 3, and Γ_7 the rule at line 4 in Figure 4.6. At this stage, this sample image interpretation provides an interesting example to discuss how the preference score S presented in Section 3.3.3 is calculated for a practical example in order to deliver 'preferred' explanations only.

As discussed in Section 3.3.3, the preference score S reflects the assumption that, in the context of multimedia interpretation, an explanation is to be preferred over others if it is more consilient and simpler. The approximated consilience value S_a and the simplicity value S_h have been defined follows:

$S_a(\mathcal{T}, \mathcal{A}, \Gamma) := \sharp\{\gamma \in \Gamma \mid \mathcal{T} \cup \mathcal{A} \models \gamma\}$
$S_h(\mathcal{T}, \mathcal{A}, \Gamma) := \sharp\{\gamma \in \Gamma \mid \mathcal{T} \cup \mathcal{A} \not\models \gamma\}$

whereas the preference score S has been defined as (see Equation 3.6):

$S(\mathcal{T}, \mathcal{A}, \Gamma) = S_a(\mathcal{T}, \mathcal{A}, \Gamma) - S_h(\mathcal{T}, \mathcal{A}, \Gamma)$

Getting back to our example, the explanation Γ_5 contains no assertion that logically follows from $\mathcal{T} \cup \mathcal{A}$. Hence, $S_a(\mathcal{T}, \mathcal{A}, \Gamma_5)$=0, $S_h(\mathcal{T}, \mathcal{A}, \Gamma_5)$=5, and $S(\mathcal{T}, \mathcal{A}, \Gamma_5)$=0-5=-5. Regarding the explanation Γ_6, the only assertion that logically follows from $\mathcal{T} \cup \mathcal{A}$ is *HorizontalBar*(bar_1). Therefore $S_a(\mathcal{T}, \mathcal{A}, \Gamma_6)$=1, $S_h(\mathcal{T}, \mathcal{A}, \Gamma_6)$=4, and $S(\mathcal{T}, \mathcal{A}, \Gamma_6)$=1-4=-3. Also for the explanation Γ_7, the only assertion that logically follows from $\mathcal{T} \cup \mathcal{A}$ is *HorizontalBar*(bar_1), thus $S_a(\mathcal{T}, \mathcal{A}, \Gamma_7)$=1, $S_h(\mathcal{T}, \mathcal{A}, \Gamma_7)$=4, and $S(\mathcal{T}, \mathcal{A}, \Gamma_7)$=1-4=-3.

As discussed in Section 3.3.3, in the case of two explanations with the same preference score, the most-specific one is preferred. For this purpose, the relaxed ABox entailment inference service offered by RacerPro is used to check whether one of the explanations is more specific than the other one. In our example, the relaxed ABox entailment relationship does not hold between Γ_6 and Γ_7: $\Gamma_6 \not\rightsquigarrow \Gamma_7$ and $\Gamma_7 \not\rightsquigarrow \Gamma_6$.

It should be noted that RacerPro's *retrieve-with-explanation* function implements the preference score S. Therefore the answer to the above-mentioned *retrieve-with-explanation* call returns Δ_6 and Δ_7, the sets of non-entailed assertions from the explanations Γ_6 and Γ_7, since both Γ_6 and Γ_7 have the highest preference score, and none of them is more specific than the other one.

Following the interpretation algorithm, for each one of the two explanations the assertions from the explanation are added to $\mathcal{A}$: $\mathcal{A}' := \mathcal{A} \cup \Delta$, and then the function *execute-or-reexecute-all-rules* is called to apply the rules in $\mathcal{R}_{ima}$ in a forward-chaining way. In this example, for both explanations there are no new consequences and the function call delivers an empty set C. Furthermore, in both cases $\mathcal{A}' \cup C$ is consistent w.r.t. $\mathcal{T}$, and the *interpret* function is called recursively with an empty set of fiat assertions. Consequently, both *interpret* function calls return an ABox, and both ABoxes are accumulated in the set $\mathcal{I}$.

Finally, the *interpret* algorithm terminates by returning the set $\mathcal{I}$ containing the two interpretation ABoxes depicted in Figure 4.8. These two interpretation ABoxes represent the two most preferred interpretations of the image in Figure 4.3 that the

$PersonFace\ (face_1)$	$PersonFace\ (face_1)$
$PersonBody\ (body_1)$	$PersonBody\ (body_1)$
$HorizontalBar\ (bar_1)$	$HorizontalBar\ (bar_1)$
$adjacent\ (face_1, body_1)$	$adjacent\ (face_1, body_1)$
$adjacent\ (body_1, bar_1)$	$adjacent\ (body_1, bar_1)$
$Person\ (new\_ind_1)$	$Person\ (new\_ind_1)$
$hasPart(new\_ind_1, face_1)$	$hasPart(new\_ind_1, face_1)$
$hasPart(new\_ind_1, body_1)$	$hasPart(new\_ind_1, body_1)$
$PoleVault\ (new\_ind_7)$	$HighJump\ (new\_ind_8)$
$hasParticipant(new\_ind_7, new\_ind_1)$	$hasParticipant(new\_ind_8, new\_ind_1)$
$PoleVaulter\ (new\_ind_1)$	$HighJumper\ (new\_ind_1)$
$hasPart(new\_ind_7, bar_1)$	$hasPart(new\_ind_8,\ bar_1)$

Figure 4.8: The interpretation ABoxes *imageABox01_interpretation1* and *imageABox01_interpretation2* returned by the semantic interpretation engine

semantic interpretation engine can compute with respect to the background knowledge and image analysis results. Consequently, the semantic interpretation engine answers the client's call of the *interpretImage* web service for the analysis ABox in Figure 4.4 with the two interpretation ABoxes shown in Figure 4.8. In the answer, the interpretation ABoxes are given some unique names, namely *imageABox01_interpretation1* and *imageABox01_interpretation2*. Notice that these names include the name of the analysis ABox, which has been provided by the client when calling the *interpretImage* web service. This is important, since during the fusion process, the semantic interpretation engine has to identify all possible interpretations of a document segment based on the segment's name in order to consider all fusion alternatives. In Section 4.3.3 we will discuss the fusion process using practical examples.

To continue the interpretation of the sample web page in Figure 4.2, assume that, in the next step, the client wants to obtain interpretations of the text from the caption of the image. Figure 4.9 depicts the caption of the image shown in Figure 4.3.

Blanka Vlasic clears 2.02 in Bastad, Sweden

Figure 4.9: The caption of the image shown in Figure 4.3

The underlined words in Figure 4.9 are key entities of the text that are extracted by the text analysis processes. The analysis results of the caption in Figure 4.9 are represented in the ABox *captionABox01*, which is shown in Figure 4.10.

$\mathit{PersonName}\ (\mathit{pName}_1)$
$\mathit{Performance}\ (\mathit{perf}_1)$
$\mathit{CityName}\ (\mathit{ciName}_1)$
$\mathit{CountryName}\ (\mathit{coName}_1)$
$\mathit{personNameToPerformance}\ (\mathit{pName}_1, \mathit{perf}_1)$
$\mathit{hasValue}\ (\mathit{pName}_1, \text{‘}\mathit{Blanka\ Vlasic}\text{’})$
$\mathit{hasValue}\ (\mathit{perf}_1, \text{‘}\mathit{2.02}\text{’})$
$\mathit{hasValue}\ (\mathit{ciName}_1, \text{‘}\mathit{Bastad}\text{’})$
$\mathit{hasValue}\ (\mathit{coName}_1, \text{‘}\mathit{Sweden}\text{’})$

Figure 4.10: The ABox *captionABox01* representing the results of text analysis for the caption in Figure 4.9

In addition, we assume that the TBox part $\mathcal{T}$ of the background knowledge contains the axioms shown in Figure 4.11. Analogous to the image modality, all objects that are detected by low-level analysis processes are instances of the SLC, whereas all objects that are hypothesized during the interpretation process are instances of the DLC. All subconcepts of the concepts SLC and DLC are disjoint from each other in order to discard 'awkward' explanations. Notice that in *captionABox01 personNameToPerformance* is a role atom, whereas *hasValue* is a concrete domain predicate. In [EKM08b], we have published patterns for ontology and rule design that facilitate the design of background knowledge for multimedia interpretation.

The background knowledge contains also a set of rules for the interpretation of texts from the captions of images. Figure 4.12 depicts an excerpt of the caption interpretation rules $\mathcal{R}_{cap}$ that are relevant for our example. The rule at line 4, by definition, is to be considered only if $\mathcal{R}_{cap}$ is applied in a forward-chaining way, whereas all other rules in $\mathcal{R}_{cap}$ are to be considered only if $\mathcal{R}_{cap}$ is applied in a backward-chaining way, i.e. for abductive inference.

We assume that the client calls the *interpretCaption* function of the semantic interpretation engine, and provides the ABox *captionABox01* in Figure 4.10 as input. Before the application of the interpretation algorithm, it has to be determined which

$$\begin{array}{rcl}
\mathit{DLC} & \sqsubseteq & \neg \mathit{SLC} \\
\mathit{Person} & \sqsubseteq & \mathit{DLC} \sqcap \exists\, \mathit{hasPersonName}.\mathit{PersonName} \sqcap \neg \mathit{SportsTrial} \\
\mathit{Athlete} & \sqsubseteq & \mathit{Person} \\
\mathit{SportsTrial} & \sqsubseteq & \mathit{DLC} \sqcap \exists_{\leq 1}\, \mathit{hasParticipant}.\mathit{Athlete} \\
 & & \exists_{\leq 1}\, \mathit{hasPerformance}.\mathit{Performance} \\
\mathit{Name} & \sqsubseteq & \mathit{SLC} \sqcap \exists\, \mathit{hasValue}.\mathit{string} \sqcap \neg \mathit{Performance} \\
\mathit{PersonName} & \sqsubseteq & \mathit{Name} \sqcap \neg \mathit{CityName} \\
\mathit{CityName} & \sqsubseteq & \mathit{Name} \sqcap \neg \mathit{CountryName} \\
\mathit{CountryName} & \sqsubseteq & \mathit{Name} \sqcap \neg \mathit{PersonName} \\
\mathit{Performance} & \sqsubseteq & \mathit{SLC} \sqcap \exists\, \mathit{hasValue}.\mathit{string}
\end{array}$$

Figure 4.11: Another excerpt of the TBox $\mathcal{T}$ for the athletics domain

$$\begin{array}{lll}
\mathbf{1}\ \ \mathit{PersonName}(Y) & \leftarrow & \mathit{Person}(X), \mathit{hasPersonName}(X, Y) \\
\mathbf{2}\ \ \mathit{Performance}(Y) & \leftarrow & \mathit{SportsTrial}(X), \mathit{hasPerformance}(X,\, Y) \\
\mathbf{3}\ \ \mathit{personToPerformance}(Y, Z) & \leftarrow & \mathit{SportsTrial}(X), \mathit{hasParticipant}(X, Y), \\
 & & \mathit{Athlete}(Y), \mathit{hasPerformance}(X, Z), \\
 & & \mathit{Performance}(Z) \\
\mathbf{4^*}\ \ \mathit{personToPerformance}(Y, Z) & \leftarrow & \mathit{Person}(Y), \mathit{hasPersonName}(Y, X), \\
 & & \mathit{PersonName}(X), \mathit{Performance}(Z), \\
 & & \mathit{personNameToPerformance}(X, Z)
\end{array}$$

Figure 4.12: An excerpt of the caption interpretation rules $\mathcal{R}_{cap}$ for the athletics domain

assertions from the ABox *captionABox01* are fiat assertions. Any assertion from *captionABox01* is considered to require a fiat, if there exists a rule $r_i \in \mathcal{R}_{cap}$ where the concept or role name of the assertion is also the predicate of a rule atom in the head of r_i. Consequently, in our example, the set *Fiats* initially contains the following assertions:

$\mathit{Fiats} = \{(\mathit{PersonName}(\mathit{pName_1}),\, 0),\ (\mathit{Performance}(\mathit{perf_1}),\, 0)\}$

In this example, the interpretation algorithm incorporated in the semantic interpretation engine takes the TBox $\mathcal{T}$, the rule set $\mathcal{R}_{cap}$, the ABox *captionABox01* and the above-mentioned set *Fiats* as input.

According to the interpretation algorithm, a fiat assertion has to be selected from *Fiats* in the beginning. Assume that the assertion $\mathit{PersonName}(\mathit{pName_1})$ is selected first. It is then removed from the set *Fiats*. Later, the assertion is transformed into a Boolean query for abduction:

```
(retrieve-with-explanation ()
   (pName1 PersonName))
```

It should be noted that for *concept* assertions that are transformed into a Boolean query for abduction, the semantic interpretation engine always calls the function *retrieve-with-explanation* without the parameter *reuse-old.* The *retrieve-with-explanation* function, by default, omits the generation of explanations where existing individuals are reused. In the next section, based on examples we discuss the reasons why the parameter *reuse-old* has to be used in the case of fiat *role* assertions only.

The rule set $\mathcal{R}_{cap}$ contains only a single rule that has the concept name *PersonName* in the head, namely the rule at line 1 in Figure 4.12. For this reason, the function *retrieve-with-explanation* considers only this rule for explanation generation and returns the following explanation:

$$\Delta_1 = \{Person(new\_ind_9),\ hasPersonName(new\_ind_9,\ pName_1)\}$$

Informally speaking, the appearance of a person name in the caption is explained through the hypothesization of a person instance, to whom the name belongs to. It should be noted that the names of hypothesized individuals are unique also across modalities. This is an essential requirement, since interpretations of segments of a multimedia document are fused later.

Following the interpretation algorithm, the assertions from Δ_1 are added to $\mathcal{A}$: $\mathcal{A}'$ $:= \mathcal{A} \cup \Delta_1$. Next, the function *execute-or-reexecute-all-rules* is called to apply the rules in $\mathcal{R}_{cap}$ to $\mathcal{A}'$ in a forward-chaining way. By definition, the rule at line 4 in Figure 4.12 is the only rule that should be considered. All premises of this rule are proven to be true such that its consequence must be true as well. Consequently, the function *execute-or-reexecute-all-rules* returns the set $C{=}\{personToPerformance(new\_ind_9, perf_1)\}$. Since $\mathcal{A}' \cup C$ is consistent w.r.t. $\mathcal{T}$, the algorithm *interpret* is called recursively with the following parameters: $\mathcal{T}$, $\mathcal{R}_{cap}$, $\mathcal{A}' \cup C$, $Fiats' \cup \{(c, level + 1) \mid c \in \mathcal{C}\}$.

In the new *interpret* function call, the set *Fiats* contains following assertions:

$$Fiats = \{(Performance(perf_1),\ 0),\ (personToPerformance(new\_ind_9, perf_1),\ 1)\}$$

Since *Performance*(*perf₁*) is the fiat assertion with the lowest level, it is selected next. The assertion is first removed from *Fiats*, and then transformed into the following Boolean query:

```
(retrieve-with-explanation ()
   (perf1 Performance))
```

To answer this query, only the rule at line 2 in Figure 4.12 can be exploited, because it is the only rule with the appropriate predicate in the head. The explanation generated using that rule is as follows:

$\Delta_2 = \{\mathit{SportsTrial}(\mathit{new\_ind_{10}}),\ \mathit{hasPerformance}(\mathit{new\_ind_{10}},\ \mathit{perf_1})\}$

According to the interpretation algorithm, the assertions from Δ_2 are added to the ABox $\mathcal{A}$: $\mathcal{A}' := \mathcal{A} \cup \Delta_2$, and the *execute-or-reexecute-all-rules* function is called to apply the rules in $\mathcal{R}_{cap}$ in a forward-chaining way. However, this time there are no consequences, and the function call delivers an empty set C. Since $\mathcal{A}' \cup C$ is consistent w.r.t. $\mathcal{T}$, the *interpret* function is called recursively.

In the new *interpret* function call, the set *Fiats* contains a single assertion, namely $\mathit{personToPerformance}(\mathit{new\_ind_9}, \mathit{perf_1})$ which is selected. The assertion is removed from *Fiats*, and transformed into a Boolean query for abduction:

```
(retrieve-with-explanation ()
   (new_ind9 perf1 personToPerformance) (:reuse-old))
```

For this query the only rule that can be exploited for explanation generation is the rule at line 3 in Figure 4.12. Notice that also the rule at line 4 in Figure 4.12 contains the role name *personToPerformance* as the predicate in the rule head. However, that rule is, by definition, not considered for explanation generation. Consequently, the explanation to the query is as follows:

$\Gamma_3 = \{\mathit{SportsTrial}(\mathit{new\_ind_{10}}),\ \mathit{hasParticipant}(\mathit{new\_ind_{10}},\ \mathit{new\_ind_9}),$
$\mathit{Athlete}(\mathit{new\_ind_9}),\ \mathit{hasPerformance}(\mathit{new\_ind_{10}},\ \mathit{perf_1}),\ \mathit{Performance}\ (\mathit{perf_1})\}$

Notice that the function *retrieve-with-explanation* is instructed to prefer the reuse of existing individuals over the hypothesization of new individuals using the *reuse-old* parameter. Therefore the existing instances of the concepts *SportsTrial* and *Performance* in $\mathcal{A}$, namely (new_ind_{10}) and ($perf_1$), are reused in Γ_3.

Following the interpretation algorithm, the set of non-entailed assertions from Γ_3, namely Δ_3, is added to $\mathcal{A}$: $\mathcal{A}' := \mathcal{A} \cup \Delta_3$, and the function *execute-or-reexecute-all-rules* is called to apply the rules in $\mathcal{R}_{cap}$ to $\mathcal{A}'$ in a forward-chaining way. However, also this time there are no consequences, and the function call delivers an empty set C. Since $\mathcal{A}' \cup C$ is consistent w.r.t. $\mathcal{T}$, the *interpret* function is called recursively with an empty set of fiat assertions. Consequently, the new *interpret* function call returns an ABox as result, which is accumulated in the set $\mathcal{I}$.

Finally, the *interpret* algorithm terminates by returning the set $\mathcal{I}$ containing the ABox depicted in Figure 4.13. Consequently, the semantic interpretation engine answers the client's call of the *interpretCaption* function with the ABox shown in Figure 4.13. In the answer, the interpretation ABox is given the unique name *caption-ABox01_interpretation1.*

To conclude the modality-specific interpretation of segments of the sample web page in Figure 4.2, assume that the client wants to obtain an interpretation of the text segment as well. In the sample web page, the text segment is the main information source and contains most information. Due to space restrictions, in this work, we only present the interpretation of a single paragraph of the text. However, the same approach applies analogously to the rest of the text segment.

Figure 4.14 depicts the first paragraph of the text segment of the sample web page in Figure 4.2, which we use to discuss the text interpretation service. The underlined words in Figure 4.14 are key entities of the text that are extracted by the text analysis processes. Figure 4.15 shows the analysis ABox of the text segment in Figure 4.14.

Furthermore, we assume that the TBox part $\mathcal{T}$ of the knowledge base contains the axioms depicted in Figure 4.16. Notice that in $\mathcal{T}$, *HighJumpCompetition* and *PoleVaultCompetition* are modeled as disjoint subconcepts of the more general concept *SportsCompetition*. Figure 4.16 depicts only a small part of the TBox $\mathcal{T}$. In fact, there exist many other subconcepts of the concept *SportsCompetition* such as *HurdlingCompetition* and *LongJumpCompetition*. However, all subconcepts of the concept *SportsCompetition* are disjoint from each other, and hence are not relevant for

$PersonName\ (pName_1)$
$Performance\ (perf_1)$
$CityName\ (ciName_1)$
$CountryName\ (coName_1)$
$personNameToPerformance\ (pName_1, perf_1)$
$hasValue\ (pName_1, \text{‘}Blanka\ Vlasic\text{’})$
$hasValue\ (perf_1, \text{‘}2.02\text{’})$
$hasValue\ (ciName_1, \text{‘}Bastad\text{’})$
$hasValue\ (coName_1, \text{‘}Sweden\text{’})$
$Person\ (new\_ind_9)$
$hasPersonName\ (new\_ind_9, pName_1)$
$personToPerformance\ (new\_ind_6, perf_1)$
$SportsTrial\ (new\_ind_{10})$
$hasPerformance\ (new\_ind_{10}, perf_1)$
$hasParticipant\ (new\_ind_{10}, new\_ind_9)$
$Athlete\ (new\_ind_9)$

Figure 4.13: The interpretation ABox *captionABox01_interpretation1* returned by the semantic interpretation engine

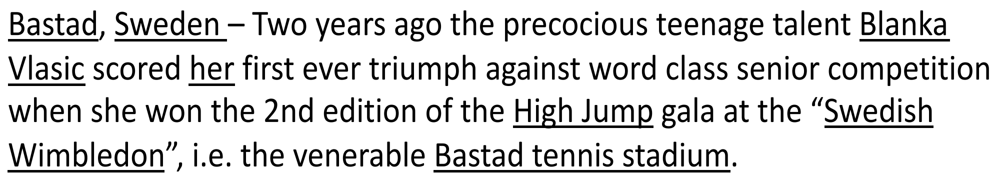

Bastad, Sweden – Two years ago the precocious teenage talent Blanka Vlasic scored her first ever triumph against word class senior competition when she won the 2nd edition of the High Jump gala at the “Swedish Wimbledon”, i.e. the venerable Bastad tennis stadium.

Figure 4.14: The first paragraph of the text segment of the sample web page

this example. Even if an explanation would have been generated to explain the appearance of a *HighJumpName* instance using subconcepts of *SportsCompetition* other than *HighJumpCompetition*, it would become inconsistent, and would be discarded.

For example, if the appearance of a *HighJumpName* is explained by hypothesizing a *PoleVaultCompetition* instance, which is in a *hasSportsName* relation with the *HighJumpName* instance, then that explanation is inconsistent w.r.t. $\mathcal{T}$, because a *PoleVaultCompetition* can only be in a *hasSportsName* relation with a *PoleVaultName* instance, and *PoleVaultName* and *HighJumpName* are disjoint concepts.

$\mathit{CityName}\ (\mathit{ciName}_2)$
$\mathit{CountryName}\ (\mathit{coName}_2)$
$\mathit{PersonName}\ (\mathit{pName}_2)$
$\mathit{Gender}\ (\mathit{gen}_2)$
$\mathit{HighJumpName}\ (\mathit{hjName}_2)$
$\mathit{StadiumName}\ (\mathit{stName}_2)$
$\mathit{StadiumName}\ (\mathit{stName}_3)$
$\mathit{hasValue}\ (\mathit{ciName}_2, \text{‘}\mathit{Bastad}\text{’})$
$\mathit{hasValue}\ (\mathit{coName}_2, \text{‘}\mathit{Sweden}\text{’})$
$\mathit{hasValue}\ (\mathit{pName}_2, \text{‘}\mathit{Blanka\ Vlasic}\text{’})$
$\mathit{hasValue}\ (\mathit{hjName}_2, \text{‘}\mathit{High\ Jump}\text{’})$
$\mathit{hasValue}\ (\mathit{stName}_2, \text{‘}\mathit{Swedish\ Wimbledon}\text{’})$
$\mathit{hasValue}\ (\mathit{stName}_3, \text{‘}\mathit{Bastad\ Tennis\ Stadium}\text{’})$
$\mathit{personNameToGender}\ (\mathit{pName}_2, \mathit{gen}_2)$
$\mathit{spoNameToStaName}\ (\mathit{hjName}_2, \mathit{stName}_2)$
$\mathit{spoNameToStaName}\ (\mathit{hjName}_2, \mathit{stName}_3)$

Figure 4.15: The ABox *textABox01* representing the results of text analysis for the text segment in Figure 4.14

In addition to $\mathcal{T}$, the background knowledge includes also a set of interpretation rules for text segments. Figure 4.17 depicts an excerpt of the text interpretation rules $\mathcal{R}_{tex}$ that are relevant for the example text segment in Figure 4.14.

Different from the set of image interpretation rules $\mathcal{R}_{ima}$ and the set of caption interpretation rules $\mathcal{R}_{cap}$, the set of text interpretation rules $\mathcal{R}_{tex}$ contains no rules that should be considered when applying rules in a forward-chaining way. In other words, the excerpt of the text interpretation rules $\mathcal{R}_{tex}$ consists of rules that have to be considered for abduction only. For the sake of brevity, the excerpt in Figure 4.17 includes only such rules that are essential for the interpretation of the text analysis ABox *textABox01* in Figure 4.15.[1]

We assume that the client calls the *interpretText* function of the semantic interpretation engine, and provides the ABox *textABox01* as input. In this case, the semantic interpretation engine determines fiat assertions from the ABox *textABox01* w.r.t. the

[1]The complete set of text interpretation rules used in the BOEMIE project contains also rules that have to be considered when applying rules in a forward-chaining way.

$$
\begin{array}{rcl}
DLC & \sqsubseteq & \neg SLC \\
Person & \sqsubseteq & DLC \sqcap \neg SportsCompetition \sqcap \exists_{\leq 1} hasGender.Gender \sqcap \\
 & & \exists\, hasPersonName.PersonName \\
SportsCompetition & \sqsubseteq & DLC \sqcap \exists\, hasSportsName.SportsName \sqcap \\
 & & \exists\, takesPlaceIn.StadiumName \\
HighJumpCompetition & \sqsubseteq & SportsCompetition \sqcap \forall\, hasSportsName.HighJumpName \\
PoleVaultCompetition & \sqsubseteq & SportsCompetition \sqcap \forall\, hasSportsName.PoleVaultName \\
Name & \sqsubseteq & SLC \sqcap \exists\, hasValue.string \\
PersonName & \sqsubseteq & Name \sqcap \neg CityName \sqcap \neg StadiumName \\
CityName & \sqsubseteq & Name \sqcap \neg CountryName \sqcap \neg SportName \\
CountryName & \sqsubseteq & Name \sqcap \neg PersonName \sqcap \neg StadiumName \\
StadiumName & \sqsubseteq & Name \sqcap \neg SportName \sqcap \neg CityName \\
SportsName & \sqsubseteq & Name \sqcap \neg CountryName \sqcap \neg PersonName \\
HighJumpName & \sqsubseteq & SportsName \\
PoleVaultName & \sqsubseteq & SportsName \sqcap \neg HighJumpName \\
Gender & \sqsubseteq & SLC \sqcap \neg Name
\end{array}
$$

Figure 4.16: Another excerpt of the TBox $\mathcal{T}$ for the athletics domain

$$
\begin{array}{llll}
\mathbf{1} & PersonName(Y) & \leftarrow & Person(X), hasPersonName(X,Y) \\
\mathbf{2} & personNameToGender(Y,Z) & \leftarrow & Person(X), hasPersonName(X,Y), \\
 & & & PersonName(Y), hasGender(X,Z), \\
 & & & Gender(Z) \\
\mathbf{3} & spoNameToStaName(Y,Z) & \leftarrow & SportsCompetition(X), \\
 & & & hasSportsName(X,Y), SportsName(Y), \\
 & & & takesPlaceIn(X,Z), StadiumName(Z) \\
\mathbf{4} & spoNameToStaName(Y,Z) & \leftarrow & HighJumpCompetition(X), \\
 & & & hasSportsName(X,Y), HighJumpName(Y), \\
 & & & takesPlaceIn(X,Z), StadiumName(Z) \\
\mathbf{5} & spoNameToStaName(Y,Z) & \leftarrow & PoleVaultCompetition(X), \\
 & & & hasSportsName(X,Y), PoleVaultName(Y), \\
 & & & takesPlaceIn(X,Z), StadiumName(Z)
\end{array}
$$

Figure 4.17: An excerpt of the text interpretation rules $\mathcal{R}_{tex}$ for the athletics domain

text interpretation rules in $\mathcal{R}_{tex}$. Following the same strategy as for the previous modalities, any assertion from the ABox *textABox01* is considered as a fiat assertion, if there exists a rule $r_i \in \mathcal{R}_{tex}$ where the concept or role name of the assertion from *textABox01* is also the predicate of a rule atom in the head of r_i. In this example, the set *Fiats* initially contains the following assertions:

$Fiats = \{(PersonName\ (pName_2), 0),$
$(personNameToGender(pName_2, gen_2), 0),$
$(spoNameToStaName(hjName_2, stName_2), 0),$
$(spoNameToStaName(hjName_2, stName_3), 0)\}$

The interpretation algorithm is initially called with the TBox $\mathcal{T}$, the rule set $\mathcal{R}_{tex}$, the ABox *textABox01*, and the above-mentioned *Fiats* set. The algorithm starts with the selection of a fiat assertion from *Fiats*. If the set *Fiats* contains both concept and role assertions like in this example, then the algorithm selects concept assertions first in order to obtain 'preferred' explanations. In the next section we discuss the reasons why concept assertions should be explained first.

In this example, the only fiat concept assertion, namely $PersonName(pName_2)$, is selected first. The assertion is removed from *Fiats* and transformed into a Boolean query. Consequently, the RacerPro instance dedicated for text interpretation is called as follows:

```
(retrieve-with-explanation ()
   (pName2 PersonName))
```

The rule at line 1 in Figure 4.17 is the only rule with the corresponding predicate in the rule head, and hence the only explanation generated as an answer to the query is:

$\Delta_1 = \{Person(new\_ind_{11}), hasPersonName(new\_ind_{11},\ pName_2)\}$

Following the interpretation algorithm, the assertions from Δ_1 are added to $\mathcal{A}$: $\mathcal{A}'$:= $\mathcal{A} \cup \Delta_1$. To apply the rules in $\mathcal{R}_{tex}$ to $\mathcal{A}'$ in a forward-chaining way the function *execute-or-reexecute-all-rules* is called. The function returns an empty set C, because all rules in $\mathcal{R}_{tex}$ are declared as rules that should only be considered for abduction. Since $\mathcal{A}' \cup C$ is consistent w.r.t. $\mathcal{T}$, the algorithm *interpret* is called recursively with the following parameters: $\mathcal{T}$, $\mathcal{R}_{tex}$, $\mathcal{A}' \cup C$, $Fiats' \cup \{(c, level + 1) \mid c \in \mathcal{C}\}$, where $C = \{\}$.

In the new *interpret* function call, the set *Fiats* contains the following assertions:

$Fiats = \{(personNameToGender(pName_2, gen_2),\ 0),$
$(spoNameToStaName(hjName_2, stName_2),\ 0),$
$(spoNameToStaName(hjName_2, stName_3),\ 0)\}$

Assume that $personNameToGender(pName_2, gen_2)$ is selected as the next fiat assertion to be explained. Again, the assertion is removed from *Fiats*, and then transformed into a Boolean query for abduction. The corresponding function call is as follows:

```
(retrieve-with-explanation ()
       (pName2 gen2 personNameToGender) (:reuse-old))
```

If a fiat role assertion such as $personNameToGender(pName_2, gen_2)$ is transformed to a query for abduction, then the function *retrieve-with-explanation* is always called with the parameter *reuse-old*. This parameter instructs the reasoner to consider not only the hypothesization of new individuals but also the reuse of existing individuals when answering the Boolean query. Therefore there exist two possible answers to the above-mentioned query:

- $\Gamma_2 = \{Person(new\_ind_{11}),\ hasPersonName(new\_ind_{11},\ pName_2),$ $PersonName(pName_2),\ hasGender(new\_ind_{11}, gen_2),\ Gender(gen_2)\}$
- $\Gamma_3 = \{Person(new\_ind_{12}),\ hasPersonName(new\_ind_{12},\ pName_2),$ $PersonName(pName_2),\ hasGender(new\_ind_{12}, gen_2),\ Gender(gen_2)\}$

Both explanations are consistent w.r.t $\mathcal{A}$ and $\mathcal{T}$. However, the explanation Γ_2 reuses the individual new_ind_{11} that exists in the ABox $\mathcal{A}$, whereas the explanation Γ_3 hypothesizes a new individual named new_ind_{12}. According to the preference score we have defined in Section 3.3.3, Γ_2 has a better preference score than Γ_3: $S_a(\mathcal{T}, \mathcal{A}, \Gamma_2)$=4, $S_h(\mathcal{T}, \mathcal{A}, \Gamma_2)$=1, and $S(\mathcal{T}, \mathcal{A}, \Gamma_2)$=4-1=3, whereas $S_a(\mathcal{T}, \mathcal{A}, \Gamma_3)$=2, $S_h(\mathcal{T}, \mathcal{A}, \Gamma_3)$=3, and $S(\mathcal{T}, \mathcal{A}, \Gamma_3)$=2-3=-1.

Since RacerPro's *retrieve-with-explanation* implements the function *select-preferred-explanations* (see Algorithm 7 in Section 3.3.3), it returns a single explanation, which contains the set of non-entailed assertions from Γ_2:

$\Delta_2 = \{hasGender(new\_ind_{11}, gen_2)\}$

According to the interpretation algorithm the assertions from Δ_2 are added to $\mathcal{A}$: $\mathcal{A}' := \mathcal{A} \cup \Delta_2$. Figure 4.18 depicts the ABox $\mathcal{A}'$ at this stage. The *execute-or-rexecute-all-rules* function call delivers again an empty set C, and the algorithm *interpret* is called again.

$CityName\ (ciName_2)$
$CountryName\ (coName_2)$
$PersonName\ (pName_2)$
$Gender\ (gen_2)$
$HighJumpName\ (hjName_2)$
$StadiumName\ (stName_2)$
$StadiumName\ (stName_3)$
$hasValue\ (ciName_2, \text{'Bastad'})$
$hasValue\ (coName_2, \text{'Sweden'})$
$hasValue\ (pName_2, \text{'Blanka Vlasic'})$
$hasValue\ (hjName_2, \text{'High Jump'})$
$hasValue\ (stName_2, \text{'Swedish Wimbledon'})$
$hasValue\ (stName_3, \text{'Bastad Tennis Stadium'})$
$personNameToGender\ (pName_2, gen_2)$
$spoNameToStaName\ (hjName_2, stName_2)$
$spoNameToStaName\ (hjName_2, stName_3)$
$Person\ (new\_ind_{11})$
$hasPersonName(new\_ind_{11},\ pName_2)$
$hasGender(new\_ind_{11}, gen_2)$

Figure 4.18: The ABox $\mathcal{A}'$ after the addition of the explanation Δ_2

In the new *interpret* function call, the set *Fiats* contains the following assertions:

$Fiats = \{(spoNameToStaName(hjName_2, stName_2), 0),$
$(spoNameToStaName(hjName_2, stName_3), 0)\}$

Assume that the assertion $spoNameToStaName(hjName_2, stName_2)$ is selected first. That assertion is removed from the set *Fiats*, and then transformed into the following query:

```
(retrieve-with-explanation ()
(hjName2 stName2 spoNameToStaName) (:reuse-old))
```

There exists three rules in $\mathcal{R}_{tex}$ with the corresponding predicate in the rule head, namely the rules at lines 3, 4, and 5 in Figure 4.17. Therefore three explanations can be generated:

- $\Gamma_4 = \{SportsCompetition(new\_ind_{13}),$
  $hasSportsName(new\_ind_{13}, hjName_2),\ SportsName(hjName_2),$
  $takesPlaceIn(new\_ind_{13}, stName_2),\ StadiumName(stName_2)\}$

- $\Gamma_5 = \{HighJumpCompetition(new\_ind_{14}),$
  $hasSportsName(new\_ind_{14}, hjName_2),\ HighJumpName(hjName_2),$
  $takesPlaceIn(new\_ind_{14}, stName_2),\ StadiumName(stName_2)\}$

- $\Gamma_6 = \{PoleVaultCompetition(new\_ind_{15}),$
  $hasSportsName(new\_ind_{15}, hjName_2),\ PoleVaultName(hjName_2),$
  $takesPlaceIn(new\_ind_{15}, stName_2),\ StadiumName(stName_2)\}$

However, the explanation Γ_6 is inconsistent, meaning that if it would be added to $\mathcal{A}'$ depicted in Figure 4.18, then $\mathcal{A}'$ would become inconsistent. The reason for the inconsistency is obvious: If Γ_6 is added to $\mathcal{A}'$, the individual $hjName_2$ becomes an instance of the concepts *HighJumpName* and *PoleVaultName*. However, according to the TBox $\mathcal{T}$, these concepts are disjoint.

The other two explanations Γ_4 and Γ_5 are both consistent, and have the same preference score. The individual *hjName2* is an instance of both *SportsName* and *HighJumpName*, because according to the TBox $\mathcal{T}$, the concept *SportsName* subsumes the concept *HighJumpName*. Therefore, the preference scores of Γ_4 and Γ_5 are the same: $S(\mathcal{T}, \mathcal{A}, \Gamma_4)$=2-3=-1, $S(\mathcal{T}, \mathcal{A}, \Gamma_5)$=2-3=-1.

As discussed in Section 3.3.3 in detail, if there exists multiple explanations with the same highest preference score, then these explanations should be checked for relaxed ABox entailment, and the most-specific explanation should be preferred. As mentioned earlier, RacerPro's *retrieve-with-explanation* function not only calculates and compares preference scores of explanations, but also checks them for relaxed ABox entailment. Therefore, in our example, the *retrieve-with-explanation* function returns only Δ_5, the set of non-entailed assertions from Γ_5, because it holds that $\Gamma_5 \rightsquigarrow \Gamma_4$. In other words, Γ_5 is the most-specific, and hence preferred, explanation for the query.

Following the interpretation algorithm, the assertions from Δ_5 are added to $\mathcal{A}$: $\mathcal{A}'$:= $\mathcal{A} \cup \Delta_5$. The *execute-or-reexecute-all-rules* function call delivers again an empty set C, since there are no rules to be considered when applying rules in a forward-chaining way. Since $\mathcal{A}' \cup C$ is consistent w.r.t. $\mathcal{T}$, the algorithm *interpret* is called recursively.

In the new *interpret* function call, the set *Fiats* contains a single assertion, namely the assertion $spoNameToStaName(hjName_2, stName_3)$, which is selected as the next fiat assertion. The assertion is removed from the set *Fiats*, and transformed into the following query:

```
(retrieve-with-explanation ()
   (hjName2 stName3 spoNameToStaName) (:reuse-old))
```

Also for this query, the rules at lines 3, 4, and 5 in Figure 4.15 can be used to compute explanations. The explanation generated using the rule at line 5 is again inconsistent for the same reason, and the explanation generated using the rule at line 4 is again more specific than the explanation generated using the rule at line 3. However, the call of the *retrieve-with-explanation* function together with the *reuse-old* parameter results in the computation of two different explanations by using the rule at line 4 in Figure 4.15:

- $\Gamma_7 = \{HighJumpCompetition(new\_ind_{14}),$
  $hasSportsName(new\_ind_{14}, hjName_2),\ HighJumpName(hjName_2),$
  $takesPlaceIn(new\_ind_{14}, stName_3),\ StadiumName(stName_3)\}$

- $\Gamma_8 = \{HighJumpCompetition(new\_ind_{16}),$
  $hasSportsName(new\_ind_{16}, hjName_2),\ HighJumpName(hjName_2),$
  $takesPlaceIn(new\_ind_{16}, stName_3),\ StadiumName(stName_3)\}$

The explanation Γ_7 reuses the existing *HighJumpCompetition* instance new_ind_{14}, whereas Γ_8 hypothesizes a new individual named new_ind_{16} as part of the explanation. Consequently, the explanation Γ_7 has the highest preference score among all explanations, and at the same time is the most-specific explanation. The *retrieve-with-explanation* function returns only Δ_7, the set of non-entailed assertions from Γ_7.

Following the interpretation algorithm, the assertions from Δ_7 are added to $\mathcal{A}$: $\mathcal{A}'$:= $\mathcal{A} \cup \Delta_7$. The *execute-or-reexecute-all-rules* function call delivers again an empty set

C. Since $\mathcal{A}' \cup C$ is consistent w.r.t. $\mathcal{T}$, the algorithm *interpret* is called recursively, this time with an empty set of fiat assertions. Consequently, the new *interpret* function call returns an ABox as result, which is accumulated in the set $\mathcal{I}$.

Finally, the *interpret* algorithm terminates by returning the set $\mathcal{I}$ containing the ABox depicted in Figure 4.19. As a result, the semantic interpretation engine answers the client's call of the *interpretText* function with the ABox shown in Figure 4.19. In the answer, the interpretation ABox is given the unique name *textABox01_interpretation1*.

$CityName\ (ciName_2)$
$CountryName\ (coName_2)$
$PersonName\ (pName_2)$
$Gender\ (gen_2)$
$HighJumpName\ (hjName_2)$
$StadiumName\ (stName_2)$
$StadiumName\ (stName_3)$
$hasValue\ (ciName_2, \text{'Bastad'})$
$hasValue\ (coName_2, \text{'Sweden'})$
$hasValue\ (pName_2, \text{'Blanka Vlasic'})$
$hasValue\ (hjName_2, \text{'High Jump'})$
$hasValue\ (stName_2, \text{'Swedish Wimbledon'})$
$hasValue\ (stName_3, \text{'Bastad Tennis Stadium'})$
$personNameToGender\ (pName_2, gen_2)$
$spoNameToStaName\ (hjName_2, stName_2)$
$spoNameToStaName\ (hjName_2, stName_3)$
$Person\ (new\_ind_{11})$
$hasPersonName(new\_ind_{11}, pName_2)$
$hasGender(new\_ind_{11}, gen_2)$
$HighJumpCompetition\ (new\_ind_{14})$
$hasSportsName(new\_ind_{14}, hjName_2)$
$takesPlaceIn(new\_ind_{14}, stName_2)$
$takesPlaceIn(new\_ind_{14}, stName_3)$

Figure 4.19: The interpretation ABox *textABox01_interpretation1* returned by the semantic interpretation engine

Informally speaking, the interpretation of the text segment in Figure 4.14 states that a high jump competition is taking place in two stadiums. Notice that in the TBox $\mathcal{T}$ shown in Figure 4.16, there are no number restrictions on the *takesPlaceIn* relation between the concepts *SportsCompetition* and *StadiumName*, and thus the interpretation ABox in Figure 4.19 is consistent. If we read the text in Figure 4.14 carefully, we can understand that the Bastad tennis stadium, where the high jump competition took place, is nicknamed the 'Swedish Wimbledon'. However, the lack of this information in the text analysis ABox in Figure 4.15 indicates that the employed text analysis tool could not identify the synonymy of these two stadium names. As a solution, more sophisticated text analysis techniques can be applied to improve text analysis results. In our hybrid multimedia interpretation approach, the employed text analysis tool can easily be replaced by any other text analysis tool that can provide for an analysis ABox as output. Furthermore, despite the replacement of the text analysis tool employed in the framework, the semantic interpretation engine can be used without any modifications.

4.3.2 Strategies for the Interpretation Process

In the previous section, the segments of a sample web page have been interpreted in the following order: first image, then caption, and finally text. As mentioned before, the interpretation of a segment is independent from other segments, and hence the segments can be interpreted in any order. The only restriction is that all segments of a multimedia document have to be interpreted before the fusion step.

Different from segments, fiats assertions cannot be processed in an arbitrary order. In order to obtain 'preferred' explanations of observations using our logic-based interpretation approach, fiat concept assertions have to be interpreted before fiat role assertions. In addition, the strategy parameter Ω of the abductive retrieval service (see Section 3.3.2) has to be selected carefully such that during explanation generation existing individuals from the knowledge base are reused only in certain cases. If RacerPro is used for abduction, the reasoner can be instructed to reuse existing individuals by calling the *retrieve-with-explanation* function with the parameter *reuse-old*. In this section we investigate appropriate strategy parameter values and the appropriate order for explaining fiat assertions during the interpretation process on the basis of examples. This will provide for additional motivation why the *select-fiats* function presented in

Section 3.4 employs a particular strategy when selecting the next fiat assertion during the interpretation process.

As discussed in Section 3.2, the logic-based interpretation approach presented in this work exploits a knowledge representation formalism where DLs are extended with rules. The use of Datalog-like rules to extend DLs guarantees to preserve decidability, however introduces some limitations, which has to be considered during explanation generation. In fact, a Datalog-like rule contains at most one positive literal, and hence only one atom in the head. For the time being, RacerPro's *retrieve-with-explanation* can only be called to explain a single observation, which is a concept or a role assertion. However, this does not mean that the observation to be explained is considered in complete isolation. In fact, all other observations exist in the ABox part of the background knowledge, and thus, affect the preference scores of explanations (see Equation 3.6 in Section 3.3.3). Therefore, 'preferred' explanations can be obtained only by appropriate configuration of the *retrieve-with-explanation* parameters.

To explain the role of the *retrieve-with-explanation* function's *reuse-old* parameter, in the following, we present a simple interpretation example. Assume that the text analysis ABox *sampleABox1* in Figure 4.20 has to be interpreted.

$PersonName\ (pName_3)$
$Age\ (age_1)$
$Gender\ (gen_3)$
$hasValue\ (pName_3, \text{‘}Blanka\ Vlasic\text{’})$
$personNameToGender\ (pName_3, gen_3)$
$personNameToAge\ (pName_3, age_1)$

Figure 4.20: The ABox *sampleABox1*

Assume that the background knowledge consists of the TBox in Figure 4.21 and the set of rules in Figure 4.22.

Figure 4.23 illustrates two interpretation ABoxes for the analysis ABox *sampleABox1* as graphs. The ABox on the left-hand side is the interpretation ABox that is preferred and hence should be computed by the semantic interpretation engine, whereas the ABox on the right-hand side is an interpretation result that should be omitted.

$$
\begin{array}{rcl}
DLC & \sqsubseteq & \neg SLC \\
Person & \sqsubseteq & DLC \sqcap \exists_{\leq 1} hasGender.Gender \sqcap \exists_{\leq 1} hasAge.Age \sqcap \\
 & & \exists\, hasPersonName.PersonName \\
Name & \sqsubseteq & SLC \sqcap \exists\, hasValue.string \\
PersonName & \sqsubseteq & Name \\
Gender & \sqsubseteq & SLC \sqcap \neg Name \\
Age & \sqsubseteq & SLC \sqcap \neg Name
\end{array}
$$

Figure 4.21: A sample TBox $\mathcal{T}$

$$
\begin{array}{lcl}
personNameToGender(Y, Z) & \leftarrow & Person(X), hasPersonName(X, Y), \\
 & & PersonName(Y), hasGender(X, Z), \\
 & & Gender(Z) \\
personNameToAge(Y, Z) & \leftarrow & Person(X), hasPersonName(X, Y), \\
 & & PersonName(Y), hasAge(X, Z), \\
 & & Age(Z)
\end{array}
$$

Figure 4.22: A set of text interpretation rules $\mathcal{R}_1$

Informally speaking, the interpretation on the right-hand side in Figure 4.23 is not preferred due to its vagueness. It includes two hypothesized *Person* instances, namely new_ind_1 and new_ind_2, that may or may not refer to the same person in the text segment.

With respect to the set of text interpretation rules $\mathcal{R}_1$ and the analysis ABox *sampleABox1*, the set of fiat assertions is as follows:

$Fiats = \{(personNameToGender(pName_3, gen_3), 0),$
$(personNameToAge(pName_3, age_1), 0)\}$

Assume that the assertion $personNameToGender(pName_3, gen_3)$ is selected first. The *retrieve-with-explanation* function call to explain this assertion is as follows:

```
(retrieve-with-explanation ()
   (pName3 gen3 personNameToGender) (:reuse-old))
```

The answer to the query contains a single explanation, namely:

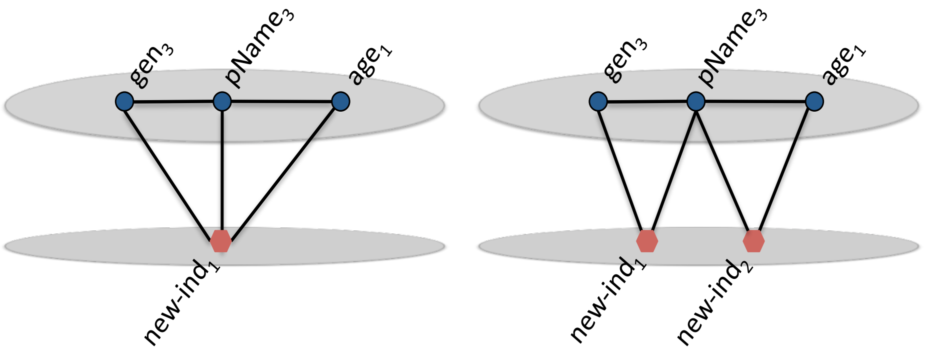


Figure 4.23: Two possible interpretation results for the same analysis ABox *sampleABox1*, where the one on the left-hand side is preferred

$\Delta_1 = \{Person(new\_ind_1),\ hasPersonName(new\_ind_1, pName_3),$
$hasGender(new\_ind_1, gen_3)\}$

For this query, the use or the exclusion of the *reuse_old* parameter makes no difference, because there exists no person instance in the ABox that can be reused. Following the interpretation algorithm, all assertions from the explanation Δ_1 are added to the ABox *sampleABox1*: $\mathcal{A}' = sampleABox1 \cup \Delta_1$.

The parameter *reuse_old* plays a crucial role for obtaining the preferred explanation of the fiat assertion $personNameToAge(pName_3, age_1)$. If the query is posed without the *reuse_old* parameter:

```
(retrieve-with-explanation ()
   (pName3 age1 personNameToAge))
```

only the following explanation is generated:

$\Delta_2 = \{Person(new\_ind_2),\ hasPersonName(new\_ind_2, pName_3),$
$hasAge(new\_ind_2, age_1)\}$

In this case, the final interpretation result corresponds to the interpretation ABox shown on the right-hand side graph in Figure 4.23, which is not preferred. However, if the same query is posed with the parameter *reuse_old*:

```
(retrieve-with-explanation ()
   (pName3 age1 personNameToAge) (:reuse-old))
```

there are two possible explanations:

- $\Gamma_3 = \{Person(new\_ind_1),\ hasPersonName(new\_ind_1, pName_3),$
  $PersonName(pName_3),\ hasAge(new\_ind_1, age_1),\ Age(age_1)\}$

- $\Gamma_4 = \{Person(new\_ind_2),\ hasPersonName(new\_ind_2, pName_3),$
  $PersonName(pName_3),\ hasAge(new\_ind_2, age_1),\ Age(age_1)\}$

with the preference scores $S(\mathcal{T}_1,\ \mathcal{A}',\ \Gamma_3)$= 4-1=3 and $S(\mathcal{T}_1,\ \mathcal{A}',\ \Gamma_4)$=2-3=-1. Consequently, Γ_3 is the preferred explanation. The *retrieve-with-explanation* function call returns the set of non-entailed assertions from Γ_3 as result, which are added to $\mathcal{A}'$ following the interpretation algorithm. By using the *reuse_old* parameter, we obtain the same interpretation result as the one depicted on the left-hand side graph in Figure 4.23, which is the preferred interpretation result.

This example shows that, in the logic-based interpretation approach, fiat role assertions have to be explained always with respect to existing individuals in the ABox, in order to obtain preferred explanations only.

In the next example, we explore the appropriate abduction strategy for explaining fiat concept assertions. Assume that the text analysis ABox *sampleABox2* in Figure 4.24 has to be interpreted. In addition, the background knowledge consists of the TBox $\mathcal{T}$ in Figure 4.21 and the set of rules $\mathcal{R}_2$ in Figure 4.25.

$PersonName\ (pName_4)$
$PersonName\ (pName_5)$
$hasValue\ (pName_4, \text{‘}Blanka\ Vlasic\text{’})$
$hasValue\ (pName_5, \text{‘}Yelena\ Slesarenko\text{’})$

Figure 4.24: The ABox *sampleABox2*

$$PersonName(Y) \quad \leftarrow \quad Person(X), hasPersonName(X, Y)$$

Figure 4.25: A set of text interpretation rules $\mathcal{R}_2$ containing a single rule

In this case, the set of fiat assertions contains both concept assertions from the ABox *sampleABox2*:

$\mathit{Fiats} = \{(\mathit{PersonName}(\mathit{pName_4}), 0), (\mathit{PersonName}(\mathit{pName_5}), 0)\}$

Assume that $\mathit{PersonName}(\mathit{pName_4})$ is the first assertion selected for abduction. The function call to explain this concept assertion is as follows:

```
(retrieve-with-explanation ()
   (pName4 PersonName))
```

where the use of the parameter *reuse-old* would not have any effect on the query answer, since, at this moment, there exists no *Person* instance in the ABox that could be reused in explanation generation. The only explanation is generated by hypothesizing a *Person* instance:

$\Delta_1 = \{\mathit{Person}(\mathit{new\_ind_1}), \mathit{hasPersonName}(\mathit{new\_ind_1}, \mathit{pName_4})\}$

After the addition of the assertions from Δ_1 to the ABox *sampleABox2*: $\mathcal{A}' =$ *sampleABox2* $\cup\ \Delta_1$, the last fiat assertion, namely $\mathit{PersonName}(\mathit{pName_5})$, is transformed into a Boolean query:

```
(retrieve-with-explanation ()
   (pName5 PersonName))
```

Since the *retrieve-with-explanation* function is called without the *reuse-old* parameter, only a single explanation is generated by hypothesizing another *Person* instance:

$\Delta_2 = \{\mathit{Person}(\mathit{new\_ind_2}), \mathit{hasPersonName}(\mathit{new\_ind_2}, \mathit{pName_5})\}$

According to the interpretation algorithm, all assertions from Δ_2 are added to $\mathcal{A}'$ to obtain the interpretation ABox. The interpretation ABox is depicted on the left-hand side of Figure 4.26.

Informally speaking, the graph on the left-hand side in Figure 4.26 illustrates the situation where two person names are considered as evidences for the existence of two persons. This interpretation corresponds to what most users would agree with intuitively. Therefore the interpretation result illustrated by the left-hand side graph is the preferred interpretation of the analysis ABox *sampleABox2*.

Notice that one would prefer the interpretation on the right-hand side graph in Figure 4.26, if both person names, namely $pName_4$ and $pName_5$, refer to the same

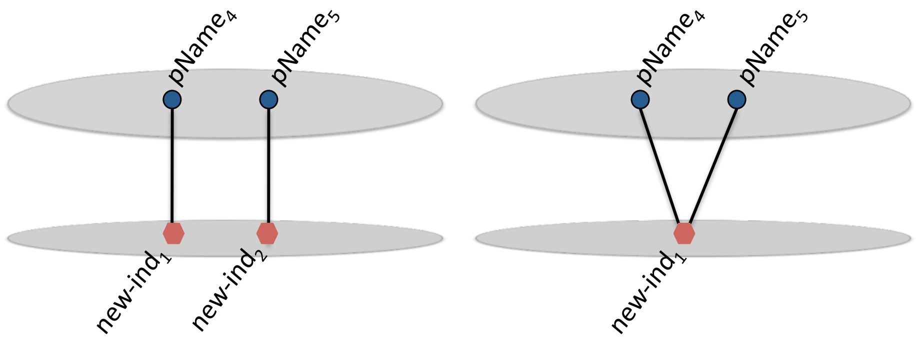


Figure 4.26: Two different interpretation results for the analysis ABox *sampleABox2*, where the one on the left-hand side is preferred

person. In our framework, this information can only be extracted by text analysis tools. Therefore, if the analysis ABox contains no additional information about two person names referring to the same person, the interpretation engine has to interpret this analysis ABox under the assumption that different person names refer to different persons, and return the interpretation result shown in the left-hand side graph.

If the *retrieve-with-explanation* function is called with the *reuse-old* parameter for the last fiat concept assertion:

```
(retrieve-with-explanation ()
   (pName5 PersonName) (:reuse-old))
```

two explanations are generated:

- $\Gamma_3 = \{Person(new\_ind_1), hasPersonName(new\_ind_1, pName_5)\}$
- $\Gamma_4 = \{Person(new\_ind_2), hasPersonName(new\_ind_2, pName_5)\}$

where, informally speaking, Γ_3 reuses the individual *new_ind*$_1$, and Γ_4 hypothesizes a *Person* instance, namely *new_ind*$_2$. The preference scores of the explanations are as follows: $S(\mathfrak{T}_1, \mathcal{A}', \Gamma_3)$= 1-1=0 and $S(\mathfrak{T}_1, \mathcal{A}', \Gamma_4)$=0-2=-2. Since the preference score of Γ_3 is higher, the *retrieve-with-explanation* function call returns the set of non-entailed assertions from Γ_3, namely $\Delta_1 = \{hasPersonName(new\_ind_1, pName_5)\}$. After the addition of Δ_1 to $\mathcal{A}'$, the interpretation depicted on the right-hand side of Figure 4.26 is obtained, which is not preferred. This simple example shows that fiat concept assertions should always be explained without the reuse of existing individuals in the ABox.

After the identification of appropriate strategies regarding the reuse of individuals, we continue with the identification of the appropriate order for the processing of fiat concept and role assertions. To this end, we use a very simple text analysis ABox as an example.

Assume that the text analysis ABox *sampleABox3* in Figure 4.27 has to be interpreted. The background knowledge consists of the TBox $\mathcal{T}$ in Figure 4.21 and the set of rules $\mathcal{R}_3$ in Figure 4.28.

$$\begin{array}{l}
PersonName\ (pName_6) \\
Gender\ (gen_4) \\
hasValue\ (pName_6, \text{‘}Blanka\ Vlasic\text{’}) \\
personNameToGender\ (pName_6, gen_4)
\end{array}$$

Figure 4.27: The sample analysis ABox *sampleABox3*

$$\begin{array}{lcl}
PersonName(Y) & \leftarrow & Person(X), hasPersonName(X,Y) \\
personNameToGender(Y,Z) & \leftarrow & Person(X), hasPersonName(X,Y), \\
 & & PersonName(Y), hasGender(X,Z), \\
 & & Gender(Z)
\end{array}$$

Figure 4.28: A set of text interpretation rules $\mathcal{R}_3$

Initially, the set *Fiats* contains following assertions:

$Fiats = \{(PersonName(pName_6),\, 0),\, (personNameToGender(pName_6, gen_4),\, 0)\}$

Assume that the assertion $personNameToGender(pName_6, gen_4)$ is selected first. The function call to explain this assertion is:

```
(retrieve-with-explanation ()
   (pName6 gen4 personNameToGender) (:reuse-old))
```

For this query only the following explanation is generated:

$\Delta_1 = \{Person(new\_ind_1),\ hasPersonName(new\_ind_1, pName_6),$
$hasGender(new\_ind_1, gen_4)\}$

Subsequently, all assertions from Δ_1 are added to the ABox *sampleABox3*: $\mathcal{A}' =$ *sampleABox3* $\cup\ \Delta_1$. In the next step, the assertion *PersonName*(*pName$_4$*) is processed:

```
(retrieve-with-explanation ()
   (pName6 PersonName))
```

As discussed earlier, for fiat concept assertions that are transformed into a Boolean query, the *retrieve-with-explanation* function is called without the parameter *reuse-old*. Therefore existing individuals from the ABox *sampleABox3* are not considered for reuse in the generated explanation:

$\Delta_2 = \{Person(new\_ind_2),\ hasPersonName(new\_ind_2, pName_6)\}$

The interpretation result for this simple example is obtained by adding all assertions from Δ_2 to $\mathcal{A}'$. The interpretation ABox is depicted on the right-hand side graph in Figure 4.29. Informally speaking, the *PersonName* instance and the *Gender* instance are considered to belong to two *Person* instances that may or may not be the same.

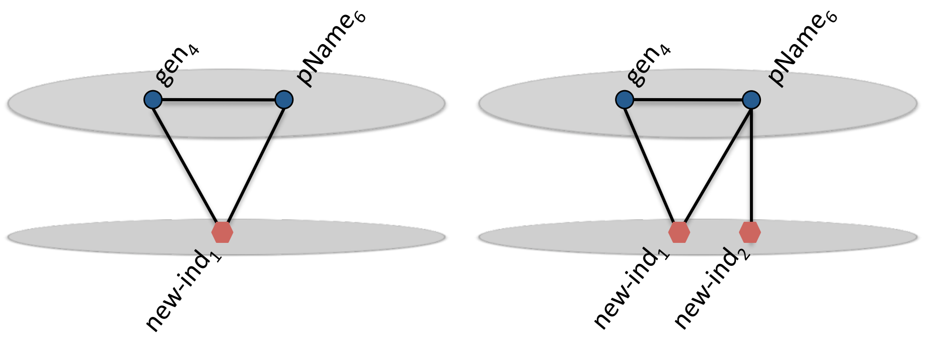


Figure 4.29: Two different interpretation results for the analysis ABox *sampleABox3*, where the one on the left-hand side is preferred

Figure 4.29 illustrates two possible interpretation results for the analysis ABox *sampleABox3*. The graph on the left-hand side depicts another interpretation where the person name and the gender are interpreted as belonging to the same person since a relation between the person name and gender has been identified by text analysis tools. Obviously, the interpretation result on right-hand side graph is vague compared to the interpretation result on the left-hand side graph. Therefore the interpretation result shown on the left-hand side is the more intuitive one, and thus should be the preferred interpretation for this example.

Figure 4.29 also shows that we could not obtain the preferred interpretation result by calling fiat role assertions first, and fiat concept assertions later. We continue our discussion by analyzing the situation in which fiat concept assertions are explained first.

To interpret the ABox *sampleABox3* in Figure 4.27, this time we start with the fiat concept assertion $PersonName(pName_6)$:

```
(retrieve-with-explanation ()
   (pName6 PersonName))
```

and retrieve the following explanation:

$\Delta_1 = \{Person(new\_ind_1),\ hasPersonName(new\_ind_1, pName_6)\}$

After the addition of all assertions from Δ_1 to the ABox *sampleABox3*: $\mathcal{A}' = sampleABox3 \cup \Delta_1$, the fiat role assertion $personNameToGender(pName_6, gen_4)$ is transformed into a Boolean query:

```
(retrieve-with-explanation ()
   (pName6 gen4 personNameToGender) (:reuse-old))
```

There exist two possible explanations for the query, since the *retrieve-with-explanation* function is instructed to consider also explanations where existing individuals from $\mathcal{A}'$ are reused.

- $\Gamma_2 = \{Person(new\_ind_1),\ hasPersonName(new\_ind_1, pName_6),$ $PersonName(pName_6),\ hasGender(new\_ind_1, gen_4),\ Gender(gen_4)\}$
- $\Gamma_3 = \{Person(new\_ind_2),\ hasPersonName(new\_ind_2, pName_6),$ $PersonName(pName_6),\ hasGender(new\_ind_2, gen_4),\ Gender(gen_4)\}$

Obviously, the explanation Γ_2 has a higher preference score than the explanation Γ_3, because it hypothesizes less assertions and both explanations have the same number of assertions. More precisely, the preference scores of the explanations are as follows:

$S(\mathcal{T}, \mathcal{A}', \Gamma_2)$= 4-1=3 and $S(\mathcal{T}, \mathcal{A}', \Gamma_3)$=2-3=-1. Therefore, the *retrieve-with-explanation* function returns only the set of non-entailed assertions from Γ_2 as result. Following the interpretation algorithm, the set of non-entailed assertions from Γ_2 is added to $\mathcal{A}'$ to obtain the interpretation result. In this case, the interpretation result we obtain is the preferred interpretation of the analysis ABox, and corresponds to the left-hand side graph in Figure 4.29.

To recapitulate, we have investigated the appropriate abduction strategy parameter Ω for the reuse or the exclusion of existing individuals from the knowledge base during the interpretation process. Using practical examples, we have shown that in order to obtain preferred interpretations, fiat role assertions should always be explained by also considering the reuse of existing individuals (Ω=*reuse existing individuals*), whereas fiat concept assertions should always be explained by considering the hypothesization of new individuals only (Ω=*use new individuals*).

Furthermore, we have investigated the appropriate order for the processing of fiat concept and role assertions. Based on examples, we have shown that fiat concept assertions have to be processed before fiat role assertions in order to obtain preferred interpretations.

The current implementation of the semantic interpretation engine employs the *select-fiats* function presented in Section 3.4, and thus computes interpretations following the strategies we have dicussed in this section. Given a set of fiat assertions with the same level, the semantic interpretation engine processes fiat concept assertions first, and then fiat role assertions. In addition, the semantic interpretation engine instructs RacerPro to follow the above-mentioned individual reuse strategy through the appropriate parameterization of the *retrieve-with-explanation* function.

4.3.3 Fusion

In Section 4.3.1, we discussed modality-specific interpretation of various segments of a sample web page, where the semantic interpretation engine provides for the essential machinery to compute interpretations. Besides the interpretation algorithm, the semantic interpretation engine implements also the fusion algorithm presented in Section 3.5. In this section, we present the fusion of modality-specific interpretations based on examples.

As discussed in Section 3.5, the goal of fusion is not only to merge modality-specific interpretation ABoxes into a fused interpretation ABox, but also to achieve an information gain through the identification of individuals in modality-specific interpretation ABoxes that describe the same real-world entity. After the addition of same-as assertions, which represent the *sameness* of individuals in different modality-specific interpretation ABoxes, certain fusion alternatives can be ruled out due to inconsistencies. Since a web page segment may have multiple interpretations, there exists multiple fusion alternatives, which cause the generation of several fused interpretation ABoxes for the same multimedia document. However, practical application scenarios require the generation of a small number of fused interpretation ABoxes for a multimedia document.

In order to reduce the number of fused interpretations, we enhance the background knowledge used by the semantic interpretation engine with a set of additional axioms. Figure 4.30 shows an excerpt of these axioms, which state background knowledge about the athletics domain. In the BOEMIE project these axioms have automatically been generated from an athletics database. Henceforth, we refer to the enhanced background knowledge as $\mathcal{T}_{enc}$.

$$\begin{aligned}
Person \sqcap \exists hasPersonName.\exists hasValue.\text{‘}Blanka\ Vlasic\text{’} &\sqsubseteq HighJumper\\
Person \sqcap \exists hasPersonName.\exists hasValue.\text{‘}Yelena\ Slesarenko' &\sqsubseteq HighJumper\\
Person \sqcap \exists hasPersonName.\exists hasValue.\text{‘}Yelena\ Isinbayeva' &\sqsubseteq PoleVaulter\\
Person \sqcap \exists hasPersonName.\exists hasValue.\text{‘}Stacy\ Dragila' &\sqsubseteq PoleVaulter
\end{aligned}$$

Figure 4.30: An excerpt of the axioms, which are added to the background knowledge $\mathcal{T}$

To show the role of the additional axioms in making certain fusion alternatives to become inconsistent, we give a simple example from the athletics domain. Assume a web page with an image and its caption. There exists two interpretations for the image segment, one containing a pole vaulter and the other one a high jumper. For the caption, there exists only one interpretation, which contains a person instance named ‘Yelena Slesarenko’. In this case, there exists two fusion alternatives, since the image segment has two interpretations. In one fusion alternative, the pole vaulter from the image interpretation and the person from the caption interpretation, and in the other fusion alternative, the high jumper from the other image interpretation and the person from the caption interpretation have been identified as describing the same real-world entity.[1]

After the addition of the corresponding same-as assertions, one of the fusion alternatives contains a pole vaulter named 'Yelena Slesarenko', whereas the other fusion alternative contains a high jumper named 'Yelena Slesarenko'. Notice that according to $\mathcal{T}_{enc}$ 'Yelena Slesarenko' is a high jumper (see Figure 4.30). Furthermore, $\mathcal{T}_{enc}$ contains also the following disjointness axiom:

$$HighJumper \sqcap PoleVaulter \sqsubseteq \perp$$

Consequently, the fused interpretation ABox that contains a pole vaulter named 'Yelena Slesarenko':

$PoleVault\ (new\_ind_{18})$
$PoleVaulter\ (new\_ind_{19})$
$hasParticipant(new\_ind_{18},\ pName_{19})$
$hasPart(new\_ind_{18},\ pole_5)$
$Pole\ (pole_5)$
$hasPart(new\_ind_{19}, face_3)$
$hasPart(new\_ind_{19}, body_3)$
$PersonFace\ (face_3)$
$PersonBody\ (body_3)$
$adjacent\ (face_3, body_3)$
$adjacent\ (body_3, pole_5)$
$Person\ (new\_ind_{20})$
$hasPersonName(new\_ind_{20},\ pName_{11})$
$PersonName\ (pName_{11})$
$hasValue\ (pName_{11},\ \text{'Yelena Slesarenko'})$
$same\text{-}as\ (new\_ind_{19}, new\_ind_{20})$

is inconsistent w.r.t. $\mathcal{T}_{enc}$. Therefore, the semantic interpretation engine discards this fusion alternative, and returns the fused interpretation ABox that contains a high jumper named 'Yelena Slesarenko' as the only fusion result. This simple example shows how the fusion algorithm disambiguates modality-specific interpretation results with respect to the background knowledge, and hence reduces the number of fused interpretation ABoxes.

[1]The queries used to identify the same-as candidates in web pages with athletics news will be presented later in this section.

It should be noted here that the modality-specific interpretation ABoxes generated by the semantic interpretation engine contain not only information about the athletics domain but also information about the segment from which information has been extracted and interpreted. Furthermore, every interpretation ABox generated by the semantic interpretation engine contains role assertions between the segment instance and DLC instances, as well as between the segment instance and SLC instances. In the interpretation ABoxes shown so far, e.g. the caption interpretation ABox *captionABox01_interpretation1* in Figure 4.13, these assertions are omitted for the sake of brevity. However, for the current discussion these assertions are indispensable, since they are exploited during fusion.

Figure 4.31 depicts the complete interpretation ABox *captionABox01_interpretation1* for the caption in Figure 4.9 including all assertions (compare to Figure 4.13). Similarly, all image interpretation ABoxes contain a unique *Image* instance, and all text interpretation ABoxes a unique *Text* instance. As shown in Figure 4.31, the role *depicts* is used to explicitly represent a relation between the segment and all DLC instances that have been hypothesized during the interpretation step, whereas the role *depictsSLC* is used to represent a relation between the segment and all SLC instances. Concepts such as *Caption*, *Image*, *Text* and roles such as *depicts* and *depictsSLC* are related to structural aspects of multimedia content. Therefore, in the BOEMIE project, these concepts and roles have been defined in the multimedia content ontology MCO.

We proceed with the fusion of modality-specific interpretations of the web page segments from Section 4.3.1. For the image in Figure 4.3, two interpretation ABoxes shown in Figure 4.8, namely *imageABox01_interpretation1* and *imageABox01_interpretation2*, are computed. For the text in Figure 4.9 that captions the image, and the text segment in Figure 4.14, the interpretation ABoxes *captionABox01_interpretation1* and *text01_interpretation1* are computed, respectively.

Notice that UNA is dropped for fusion, and all interpretation ABoxes generated by the semantic interpretation have unique names. Interpretation ABoxes with unique names can be identified and referenced unambiguously, which is essential for fusion. In this example there exist two fusion alternatives, since the image segment has two interpretations.

Following the fusion algorithm presented in Section 3.5 (see Algorithm 10), for each fusion alternative, i.e. the cross product of the sets of modality-specific interpretation

$Caption\ (cap_1)$
$depicts\ (cap_1, new\_ind_9)$
$depicts\ (cap_1, new\_ind_{10})$
$depictsSLC\ (cap_1, pName_1)$
$depictsSLC\ (cap_1, perf_1)$
$depictsSLC\ (cap_1, ciName_1)$
$depictsSLC\ (cap_1, coName_1)$
$PersonName\ (pName_1)$
$Performance\ (perf_1)$
$CityName\ (ciName_1)$
$CountryName\ (coName_1)$
$personNameToPerformance\ (pName_1, perf_1)$
$hasValue\ (pName_1, \text{‘}\mathit{Blanka\ Vlasic}\text{’})$
$hasValue\ (perf_1, \text{‘2.02'})$
$hasValue\ (ciName_1, \text{‘}\mathit{Bastad}\text{’})$
$hasValue\ (coName_1, \text{‘}\mathit{Sweden}\text{’})$
$Person\ (new\_ind_9)$
$hasPersonName\ (new\_ind_9, pName_1)$
$personToPerformance\ (new\_ind_6, perf_1)$
$SportsTrial\ (new\_ind_{10})$
$hasPerformance\ (new\_ind_{10}, perf_1)$
$hasParticipant\ (new\_ind_{10}, new\_ind_9)$
$Athlete\ (new\_ind_9)$

Figure 4.31: All assertions of the interpretation ABox *captionABox01_interpretation1* as returned by the semantic interpretation engine

ABoxes, the corresponding modality-specific interpretation ABoxes are accumulated into an ABox $\mathcal{A}$. Next, the function *query-same-as-candidates* is called to obtain a set of pairs of individuals, which are believed to refer to the same real-world entity. To this end, the function *query-same-as-candidates* poses a set of application-dependent conjunctive queries $\mathcal{CQ}s$ to $\mathcal{T}_{enc} \cup \mathcal{A}$.

In the BOEMIE project, experiments have been conducted on web pages with athletics news in order to identify the application-dependent conditions that have to be fulfilled to consider two individuals as the same. Consequently, the set of conjunctive queries $\mathcal{CQ}s$ defined for the BOEMIE project contains three queries. In a nutshell, one of these queries returns pairs of *SportsTrial* instances, whereas the other two return pairs of *Person* instances.

More precisely, the first query returns pairs of *SportsTrial* instances where the first *SportsTrial* instance originates from the image interpretation ABox, and the second *SportsTrial* instance from the caption interpretation ABox:

```
(retrieve (?st1 ?st2)
   (and (?x Image)
        (?x ?st1 depicts)
        (?st1 SportsTrial)
        (?y Caption)
        (?y ?st2 depicts)
        (?st2 SportsTrial)))
```

The second query returns pairs of *Person* instances where the first *Person* instance originates from the image interpretation ABox, and the second *Person* instance from the caption interpretation ABox:

```
(retrieve (?p1 ?p2)
  (and (?x Image)
       (?x ?p1 depicts)
       (?p1 Person)
       (?y Caption)
       (?y ?p2 depicts)
       (?p2 Person)))
```

The third query returns pairs of *Person* instances with the same person names where the first *Person* instance originates from the caption interpretation ABox, and the second *Person* instance from the text interpretation ABox:

```
(retrieve (?p1 ?p2)
    (and (?x Caption)
         (?x ?p1 depicts)
         (?p1 Person)
         (?p1 ?pName1 hasPersonName)
         (?y Text)
(?y ?p2 depicts)
(?p2 Person)
(?p2 ?pName2 hasPersonName)
(?pName1 ?pName2 (constraint hasValue hasValue string=))))
```

Following the fusion algorithm, the set of pairs of individuals $\mathcal{C}$ returned by the *query-same-as-function* function call is used to generate the set of same-as assertions $\mathcal{SA}$. In the next step, the function *compute-consistent-fused-interpretations* is called with the enhanced background knowledge $\mathcal{T}_{enc}$, the ABox $\mathcal{A}$, and the set of same-as assertions $\mathcal{SA}$.

As discussed earlier in this section, the ABox $\mathcal{A}$ might become inconsistent after the addition of same-as assertions. The reason for the inconsistency of the fused interpretation ABox might be imperfect interpretation results that are probably caused by imperfect analysis results. In fact, experiments conducted on web pages with athletics news have shown that in many cases, image analysis results are less reliable than text analysis results. Therefore, for images with captioned text, the interpretation results from the caption segment should be preferred over the interpretation results from the image segment. For this purpose, image interpretation results can be 'relaxed' in the sense that assertions regarding specific sports trial and athlete types can be replaced with assertions representing more general sports trial and athlete types. For example, $PoleVault(i_1)$ can be replaced with $SportsTrial(i_1)$, and $PoleVaulter(i_2)$ can be replaced with $Jumper(i_2)$.

In the BOEMIE project, this insight has been exploited to develop a specific repair mechanism in case of inconsistent fused interpretation ABoxes. In order to apply the idea of 'relaxing' image interpretation results to repair inconsistencies, the application-specific *compute-consistent-fused-interpretations* function shown in Algorithm 14 has been defined, which overwrites the general *compute-consistent-fused-interpretations* function presented in Section 3.5 (see Algorithm 12).

```
Algorithm compute-consistent-fused-interpretations($\mathcal{T}_{enc}, \mathcal{A}, \mathcal{SA}$)
Output: a set of consistent fused interpretation ABoxes
if $\mathcal{T} \cup \mathcal{A} \cup \mathcal{SA} \models \perp$ then
  | return {relax-image-interpretation-results($\mathcal{T}_{enc}, \mathcal{A}, \mathcal{SA}$)}
else
  | return {$\mathcal{A} \cup \mathcal{SA}$}
end
```

Algorithm 14: The algorithm for computing consistent fused interpretations

If the fused interpretation ABox ($\mathcal{A} \cup \mathcal{SA}$) is inconsistent, the *compute-consistent-fused-interpretations* function calls the *relax-image-interpretation-results* function. The goal of the *relax-image-interpretation-results* function is to relax image interpretation results, and then to propagate caption interpretation results in order to repair the inconsistency. Later in this section, we will discuss the details of the *relax-image-interpretation-results* function based on a practical example.

Following the fusion algorithm in Algorithm 10, all consistent fused interpretation ABoxes returned by the *compute-consistent-fused-interpretations* function are accumulated in the set $\mathcal{F}$. Finally, the function *select-preferred-fused-interpretations* is called to select preferred fused interpretations from $\mathcal{F}$, which are then returned as the result. The general *select-preferred-fused-interpretations* function presented in Section 3.5 (see Algorithm 13) has not been implemented in the BOEMIE project. It has been overwritten by an application-specific *select-preferred-fused-interpretations*, which employs, due to time restrictions of the project, no strategy for preferring fused interpretations. It simply returns all consistent fused interpretation ABoxes as result.

Let us proceed with a concrete example to clarify how the preference of caption interpretation results over image interpretation results is done in practice. We assume that a client wants to obtain fused interpretations for a web page. For simplicity, the web page consists of an image and a caption segment only. For fusion, the client calls the *interpetWebPage* web service. Figure 4.32 depicts a so-called *web page analysis ABox*, which the client of the semantic interpretation engine sends as a parameter when calling the *interpetWebPage* web service.

The web page analysis ABox describes the structure of the multimedia document. In addition, it serves as a blueprint for the fusion of the interpretation results of segments into a fused interpretation ABox of the whole multimedia document. The web page

$\mathit{WebPage}\ (\mathit{wep}_1)$
$\mathit{CaptionedImage}\ (\mathit{cim}_1)$
$\mathit{Image}\ (\mathit{ima}_1)$
$\mathit{Caption}\ (\mathit{cap}_1)$
$\mathit{contains}\ (\mathit{wep}_1, \mathit{cim}_1)$
$\mathit{contains}\ (\mathit{cim}_1, \mathit{ima}_1)$
$\mathit{contains}\ (\mathit{cim}_1, \mathit{cap}_1)$
$\mathit{hasURL}\ (\mathit{wep}_1$, *'http://www.iaaf.org/.../news.html'*)
$\mathit{hasURL}\ (\mathit{ima}_1$, *'http://www.iaaf.org/.../ima01.jpg'*)

Figure 4.32: The analysis ABox of a sample web page

analysis ABox contains also additional information such as the URL of the web page, which is useful for the retrieval of the document.

$\mathit{Image}\ (\mathit{ima}_1)$
$\mathit{depicts}\ (\mathit{ima}_1, \mathit{new\_ind}_1)$
$\mathit{depicts}\ (\mathit{ima}_1, \mathit{new\_ind}_2)$
$\mathit{depictsSLC}\ (\mathit{ima}_1, \mathit{face}_1)$
$\mathit{depictsSLC}\ (\mathit{ima}_1, \mathit{body}_1)$
$\mathit{depictsSLC}\ (\mathit{ima}_1, \mathit{pole}_1)$
$\mathit{PersonFace}\ (\mathit{face}_1)$
$\mathit{PersonBody}\ (\mathit{body}_1)$
$\mathit{Pole}\ (\mathit{pole}_1)$
$\mathit{adjacent}\ (\mathit{face}_1, \mathit{body}_1)$
$\mathit{adjacent}\ (\mathit{body}_1, \mathit{pole}_1)$
$\mathit{Person}\ (\mathit{new\_ind}_1)$
$\mathit{hasPart}(\mathit{new\_ind}_1, \mathit{face}_1)$
$\mathit{hasPart}(\mathit{new\_ind}_1, \mathit{body}_1)$
$\mathit{PoleVault}\ (\mathit{new\_ind}_2)$
$\mathit{hasParticipant}(\mathit{new\_ind}_2, \mathit{new\_ind}_1)$
$\mathit{PoleVaulter}\ (\mathit{new\_ind}_1)$
$\mathit{hasPart}(\mathit{new\_ind}_2,\ \mathit{pole}_1)$

Figure 4.33: A sample image interpretation ABox

Assume that for the image in the web page, the interpretation ABox in Figure 4.33, and for the text that captions the image, the interpretation ABox in Figure 4.34 have been generated and returned by the semantic interpretation engine before the *interpetWebPage* web service call.

$Caption\ (cap_1)$
$depicts\ (cap_1, new\_ind_3)$
$depicts\ (cap_1, new\_ind_4)$
$depicts\ (cap_1, new\_ind_5)$
$depicts\ (cap_1, new\_ind_6)$
$depictsSLC\ (cap_1, pName_1)$
$depictsSLC\ (cap_1, sName_1)$
$PersonName\ (pName_1)$
$Running100mName\ (sName_1)$
$hasValue\ (pName_1, \text{‘}Usein\ Bolt\text{’})$
$hasValue\ (sName_1, \text{‘}100m')$
$Person\ (new\_ind_3)$
$hasPersonName\ (new\_ind_3, pName_1)$
$Running100mCompetition\ (new\_ind_4)$
$hasSportsName\ (new\_ind_4, sName_1)$
$hasPart\ (new\_ind_4, new\_ind_5)$
$Running100mRound\ (new\_ind_5)$
$hasPart\ (new\_ind_5, new\_ind_6)$
$Running100m\ (new\_ind_6)$

Figure 4.34: A sample caption interpretation ABox

Following the fusion algorithm in Algorithm 10, the image and caption interpretation ABoxes are merged into a single ABox $\mathcal{A}$. Afterwards, the function *query-same-as-candidates* is called to obtain the set of pairs of candidate individuals, which are then used to generate the set of same-as assertions $\mathcal{SA}$. In our example, $\mathcal{SA}$ contains the following assertions:

$(same\text{-}as\ new\_ind_1\ new\_ind_3)$
$(same\text{-}as\ new\_ind_2\ new\_ind_6)$

In the next step, the *compute-consistent-fused-interpretations* function is called with the enhanced background knowledge $\mathcal{T}_{enc}$, the ABox $\mathcal{A}$, and the set of same-as assertions $\mathcal{SA}$. According to $\mathcal{T}_{enc}$, *PoleVault* and *Running100m* are disjoint subconcepts of *SportsTrial*:

$$\begin{array}{rcl} Jumping & \sqsubseteq & SportsTrial \\ Running & \sqsubseteq & SportsTrial \sqcap \neg Jumping \\ PoleVault & \sqsubseteq & Jumping \\ Running100m & \sqsubseteq & Running \end{array}$$

Therefore, $\mathcal{A} \cup \mathcal{SA}$ is inconsistent w.r.t. $\mathcal{T}_{enc}$, and the *relax-image-interpretation-results* function is called. Informally speaking, the function *relax-image-interpretation-results* performs four subtasks:

1. Remove all SLC instances that are in the *depictsSLC* relation with the *Image* instance, and all relations between these SLC instances.

2. Remove all concept assertions from image interpretation results that declare a specific type of a *SportsTrial* or a specific type of an *Athlete*.

3. For the concept assertions removed in step 2, add new concept assertions that declare a *SportsTrial* or a *Athlete* instance in the image.

4. If $\mathcal{A} \cup \mathcal{SA}$ is consistent w.r.t. $\mathcal{T}_{enc}$, return $\{\mathcal{A} \cup \mathcal{SA}\}$, otherwise an empty set as result.

In our example, *relax-image-interpretation-results* proceeds as follows: First, all SLC instances in the *depictsSLC* relation with the *Image* instance, and all relations between these SLC instances are removed from $\mathcal{A}$. The exact syntax of the necessary RacerPro function calls is as follows:

(*forget-individual* $face_1$)
(*forget-individual* $body_1$)
(*forget-individual* $pole_1$)

It should be noted that RacerPro's *forget-individual* function removes not only concept assertions in which the given individual name appears, but also all role assertions in which the individual name appears.

Second, all assertions that declare a specific sports trial or a specific athlete in the image segment are removed from the ABox $\mathcal{A}$.

(*forget-concept-assertion new_ind$_1$ PoleVaulter*)
(*forget-concept-assertion new_ind$_2$ PoleVault*)

Third, new concept assertions are added to $\mathcal{A}$ in order to declare *new_ind$_1$* and *new_ind$_2$* as *Athlete* and *SportsTrial* instances, respectively.

(*instance new_ind$_1$ Athlete*)
(*instance new_ind$_2$ SportsTrial*)

Finally, the *relax-image-interpretation-results* function returns $\{\mathcal{A} \cup \mathcal{SA}\}$ as result, since $\mathcal{A} \cup \mathcal{SA}$ is consistent w.r.t. $\mathcal{T}_{enc}$. Consequently, the semantic interpretation engine answers the *interpretWebPage* web service call with the ABox shown in Figure 4.35, which is the fused interpretation ABox of the sample web page.

$\mathit{WebPage}\ (\mathit{wep}_1)$
$\mathit{CaptionedImage}\ (\mathit{cim}_1)$
$\mathit{Image}\ (\mathit{ima}_1)$
$\mathit{Caption}\ (\mathit{cap}_1)$
$\mathit{contains}\ (\mathit{wep}_1, \mathit{cim}_1)$
$\mathit{contains}\ (\mathit{cim}_1, \mathit{ima}_1)$
$\mathit{contains}\ (\mathit{cim}_1, \mathit{cap}_1)$
$\mathit{hasURL}\ (\mathit{wep}_1,$ *‘http://www.iaaf.org/.../news.html’*$)$
$\mathit{hasURL}\ (\mathit{ima}_1,$ *‘http://www.iaaf.org/.../ima01.jpg’*$)$
$\mathit{Image}\ (\mathit{ima}_1)$
$\mathit{depicts}\ (\mathit{ima}_1, \mathit{new\_ind}_1)$
$\mathit{depicts}\ (\mathit{ima}_1, \mathit{new\_ind}_2)$
$\mathit{Athlete}\ (\mathit{new\_ind}_1)$
$\mathit{SportsTrial}\ (\mathit{new\_ind}_2)$
$\mathit{Caption}\ (\mathit{cap}_1)$
$\mathit{depicts}\ (\mathit{cap}_1, \mathit{new\_ind}_3)$
$\mathit{depicts}\ (\mathit{cap}_1, \mathit{new\_ind}_4)$
$\mathit{depicts}\ (\mathit{cap}_1, \mathit{new\_ind}_5)$
$\mathit{depicts}\ (\mathit{cap}_1, \mathit{new\_ind}_6)$
$\mathit{depictsSLC}\ (\mathit{cap}_1, \mathit{pName}_1)$
$\mathit{depictsSLC}\ (\mathit{cap}_1, \mathit{sName}_1)$
$\mathit{PersonName}\ (\mathit{pName}_1)$
$\mathit{Running100mName}\ (\mathit{sName}_1)$
$\mathit{hasValue}\ (\mathit{pName}_1,$ *‘Usein Bolt’*$)$
$\mathit{hasValue}\ (\mathit{sName}_1, `100m')$
$\mathit{Person}\ (\mathit{new\_ind}_3)$
$\mathit{hasPersonName}\ (\mathit{new\_ind}_3, \mathit{pName}_1)$
$\mathit{Running100mCompetition}\ (\mathit{new\_ind}_4)$
$\mathit{hasSportsName}\ (\mathit{new\_ind}_4,\ \mathit{sName}_1)$
$\mathit{hasPart}\ (\mathit{new\_ind}_4, \mathit{new\_ind}_5)$
$\mathit{Running100mRound}\ (\mathit{new\_ind}_5)$
$\mathit{hasPart}\ (\mathit{new\_ind}_5, \mathit{new\_ind}_6)$
$\mathit{Running100m}\ (\mathit{new\_ind}_6)$
$\mathit{same\text{-}as}\ (\mathit{new\_ind}_1\ \ \mathit{new\_ind}_3)$
$\mathit{same\text{-}as}\ (\mathit{new\_ind}_2\ \ \mathit{new\_ind}_6)$

Figure 4.35: The fused interpretation ABox of the sample web page

Chapter 5

Evaluation

In this work we present a logic-based multimedia interpretation approach, which serves as the formal foundation for the automatic computation of deep-level semantic descriptions of multimedia documents. In order to obtain deep-level semantic descriptions of multimedia content, we propose a hybrid approach. As discussed in Chapter 2, the approach is hybrid in the sense that it combines surface-level information extraction and multimedia interpretation. Modality-specific analysis tools extract information based on low-level features of the content, and provide for surface-level semantic descriptions. Surface-level semantic descriptions serve as input for the semantic interpretation engine, which is an implementation of the logic-based multimedia interpretation approach. The semantic interpretation engine exploits background knowledge about the application domain and provides for deep-level semantic descriptions of the multimedia content.

The primary goal of this work is to show that knowledge representation formalisms and state-of-the-art reasoning engines can be used as a basis for building software systems, which solve practical problems of the present such as the automatic computation of deep-level semantic descriptions of multimedia content. Software systems exploiting formal methods, e.g. the semantic interpretation engine, offer the advantage that they can be reused easily to solve problems in different domains. Due to the declarative nature of formal methods that underly such software systems, it is sufficient to provide information at a high-level of abstraction. For example to use the semantic interpretation engine in a domain other than athletics, a domain expert does not need to specify a procedure to solve the problem but has to define just the domain ontologies and sets of interpretation rules. It is widely accepted that the definition of high-level information is an easier task, and thus less prone to errors compared to the definition of low-level program code.

Despite the advantages offered by the use of software systems exploiting formal methods, it is crucial to address issues such as performance of the systems and quality of the results obtained. As an important contribution of this work, we would like to explore the semantic interpretation engine, an implementation of the ideas developed in this work, and evaluate the results in a practical scenario.

This chapter is structured as follows: First, we explore the performance of the semantic interpretation engine through an experimental study. To provide a rigorous evolution, we use a test corpus consisting of 500 documents taken from the athletics domain. Second, we analyze the quality of the deep-level semantic descriptions, which have been computed automatically by the semantic interpretation engine. For the evaluation of the quality we exploit a large corpus of documents, which have been annotated by human experts by using concept and role names from the domain ontologies.

5.1 Performance and Scalability

In this section, we analyze the performance and scalability of our semantic interpretation engine through an experimental study. We describe the setup of the experiments used for the evaluation in detail. Afterwards, we present and discuss the results of our experiments.

This experimental study has two main goals:

- To analyze the performance of the system in terms of time spent in processing interpretation requests.
- To test the scalability of the system by analyzing the sensitivity of the interpretation service to the growth in analysis ABox size.

To achieve a rigorous evolution of the semantic interpretation engine, we test it using a reasonably large corpus of documents taken from websites. In particular, we use 500 web pages taken from the IAAF and USTAF web sites [Int09, USA09] as test data.

In order to obtain high-quality training data for analysis tools, these web pages have been annotated by multiple human annotators. More precisely, each web page has been annotated by two human experts, and then a third expert has adjudicated on discrepancies between the annotations of different annotators. In the literature, this process of obtaining high-quality annotation data is called *gold-standard annotation*, and the resulting gold-standard annotations are often used to measure system performance [MMSW06].

These gold-standard annotations created for the BOEMIE project include both surface-level information, e.g. sport event or person names, and deep-level information, e.g. such as sport trials and athletes. In the annotation process concept and role names from the athletics domain ontology AEO are used. Therefore, these annotations can be referred to as semantic annotations. Later, the gold-standard annotations have been transformed to DL ABoxes. We henceforth call these ABoxes *gold-standard interpretation ABoxes* to emphasize the fact that they are derived from gold-standard annotations and contain both surface and deep-level information.

In our first experiment, we want to study the performance and scalability of the interpretation service. In order to test the text interpretation web service of the semantic interpretation engine, we need analysis ABoxes, which contain surface-level information only. Therefore, we remove all deep-level information from gold-standard interpretation ABoxes, and name the resulting ABoxes *gold-standard analysis ABoxes*.

Our first experiment has the following setup: A client application calls the *interpretText* web service of the semantic interpretation engine serially for each one of the 500 web pages. The gold-standard analysis ABoxes serve as input for the interpretation process. We use three metrics as performance measures for the interpretation of an analysis ABox: the number of fiat assertions, the number of all assertions, and the time spent to process the interpretation request.

We define the time spent for interpretation as the time spent between the moment at which the interpretation request arrives at the semantic interpretation engine, and the moment at which the response to the call, i.e. the interpretation ABox, is ready to be sent to the client. This definition considers solely the time needed by the interpretation algorithm, and does not involve the time needed for the communication between the client and the semantic interpretation engine. This enables us to measure the system performance independent of external factors such as network latency that may vary substantially.

The experiments were run on a Macintosh machine (OSX 10.4.11) with a 2.16 GHz Intel Core Duo processor and 2 GB of main memory. The semantic interpretation engine was deployed in the servlet container Apache Tomcat 6.0.14. We used Sun JVM version 1.5.0_16. The maximum heap size allocated to Java was set to 512 megabytes. The semantic interpretation engine was configured to manage a single instance of the DL reasoner RacerPro in version 1.9.3. The background knowledge used by the semantic interpretation consists of the ontologies AEO version 2.13, MCO version 2.13, GIO version 2.5 and the text interpretation rule file sports_rules_text_1.racer, which was last modified at 6th of March 2009.

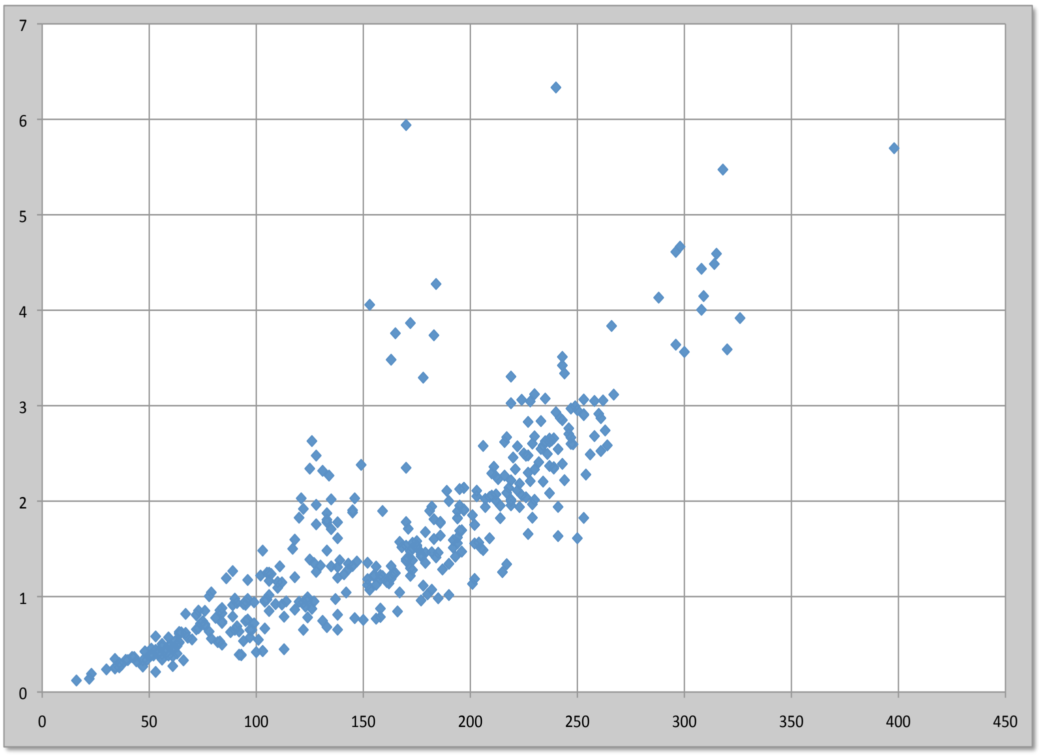


Figure 5.1: The number of fiat assertions (x) and the time (y) spent in minutes for the interpretation of 500 text analysis ABoxes.

Figure 5.1 shows the performance results of our semantic interpretation engine for the interpretation of a corpus consisting of 500 text analysis ABoxes. Each point in the diagram is a different text analysis ABox. For each text analysis ABox the value at the horizontal axis denotes the number of fiat assertions, and the value at the vertical axis denotes the time spent for interpretation in minutes.

To get a clear picture of the relation between the number of fiat assertions and the time spent for interpretation, we built clusters of ABoxes with similar amount of fiat assertions. Later we have selected an ABox from each cluster as representative average members of that cluster. The diagram in Figure 5.2 shows the number of fiat assertions and the time spent for the interpretation of the selected text analysis ABox.

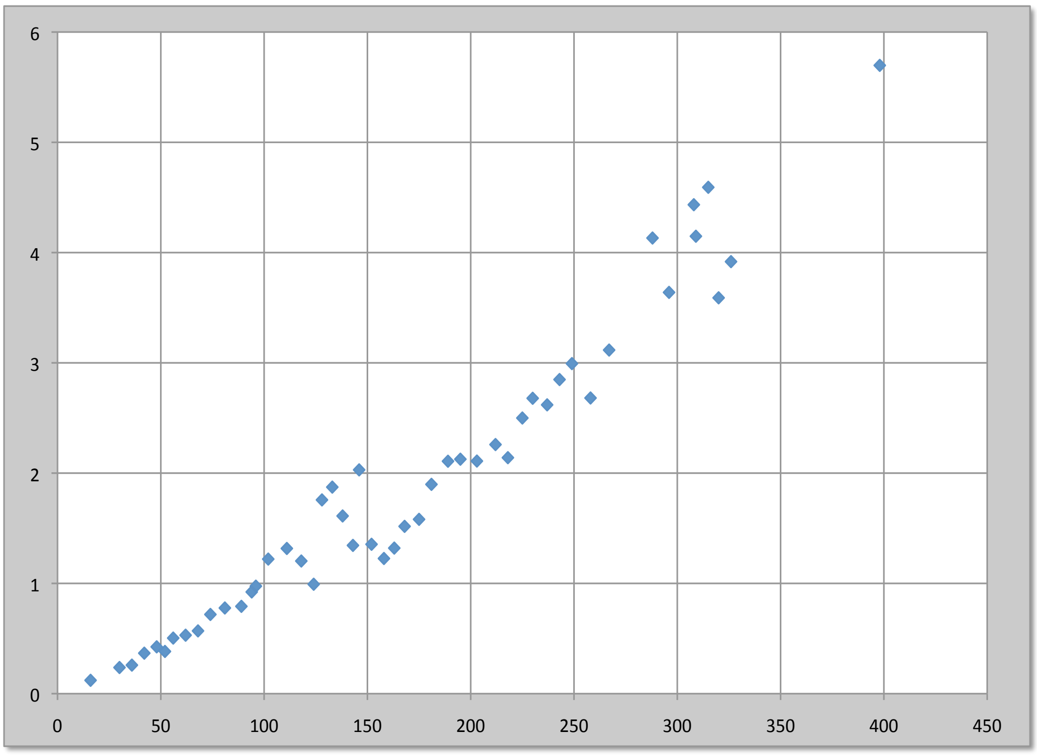


Figure 5.2: The number of fiat assertions (x) and the time (y) spent in minutes for the interpretation of selected text analysis ABoxes.

The number of fiat assertions in an analysis ABox plays an important role on the amount of time needed to interpret that analysis ABox, because every fiat assertion represents an observation that is questioned. Therefore, the semantic interpretation engine is requested to compute preferred explanations for each fiat assertion.

Figure 5.3 shows the performance results of the semantic interpretation engine for the interpretation of the same 500 text analysis ABoxes, but with respect to the number of all assertions in the ABoxes. Like the diagram in Figure 5.1, each point in the diagram is a different analysis ABox, and the vertical axis denotes the time spent for interpretation in minutes. However, different from the diagram in Figure 5.1, the horizontal axis denotes the sum of fiat and bona-fide assertions.

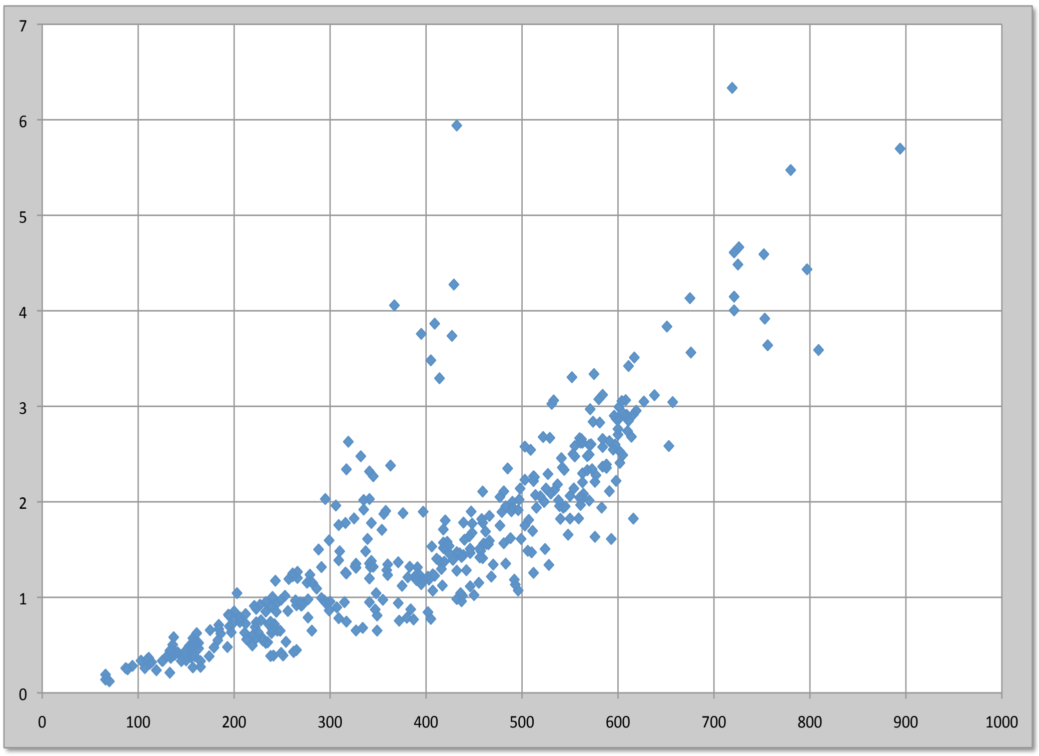


Figure 5.3: The sum of fiat and bona fide assertions (x) and the time (y) spent in minutes for the interpretation of 500 text analysis ABoxes.

In this experiment, assertions that regard concrete domains, so-called *data properties* in OWL, are excluded. The values shown in the horizontal axis of the diagram in Figure 5.3 include only so-called *object properties*, i.e. assertions that are not related to concrete domains. In our approach, contrary to data properties, object properties are used as predicates in text interpretation rules, and thus affect the time spent for interpretation.

Comparing the two diagrams in Figure 5.1 and Figure 5.3, we observe that not only the amount of fiat assertions affect the amount of processing time, but also the sum of fiat and bona fide assertions. For example, in Figure 5.1 we observe that the largest amount of time, more than 6 minutes, was requested for the interpretation of an analysis ABox with less than 250 fiat assertions. We can also see that the interpretation of many other ABoxes with a similar number of fiat assertions took between 1.6 and 3.6 minutes. In fact, the difference indicates the existence of another factor on the performance. In Figure 5.3 we identify the reason for this difference: The sum of fiat and bona-fide assertions in the analysis ABox that required more than 6 minutes for interpretation is considerably higher than the sum of fiat and bona fide assertions in other analysis ABoxes, which have a similar amount of fiat assertions.

To clarify the relationship between the number of fiat and bona fide assertions, and the time spent for interpretation, we built again clusters of ABoxes with similar amount of fiat assertions. From each cluster we have selected an ABox to represent all ABoxes in that cluster. The plot in Figure 5.4 shows the number of fiat and bona-fide assertions, and the time spent for the interpretation of the selected text analysis ABox.

The diagrams in Figure 5.2 and Figure 5.4 visualize the progression of the time required for interpretation. We can observe that the time required for interpretation increases approximately linear to the increase in the amount of fiat assertions and the sum of fiat and bona fide assertions. The results obtained are very good, because they show that the system scales quite well.

In a practical scenario, the multimedia interpretation process can be considered as part of an offline process where a repository of semantic descriptions is prepared before the repository can be exploited by a multimedia retrieval system. Considering also the fact that this experimental study has been conducted on hardware below today's standard, the performance of the system is quite promising.

In light of this experimental study, we identify further possibilities for improving the performance and scalability of the semantic interpretation engine, especially in practical settings:

- Several reasoning tasks provided by RacerPro are constantly improved. The semantic interpretation engine will benefit from future improvements in RacerPro such as the support for incremental reasoning or optimizations of the abductive inference service.

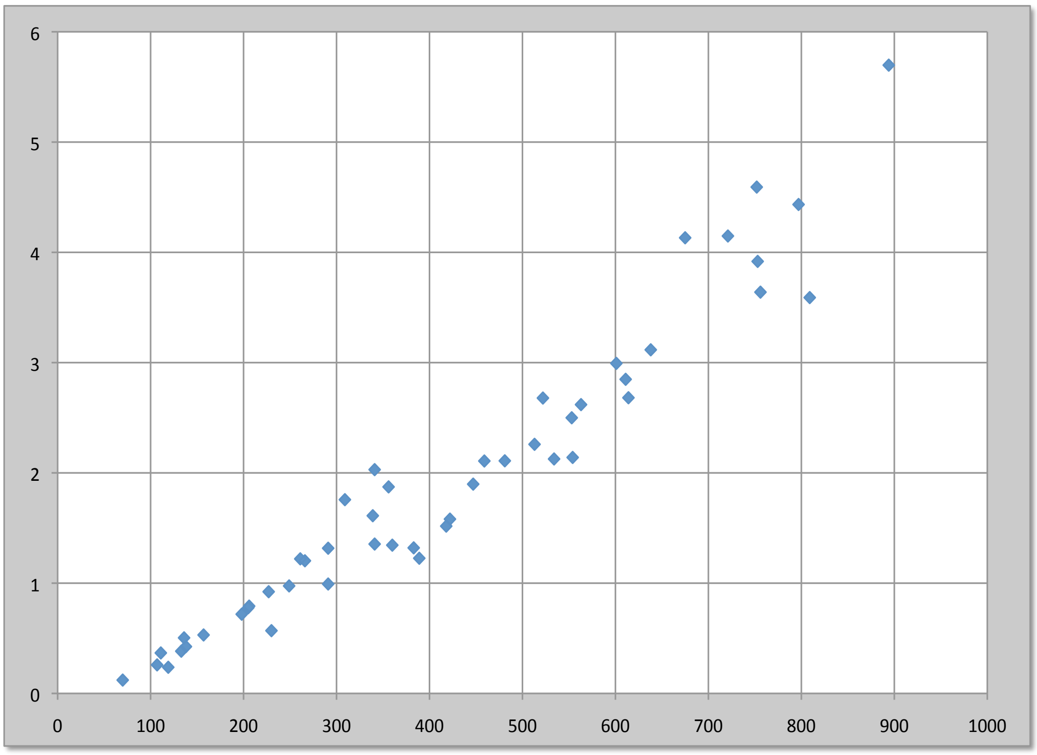


Figure 5.4: The number of fiat and bona fide assertions (x) and the time (y) spent in minutes for the interpretation of selected text analysis ABoxes.

- Analysis ABoxes can be interpreted in parallel to reduce the overall time spent for interpretation. This is particularly important, if the document corpus to be interpreted is large. For this purpose, the semantic interpretation engine should manage multiple RacerPro instances, which are dedicated to text interpretation, and distribute the requests among the RacerPro instances.

These ideas can be realized with reasonable effort, due to the loosely-coupled and flexible design of the semantic interpretation engine.

5.2 Quality of Interpretation Results

In the previous chapters of this work, we have devised an algorithm to realize a logic-based multimedia interpretation approach. As with any algorithm proposed to solve a problem, it is not only crucial to analyze the performance and scalability of its implementation, but also to investigate how well the computed solutions are. Therefore the quality of the interpretation results computed by the semantic interpretation engine have to be examined carefully in order to make a statement of its applicability in practice.

In this section, we analyze and evaluate the quality of the interpretation results computed by the semantic interpretation engine through an experimental study on web pages with athletics news. To this end, we propose a method for the comparison of interpretations computed by the semantic interpretation engine with human-made annotations. Later, we present the results of the comparison that enables the evaluation of the quality of interpretation results.

In this experiment, our main goal is to investigate how well the interpretations, i.e. deep-level semantic descriptions, computed by the semantic interpretation engine are. This can only be done by comparing the computed interpretations of a document with interpretations of the same document, which are known to be correct and thus can serve as ground truth. For this purpose, we exploit gold-standard interpretation ABoxes, which have been generated from surface- and deep-level annotations created by human experts. In our comparison gold-standard interpretation ABoxes are considered as free from errors and serve as ground truth. Therefore, in the ideal case, automatically computed interpretations should semantically be identical to gold-standard interpretation ABoxes, even though the individual names, which are instances of DLCs, are different. Notice that SLC instances have the same names both in automatically generated and gold-standard interpretation ABoxes, since automatically generated interpretation ABoxes are computed based on the information in gold-standard analysis ABoxes, which contain SLC instances only.

For comparing an automatically generated interpretation ABox with the corresponding gold-standard interpretation ABox, the straightforward approach is to transform the gold-standard interpretation ABox into a Boolean conjunctive query. The transformation is done by replacing the individual names with variable names, as discussed in the definition of the function *transform* in Section 3.3.2. Later, the query is posed to a knowledge base consisting of the automatically generated interpretation ABox and the TBox. If the query can be answered positively, then it can be said that the automatically generated interpretation ABox contains semantically equivalent information.

However, the straightforward approach has two crucial disadvantages:

- The approach is too coarse-gained in the sense that it only returns true or false but no gradual values to indicate the degree of matching between the two ABoxes. The Boolean conjunctive query can only be answered positively if for every assertion in gold-standard interpretation ABox, a semantically equivalent assertion can be found in the automatically generated interpretation ABox. This means that even if for a single assertion a semantic equivalent cannot be found, we get a negative answer for the query, and do not know for how many percent of the assertions from the gold-standard interpretation ABox semantically equivalent assertions could be found in the automatically generated interpretation ABox.

- Considering the amount of assertions in the ABoxes used, the approach has limited usability in practice. In a practical scenario, both automatically generated and gold-standard interpretation ABoxes have a large number of assertions, i.e., several hundreds and more. Depending on the expressive power of the DL used, conjunctive query answering has high computational complexity.

In view of the disadvantages of the straightforward approach, we pursue a different approach. To evaluate the quality of interpretation results, our approach exploits recall and precision, which are established metrics for benchmarking of keyword-based retrieval, also known as *Boolean retrieval*, of documents. However, the standard use of these metrics in the context of Boolean retrieval, in which appearances of certain keywords in a document are counted, is too vague to evaluate deep-level semantic descriptions. Considering the fact that the deep-level semantic descriptions computed by the semantic interpretation engine are not keywords or labels, but relational structures, i.e. concept and role assertions, it is obvious that counting the number of concept assertions is not sufficient for a qualitative evaluation. Therefore, we modify the standard approach used in the evaluation of Boolean retrieval techniques.

Our approach exploits Boolean conjunctive queries to check for the existence of certain relational structures in interpretation ABoxes. This enables more precise computation of recall and precision values, and thus, a more precise evaluation of the interpretation results. For example, instead of counting the number of Person instances in an interpretation ABox, in our approach, we count only such Person instances, which are correctly related to surface-level information such as name, gender and age. More precisely, we check whether certain relational structures found in the human-made annotations can also be found in the interpretation ABoxes computed by the semantic interpretation engine.

In order to achieve better scalability than the straightforward approach, our approach starts with the examination of surface-level information and proceeds bottom-up. To this end, surface-level information found in a gold-standard interpretation ABox is transformed into a Boolean conjunctive query and posed to a knowledge base consisting of the corresponding automatically generated interpretation ABox and the TBox. Next, the approach proceeds with the examination of more abstract, i.e. deep-level, information. For this purpose, the approach checks first the existence of aggregates, which contain only surface-level concept instances as parts. Only if such aggregates exist in the automatically generated interpretation ABox, the existence of more abstract aggregates that contain other aggregates are checked.

The approach inspects relational structures bottom-up starting from surface-level information and proceeds with deep-level information, i.e. aggregates. The approach proceeds in the same way as the logic-based interpretation approach that computes the relational structures. It only checks for more abstract information if less abstract information necessary to hypothesize such abstract information exists. Therefore, the bottom-up Boolean conjunctive query based evaluation approach is particularly suitable for the evaluation of the interpretation results.

In our second experiment, we use a corpus of 100 web pages with athletics news. The gold-standard analysis ABoxes of this corpus serve as input for the semantic interpretation engine. The resulting automatically generated interpretation ABoxes are compared with corresponding gold-standard interpretation ABoxes of the corpus. In the context of the BOEMIE project, a software tool has been implemented to realize our evaluation approach. The software tool automatically generates a set of Boolean conjunctive queries from a gold-standard interpretation ABox, and poses it to a knowledge base consisting of the corresponding automatically generated interpretation ABox and TBox. Furthermore the software tool calculates recall and precision values for each document in the corpus, and for the whole corpus.

Following Koshafian and Baker's notation [KB96, page 358], we introduce the classical definition of the terms *recall* and *precision*:

$$Recall = \frac{Number\ of\ Relevant\ Objects\ Returned}{Total\ Number\ of\ Relevant\ Objects\ in\ the\ Collection} \tag{5.1}$$

$$Precision = \frac{Number\ of\ Relevant\ Objects\ Returned}{Total\ Number\ of\ Objects\ Returned} \tag{5.2}$$

where both recall and precision have values in the interval [0,1]. In these formulas objects are documents from a collection, also known as corpus. In the information retrieval context, the goal is to retrieve all relevant documents for a query, and no other documents which are irrelevant. In terms of the recall and precision the goal is to achieve for both values as close to 1 as possible.

In our context, we do not count the number of documents to calculate recall and precision values, but the number of aggregates, i.e. deep-level concept instances, in a document. More precisely, we use Boolean conjunctive queries to check the existence of certain relational structures. We consider the existence of an aggregate in the automatically generated ABox as a relevant answer only if it is in the same relations with its parts, i.e. surface-level instances or other deep-level instances, as its counterpart in the corresponding gold-standard interpretation ABox.

Therefore, in our case, the recall and precision values for a DLC concept are calculated according to the following formula:

$$Recall = \frac{|Rel_{in}|}{|Rel_{go}|} \tag{5.3}$$

$$Precision = \frac{|Rel_{in}|}{|Ret_{in}|} \tag{5.4}$$

where $|Ret_{in}|$ denotes the number of all instances of a certain DLC concept in the automatically generated interpretation ABox, and $|Rel_{in}|$ the number of instances, which are relevant. The relevant DLC concept instances are identified through Boolean conjunctive queries. $|Rel_{go}|$ denotes the number of all instances of a certain DLC concept in the gold-standard interpretation ABox, which are relevant by definition since the gold-standard interpretation ABox serves as ground truth for the comparison.

Table 5.1 shows the average recall and precision values measured in our second experiment for a corpus consisting of 100 web pages with athletics news. The values are measured for each deep-level concept, of which at least an instance exists in the corpus.

The values in Table 5.1 indicate that there exist 15 instances of the deep-level concept *SportsCompetition* in the gold-standard interpretation ABoxes, whereas in the interpretation ABoxes generated by the semantic interpretation engine 16 instances of *SportsCompetition* exist, of which 11 are relevant.

Deep-Level Concept	$\|Rel_{in}\|$	$\|Rel_{go}\|$	$\|Ret_{in}\|$	Recall	Precision
JavelinThrowCompetition	2	2	2	1.00	1.00
Running100mCompetition	13	13	13	1.00	1.00
MarathonCompetition	12	16	15	0.75	0.80
PoleVaultCompetition	6	9	8	0.67	0.75
HammerThrowCompetition	7	7	7	1.00	1.00
LongJumpCompetition	8	8	8	1.00	1.00
HighJumpCompetition	4	4	4	1.00	1.00
SportsCompetition	11	15	16	0.73	0.69
SportsTrial	347	427	482	0.81	0.72
SportsEvent	47	53	68	0.89	0.69
Athlete	292	325	355	0.90	0.82
Person	134	140	769	0.96	0.17

Table 5.1: Recall and precision values for deep-level concepts

The recall and precision values measured in this experimental study are very good with an exception, namely the precision value for the concept *Person*. The low precision value measured for the concept *Person* is caused by different strategies followed by human annotators creating annotations, and domain experts defining the interpretation rules for the text modality. Our analysis of the gold-standard interpretation ABoxes showed that human annotators have annotated a *Person* instance only if they could observe a person's name in relation with another personal attribute such as gender, age or nationality. On the contrary, the text interpretation rules have been defined under the assumption that even if further personal attributes cannot be related to a person name, the existence of a person name gives enough evidence to hypothesize a *Person* instance.

The recall and precision values in Table 5.1 confirm us in our belief that deep-level semantic descriptions of multimedia documents computed by the semantic interpretation engine are of high-quality. The deep-level semantic descriptions can be exploited to build a repository where the semantics of multimedia content are represented appropriately in the sense that the repository can be used for the retrieval of documents with high recall and precision.

Another advantage of building a repository with deep-level semantic descriptions of multimedia content is the support for flexible and convenient querying that is intrinsic to ontology-based retrieval. Compared to keyword-based retrieval techniques, queries can be composed flexibly using the concept and role names defined in the ontology such that the multimedia content can be retrieved more precisely. For example, instead of asking for documents containing information about Blanka Vlasic as in the case of keyword-based retrieval, it is possible to ask for multimedia documents that contain not only information about the performance of Blanka Vlasic in a specific sports event, but also an image of Blanka Vlasic which depicts her during a high jump trial in that sports event.

Chapter 6

Conclusions

The realization of so-called 'semantic' or 'intelligent' applications requires the semantics of information be represented in an appropriate form in order to make it possible to understand and fulfill the requests of users and machines for accessing information found in the web or in document repositories. Besides the Semantic Web vision, also present information systems such as content management systems require intelligent retrieval of content in order to offer more valuable information and services. Moreover, the realization of convenient and intelligent information retrieval requires the availability and accessibility of the semantics of information. In many scenarios, the available information is not strictly structured, as in the case of information stored in databases, but is loosely structured and exists in multimedia documents. Therefore semantic metadata describing the semantics of information in multimedia documents is essential for building semantic applications.

Motivated by the need for valuable semantic metadata about multimedia documents, we developed a logic-based approach to multimedia interpretation in this thesis. In this chapter we conclude this thesis by summarizing the major achievements of the work and presenting promising directions for future work.

6.1 Summary

With respect to the amount of time and other resources needed to manually annotate documents, our multimedia interpretation approach is required to automatically generate semantic metadata with as few resources as possible, and to be flexible enough for application in different domains with little effort and low cost.

The fact that standard inference problems have been formalized and well-studied in formal languages in the past, and the availability of inference engines that provide for successful implementations of standard inference services make formal languages a promising candidate for solving the multimedia interpretation problem. Therefore we have chosen a logic-based formalism for knowledge representation and reasoning as the foundation of our approach.

We have formalized a declarative multimedia interpretation algorithm based on formal inference services. Besides standard DL inference services, our interpretation algorithm exploits abduction as the key inference service for explanation generation. As part of this work, we also formalized ABox abduction in DLs as a non-standard inference service.

Typically multimedia documents comprise of multiple segments, where each segment contains information in at least one modality. On the other hand, current analysis tools are specialized in the extraction of information from a particular modality. Therefore a solution for the generation of semantic metadata about multimedia documents using modality-specific analysis tools is required.

In this thesis, we proposed a hybrid approach for the realization of the multimedia interpretation task. We presented the semantic interpretation engine, an implementation of the multimedia interpretation algorithm, which has been realized as a distributed software system incorporating reasoning engines for inference tasks. Our approach is hybrid in the sense that it integrates modality-specific analysis tools and the semantic interpretation engine into a coherent framework. In this framework the interpretation of a multimedia document consists of two steps: First, each segment of a multimedia document is interpreted with respect to information extracted from a particular modality. Second, interpretations of segments are fused to obtain an interpretation of the whole multimedia document.

From a software engineering point of view, the hybrid approach has several advantages compared to the alternative solution in which existing analysis tools have to be enhanced in order to extract, in addition to surface-level information, also deep-level information. In the hybrid approach, the background knowledge required to interpret surface-level information is defined and later exploited by a central component, namely, by the semantic interpretation engine. Therefore, in the hybrid approach the definition of the background knowledge by the domain expert is sufficient, whereas in the alternative approach every analysis tool has to be enhanced and modified individually. Furthermore, the hybrid approach is more flexible and open, because an analysis tool can easily be replaced by another one without modifications in the rest of the framework.

An important objective of this work has been the development of a multimedia interpretation approach that can be realized as a software component and applied to solve practical problems. Therefore, we conducted an experimental study to analyze the runtime performance and scalability of the semantic interpretation engine. In addition, we examined the quality of the semantic metadata generated by the semantic interpretation engine through another experimental study, and evaluated the quality in terms of recall and precision.

Putting it all together, we showed that semantic metadata about multimedia documents can be generated automatically using a declarative, logic-based approach. In [EKM+07a], we discussed how this can be achieved without changing the logic or the tableaux calculi used, but by enhancing DLs with rules and by exploiting a state-of-the-art reasoning engine that supports abductive inference in this expressive formalism.

Furthermore, in [EKM+07a, EKM08a, CEF+07] we showed that the approach can be realized as a distributed software system that serves as the core component of a large application project aiming at the extraction, interpretation and retrieval of multimedia documents from the athletics domain. The results of our experimental studies are very promising and prove the fitness of our approach for practical use. We believe that due to its flexible and open architecture, the semantic interpretation engine can easily be integrated into other software systems that require cost- and time-efficient generation of semantic metadata.

6.2 Outlook

In the light of insights gained in this thesis, we identified several promising directions for future work. We are planning to employ our semantic interpretation engine in domains other than the athletics domain in order to study the performance of the context-dependent criteria used in this work, for reducing the number of explanations by selecting preferred explanations only. This study might enable the identification of further context-dependent criteria for multimedia interpretation.

In its current state, the multimedia interpretation approach developed in this work considers all surface-level information extracted from a multimedia segment as input. Another interesting research direction is to develop methods for focusing the attention of the interpretation agent on relevant parts of the surface-level information extracted from a multimedia segment. Successful determination of the interpretation discourse will improve not only the runtime performance of the approach, but also the quality of the generated semantic metadata.

Finally, more comprehensive experimental studies are required to measure and evaluate the quality of semantic metadata. Due to the lack of gold-standard annotations for all segments of multimedia documents, the qualitative evaluation presented in this thesis is done based on textual segments of web pages. Similar to benchmarks designed to compare information retrieval systems, benchmarks for software systems generating semantic annotations have to be defined in the future.

References

[AGP07] M. Anthimopoulos, B. Gatos, and I. Pratikakis. Multiresolution Text Detection in Video Frames. In *Proceedings of 2nd International Conference on Computer Vision Theory and Applications (VISAPP 1007)*, pages 161–166, Barcelona, Spain, March 2007. {108}

[AGPP07] M. Anthimopoulos, B. Gatos, I. Pratikakis, and S. J. Perantonis. Detecting Text in Video Frames. In *Proceeding of the 4th IASTED International Conference on Signal Processing, Pattern Recognition, and Applications (SPPRA 2007)*, pages 39–44, Innsbruck, Austria, February 2007. {108}

[AHB$^+$93] D. E. Appelt, J. R. Hobbs, J. Bear, D. J. Israel, and M. Tyson. FASTUS: A Finite-state Processor for Information Extraction from Real-world Text. In *Proceedings of IJCAI*, pages 1172–1178, 1993. {14}

[AL97] A. Aliseda-Llera. *Seeking Explanations: Abduction in Logic, Philosophy of Science and Artificial Intelligence.* PhD thesis, University of Amsterdam, 1997. {67, 82}

[Ali06] A. Aliseda. *Abductive Reasoning: Logical Investigations into Discovery and Explanation*, volume 330 of *Synthese Library.* Springer, 2006. {20}

[And76] J. R. Anderson. *Language, Memory, and Thought.* Lawrence Erlbaum Associates, Hillsdale, NJ, 1976. {49}

[Apa09] Apache Software Foundation. The Apache Tomcat Servlet Container website http://tomcat.apache.org/, February 2009. {111}

[AWB87] J. Y. Aloimonos, I. Weiss, and A. Bandyopadhyay. Active Vision. In *Proceedings of 1st International Conference on Computer Vision*, pages 35–54, Washington DC, 1987. IEEE Computer Society. {78}

[Bal91] D. H. Ballard. Animate Vision. In *Artificial Intelligence Journal*, volume 48, pages 57–86, 1991. {78}

[BCM$^+$03] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications.* Cambridge University Press, 2003. {28, 37, 38, 48}

[BH91] F. Bader and P. Hanscke. A Schema for Integrating Concrete Domains into Concept Languages. In *Proceedings of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91)*, pages 452–457, 1991. {40}

[BMP02] S. Belongie, J. Malik, and J. Puzicha. Shape Matching and Object Recognition Using Shape Contexts. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, volume 24(24), pages 509–522, 2002. {14}

[Boß08] S. Boßung. *Conceptual Content Modeling Languages, Applications, and Systems.* PhD thesis, Hamburg University of Technology, 2008. {35}

[Bri93] D. Brill. *Loom Reference Manual.* Information Sciences Institute, University of Southern California, 4676 Admiralty Way, Marina del Rey, CA 90292, December 1993. {25}

[BVDR$^+$03] F. Brun-Vezinet, D. Descamps, A. Ruffault, B. Masquelier, V. Calvez, G. Peytavin, F. Telles, L. Morand-Joubert, J.L. Meynard, M. Vray, D. Costagliola, and the Narval (ANRS 088) Study group. Clinically Relevant Interpretation of Genotype for Resistance to Abacavir. *AIDS*, 17(12):1795–1802, August 15 2003. {76}

[Can86] J. F. Canny. A Computational Approach To Edge Detection. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 8(6):679–698, November 1986. {26}

[CDD$^+$06] S. Castano, K. Dalakleidi, S. Dasiopoulou, S. Espinosa, A. Ferrara, G. N. Hess, V. Karkaletsis, A. Kaya, S. Melzer, R. Möller, S. Montanelli, and G. Petasis. 4.1 Methodology and Archtecture for Multimedia Ontology Evolution. Project Deliverable Version 1.0 Final, The BOEMIE Consortium, December 2006. {108}

[CEF$^+$07] S. Castano, S. Espinosa, A. Ferrara, V. Karkaletsis, A. Kaya, S. Melzer, R. Möller, S. Montanelli, and G. Petasis. Ontology Dynamics with Multimedia Information: The BOEMIE Evolution Methodology. In *Proceedings of the ESWC International Workshop on Ontology Dynamics (IWOD 07)*, Innsbruck, Austria, June 2007. {68, 177}

[CFML08] S. Castano, A. Ferrara, S. Montanelli, and D. Lorusso. Instance Matching for Ontology Population. In *Proceedings of the Sixteenth Italian Symposium on Advanced Database Systems (SEBD 2008)*, pages 121–132, Mondello, PA, Italy, June 2008. {109}

[CG91] E. Charniak and R. Goldman. Probabilistic Abduction For Plan Recognition. Technical report, Brown University, Tulane University, 1991. {20, 72, 73, 75, 77}

[CGL08] D. Calvanese, G. D. Giacomo, and M. Lenzerini. Conjunctive Query Containment and Answering under Description Logic Constraints. *ACM Transactions on Computational Logic*, 9(3):1–31, 2008. {48, 51}

[CGT89] S. Ceri, G. Gottlob, and L. Tanca. What you always wanted to know about datalog (and never dared to ask). *IEEE Transactions on Knowledge and Data Engineering*, 1:146–166, 1989. {56}

[Coo90] G. F. Cooper. The Computational Complexity of Probabilistic Inference using Bayesian Belief Networks. *Artificial Intelligence*, 42:393–405, 1990. {74}

[CR80] R. A. Cote and S. Robboy. Progress in medical information management. Systematized nomenclature of medicine (SNOMED). *The Journal of the American Mdeical Association*, 243:756–762, February 1980. {36}

[CY90] J. J. Clark and A. L. Yuille. *Data Fusion for Sensory Information Processing Systems*. Kluwer Academic, Norwell, MA, USA, 1990. {65, 79}

[CY99] S. Cucerzan and D. Yarowsky. Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence. In *Proceedings of Joint SIGDAT Conf. on Emprical Methods in Natural Language Processing and Very Large Corpora*, 1999. {14}

[DDG$^+$07] K. Dalakleidi, S. Dasiopoulou, E. Giannakidou, A. Kaya, V. K. Papastathis, G. Petasis, and V. Tzouvaras. 3.2 Domain Ontologies - Version 1. Project Deliverable Version 2.0 Final, The BOEMIE Consortium, February 2007. {106}

[DNR97] F. M. Donini, D. Nardi, and R. Rosati. Autoepistemic Description Logics. In *Proceedings of the International Joint Conference on Artificial Intelligence 97)*, pages 136–141, 1997. {52}

[EKM$^+$07a] S. Espinosa, A. Kaya, S. Melzer, R. Möller, and M. Wessel. Multimedia Interpretation as Abduction. In *Proceedings of the International Workshop on Description Logics DL-2007*, 2007. {68, 177}

[EKM$^+$07b] S. Espinosa, A. Kaya, S. Melzer, R. Möller, and M. Wessel. Towards a Media Interpretation Framework for the Semantic Web. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI'07)*, pages 374–380, Washington, DC, USA, November 2007. IEEE Computer Society. {68, 108}

[EKM08a] S. Espinosa, A. Kaya, and R. Möller. On Ontology Based Abduction for Text Interpretation. In *Proceedings of 9th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2008)*, number 4919 in Lecture Notes in Computer Science, pages 194–2005, Haifa, Israel, February 2008. Springer. {68, 108, 177}

[EKM08b] S. Espinosa, A. Kaya, and R. Möller. Ontology and Rule Design Patterns for Multimedia Interpretation. In *BOEMIE workshop*, Koblenz, Germany, December 2008. {124}

[EKM09] S. Espinosa, A. Kaya, and R. Möller. The BOEMIE Semantic Browser: A Semantic Application Exploiting Rich Semantic Metadata. In *Proceedings of the Applications of Semantic Technologies Workshop (AST-2009)*, Lübeck, Germany, October 2009. {109}

[EKS06] C. Elsenbroich, O. Kutz, and U. Sattler. A Case for Abductive Reasoning over Ontologies. In *OWL: Experiences and Directions*, 2006. {67, 81, 82}

[FHH04] R. Fikes, P. Hayes, and I. Horrocks. OWL-QL – A Language for Deductive Query Answering on the Semantic Web. *Web Semantics: Science, Services and Agents on the World Wide Web*, 2(1):19–29, December 2004. {48}

[Fow04] M. Fowler. *UML Distilled*. Addison-Wesley, 3th edition edition, 2004. {52}

[GHJV93] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Abstraction and Reuse of Object-Oriented Design*. Addison-Wesley, 1993. {102}

[GHLS07] B. Glimm, I. Horrocks, C. Lutz, and U. Sattler. Conjunctive Query Answering for the Description Logic SHIQ. In *Proceedings of the 20th International Joint Conference on Artificial Intelliegence IJCAI-07*. AAAI Press, 2007. {45}

[GL02] R. Guigno and T. Lukasiewicz. A Probabilistic Extension of SHOQ(D) for Probabilistic Ontologies in Semantic Web. In *Proceedings JELIA-2002*, volume 2424 of *Lecture Notes in Computer Science*, pages 86–97. Springer, 2002. {53}

[GN87] M. R. Genesereth and N. J. Nilsson. *Logical Foundations of Artificial Intelligence*. Morgan Kaufmann Publ. Inc., Los Altos, CA, 1987. {20}

[Gol90] R. Goldman. A Probabilistic Approach to Language Understanding. Technical report, Brown University, Providence, RI, USA, 1990. {73}

[Gri03] R. Grishman. Information Extraction. In *Handbook of Computational Linguistics Information Extraction*, 2003. {14}

[Gru93a] T. R. Gruber. A Translation Approach to Portable Ontology Specifications. Technical Report KSL 92-71, Knowledge Systems Laboratory, Computer Science Department, Stanford University, Stanford, California 94305, April 1993. {36}

[Gru93b] T. R. Gruber. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. *International Journal Human-Computer Studies*, 43:907–928, March 1993. {36}

[HBN07] M. Horridge, S. Bechhofer, and O. Noppens. Igniting the OWL 1.1 Touch Paper: The OWL API. In *OWL: Experiences and Directions Third International Workshop (OWLED 07)*, 2007. {48}

[HM87] J. Hobbs and P. Martin. Local Pragmatics. In *Proceedings of International Conference on Artificial Intelligence*, pages 520–523, Milano, Italy, August 1987. {69}

[HM01] V. Haarslev and R. Möller. RACER System Description. In *Proceedings of the International Joint Conference on Automated Reasoning (IJCAR 2001)*, volume 2083 of *Lecture Notes in Computer Science*, pages 701–705. Springer, 2001. {28, 45}

[HMS04] U. Hustadt, B. Motik, and U. Sattler. Reducing SHIQ-Description Logic to Disjunctive Datalog Programs. In *Proceedings of the 9th International Conference on the Principles of Knowledge Representation and Reasoning (KR 2004)*, pages 152–162, 2004. {45}

[HS88] C. Harris and M. Stephens. A Combined Corner and Edge Detector. In *Proceedings of 4th Alvey Vision Conference*, pages 147–151, 1988. {14}

[HSAM90] J. R. Hobbs, M. Stickel, D. Appelt, and P. Martin. Interpretation as Abduction. Technical Report 499, AI Center, SRI International, 1990. {70, 72, 93}

[HSME88] J. Hobbs, M. Stickel, P. Martin, and D. Edwards. Interpretation as Abduction. In *26th Annual Meeting of the Association for Computational Linguistics: Proceedings of the Conference*, 1988. {69, 72, 77}

[HSTT99] I. Horrocks, U. Sattler, S. Tessaris, and S. Tobies. Query Containment using a DLR ABox. LTCS report LTCS-99-15, LuFG Theoretical Computer Science, RWTH Aachen, Germany, 1999. {51}

[HSTT00] I. Horrocks, U. Sattler, S. Tessaris, and S. Tobies. How to Decide Query Containment Under Constraints Using a Description Logic. In *Proceedings of 7th International Conference on Logic for Programming and Automated Reasoning (LPAR 2000)*, volume 1955 of *Lecture Notes in Computer Science*, pages 326–343. Springer, 2000. {45}

[Hum09] Britta Hummel. *Description Logic for Scene Understanding at the Example of Urban Road Intersections.* PhD thesis, Karlsruhe Institute of Technology, 2009. {28, 29, 30}

[Int09] International Association of Athletics Federations IAAF. The IAAF website http://www.iaaf.org, February 2009. {106, 114, 162}

[JB08] Y. Jing and S. Baluja. PageRank for Product Image Search. In *Proceedings of 17th International World Wide Web Conference WWW 2008*, April 2008. {13, 14}

[KB96] S. Khoshafian and A. B. Baker. *Multimedia and Imaging Databases.* Morgan Kaufmann Publ. Inc., 1996. {171}

[KD93] A. C. Kakas and M. Denecker. Abductive Logic Programming. *Journal of Logic and Computation*, 2:719–770, 1993. {76, 82}

[KKT98] A. Kakas, R. Kowalski, and F. Toni. The Role of Abduction in Logic Programming. *Handbook of Logic in Artificial Intelligence and Logic Programming*, 5:235–324, 1998. {76, 77}

[KLSG03] B. Katz, J. Lin, C. Stauffer, and E. Grimson. Answering Questions About Moving Objects in Surveillance Videos. In *Proceedings of AAAI Spring Symposium on New Directions in Question Answering*, March 2003. {15}

[KMR04] H. Knublauch, M. A. Musen, and A. L. Rector. Editing Description Logic Ontologies with the Protégé OWL Plugin. In *International Workshop on Description Logics*, 2004. {49}

[Kow74] R. Kowalski. Predicate Logic as a Programming Language. In *Proceedings of the IFIP-74*, pages 569–574. Elsevier/North-Holland, 1974. {56}

[Kow79a] R. Kowalski. Algorithm = Logic + Control. In *Communications of the Association for Computing Machinery*, volume 22(7), pages 424–436, 1979. {56}

[Kow79b] R. Kowalski. *Logic for Problem Solving.* Elsevier/North-Holland, New York, 1979. {56}

[KPH05] A. Kalyanpur, B. Parsia, and J. A. Hendler. A Tool for Working with Web Ontologies. *International Journal on Semantic Web and Information Systems*, pages 36–49, 2005. {49}

[KS86] R. Kowalski and M. J. Sergot. A Logic-Based Calculus of Events. *New Generation Computing*, 4:67–95, 1986. {81}

[Len02] M. Lenzerini. Data Integration: A Theoretical Perspective. In *Proceedings of Symposium on Principles of Database Systems*, pages 233–246, 2002. {65}

[LGMR91] P. A. Longley, M. F. Goodchild, D. J. Maguire, and D. W. Rhind. *Geographic Information Systems and Science.* Jon Wiley and Sons Ltd., first edition, 1991. {107}

[Low04] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. In *International Journal of Computer Vision*, volume 60(2), pages 91–110, 2004. {14}

[LR98] A. Y. Levy and M.-C. Rousset. Combining Horn Rules and Description Logics in CARIN. In *Artificial Intelligence*, volume 104(1-2), pages 165–209, 1998. {59}

[LS03] B. Leibe and B. Schiele. Interleaved Object Categorization and Segmentation. In *Proceedings of British Machine Vision Conference (BMVC'03)*, September 2003. {14}

[LST87] J. W. Lloyd, E. A. Sonenberg, and R. W. Topor. Integrity Constraint Checking in Stratified Databases. *Journal of Logic Programming*, 4:331–343, December 1987. {52}

[MB87] R. M. MacGregor and R. Bates. The Loom Representation Language. Technical Report ISI/RS-87-188, Information Sciences Institute, University of Southern California, 1987. {25}

[McG03] D. L. McGuinness. Ontologies Come of Age. In *Spinning the Semantic Web*, pages 171–194, 2003. {36}

[MH90] T. Matsuyama and V. S. Hwang. *SIGMA: A Knowledge-Based Aerial Image Understanding System.* Perseus Publishing, 1990. {21}

[MHRS06] B. Motik, I. Horrocks, R. Rosati, and U. Sattler. Can OWL and Logic Programming Live Together Ever After? In *Proceedings of the 5th Semantic Web Conference (ISWC-06)*, number 4273 in Lecture Notes in Computer Science, pages 501–514. Springer, 2006. {50, 52}

[Mic09] Sun Microsystems. The Java Servlet 2.5. Specification (JSR-000154) http://jcp.org/aboutjava/communityprocess/mrel/jsr154/index.html, February 2009. {111}

[Min75] M. Minsky. A framework for representing knowledge. In P. H. Winston, editor, *The Psychology of Computer Vision*, pages 211–277. Mc-Graw-Hill, New York, 1975. {49}

[MMH87] J. A. Mulder, A. K. Mackworth, and W. S. Havens. Knowledge Structuring and Constrataint Satisfaction: The Mapsee Approach. Technical Report 87-21, Department of Computer Science, University of British Columbia, Vancouver, Canada, 1987. {17, 20}

[MMSW06] J. Medero, K. Maeda, Stephanie Strassel, and Christopher Walker. An Efficient Approach to Gold Standard Annotation: Decision Points for Complex Tasks. In *Proceedings of 5th international conference on Language Resources and Evaluation (LREC 2006)*, Genoa, Italy, May 2006. {163}

[MN08] R. Möller and B. Neumann. Ontology-based Reasoning Techniques for Multimedia Interpretation and Retrieval. In *Semantic Multimedia and Ontologies : Theory and Applications*. Springer, 2008. {61}

[MP93] M. C. Mayer and F. Pirri. First-order Abduction via Tableau and Sequent Calculi. In *Logic Journal of the IGPL 1*, volume 1, pages 141–155, 1993. {82}

[MP96] M. C. Mayer and F. Pirri. Abduction is not Deduction-in-Reverse. In *Journal of the IGPL*, volume 4(1), pages 95–108, 1996. {90, 91}

[MRS08] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. {12}

[MS05] K. Mikolajczyk and C. Schmid. A Performance Evaluation of Local Descriptors. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, volume 27(10), pages 1615–1630, 2005. {14}

[NM95] U. Nilsson and J. Maluszynski. *Logic, Programming and Prolog*. Jon Wiley and Sons Ltd., 1995. {54}

[NM06] B. Neumann and R. Möller. On Scene Interpretation with Description Logics. In *Cognitive Vision Systems: Sampling the Spectrum of Approaches*, number 3948 in Lecture Notes in Computer Science, pages 247–278. Springer, 2006. {61}

[OWL09] W3C OWL Working Group. *OWL-2 Web Ontology Language: Document Overview*. W3C Recommendation, Available at `http://www.w3.org/TR/owl2-overview/`, 27 October 2009. {50}

[Pau93] G. Paul. Approaches to Abductive Reasoning: An Overview. *AI Review*, 7:109–152, 1993. {67, 82}

[Pei78] C. S. Peirce. Deduction, Induction and Hypothesis. In *Popular Science Monthly 13*, pages 470–482, 1878. {20}

[PGA87] D. Poole, R. Goebel, and R. Aleliunas. Theorist: A Logical Reasoning System for Defaults and Diagnosis. In Nick Cercone and Gordon McCalla, editors, *The Knowledge Frontier: Essays in the Representation of Knowledge*, pages 331–352. Springer, 1987. {20, 23}

[Poo89] D. Poole. Explanation and Prediction: An Architecture for Default and Abductive Reasoning. *Computational Intelligence*, 5(2):97–110, 1989. {20, 23}

[PSHH03] P. F. Patel-Schneider, P. Hayes, and I. Horrocks. OWL Web Ontology Language: Semantics and Abstract Syntax http://www.w3.org/tr/owl-semantics. W3C Candidate Recommendation, August 2003. {37}

[PSS93] P. F. Patel-Schneider and B. Swartout. Description-Logic Knowledge Representation System Specification from the KRSS Group of the ARPA Knowledge Sharing Effort, November 1993. {114}

[PTK$^+$08] S. Petridis, N. Tsapatsoulis, D. Kosmopoulos, V. Gatos, G. Petasis, P. Fragou, V. Karkaletsis, W. Hesseler, K. Baitov, S. Espinosa, S. Melzer, A. Kaya, and S. Perantonis. 2.6 Semantics Extraction from Fused Multimedia Content. Project Deliverable Version 1.0 Final, The BOEMIE Consortium, February 2008. {108}

[QPJ07] G. Qi, J. Z. Pan, and Q. Ji. Extending Description Logics with Uncertainty Reasoning in Possibilistic Logic. In *9th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU-07)*, number 4724 in Lecture Notes in Computer Science, pages 828–839. Springer, 2007. {53}

[Qui68] M. Quillian. Semantic memory. In M. Minsky, editor, *Semantic Information Processing*, pages 227–270. MIT Press, 1968. {49}

[RAKD06] O. Ray, A. Antoniades, A. Kakas, and I. Demetriades. Abductive Logic Programming in the Clinical Management of HIV/AIDS. In G. Brewka, S. Coradeschi, A. Perini, and P. Traverso, editors, *Proceedings of the 17th European Conference on Artificial Intelligence*, volume 141 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2006. {75, 76}

[Rei84] R. Reiter. Towards a Logical Reconstruction of Relational Database Theory. In M. L. Brodie, J. Mylopoulos, and J. W. Schmidt, editors, *On Conceptual Modelling: Perspectives from Artificial Intelligence, Databases, and Programming Languages*, pages 191–233. Springer, 1984. {47}

[RM87] R. Reiter and A. K. Mackworth. The Logic of Depiction. Technical Report 87-24, Department of Computer Science, University of British Columbia, Vancouver, Canada, 1987. {17}

[RM90] R. Reiter and A. K. Mackworth. A Logical Framework for Depiction and Image Interpretation. *Artificial Intelligence*, 41(2):125–155, 1989/90. {17}

[RMS97] T. A. Russ, R. M. MacGregor, and B. Salemi. VEIL: Combining Semantic Knowledge with Image Understanding. In O. Firschein and T.M. Strat, editors, *Radius: Image Understanding for Imagery Intelligence*, pages 409–418, San Francisco, CA, 1997. Morgan Kaufmann. {25}

[Ros05] R. Rosati. On the Decidability and Complexity of Integrating Ontologies and Rules. In *Web Semantics: Science, Services and Agents on the World Wide Web*, volume 3(1), pages 41–60, 2005. {56}

[Ros06] R. Rosati. The Limits and Possibilities of Combining Description Logics and Datalog. In *RULEML '06: Proceedings of the Second International Conference on Rules and Rule Markup Languages for the Semantic Web*, pages 3–4. IEEE Computer Society, 2006. {53}

[RPM$^+$98] T. A. Russ, K. Price, R. M. MacGregor, R. Nevatia, and B. Salemi. VEIL: Research in Knowledge Representation for Computer Vision, Final Report. Technical Report A051143, Information Sciences Institute, University of Southern California, February 1998. {26, 27}

[Sch98] C. Schröder. *Bildinterpretation durch Modellkonstruktion: Eine Theorie zur rechnergestützten Analyse von Bildern.* PhD thesis, University of Hamburg, 1998. {24}

[Sha96] M. P. Shanahan. Robotics and the Common Sense Informatic Situation. In W. Wahlster, editor, *Proceedings of the European Conference on Artificial Intelligence (ECAI 96)*, pages 684–688, Chichester, England, 1996. Wiley. {77, 78}

[Sha99] M. P. Shanahan. The Event Calculus Explained. In M. J. Wooldridge and M. Veloso, editors, *Artificial Intelligence Today*, Lecture Notes in Computer Science, pages 409–430, New York, 1999. Springer. {81}

[Sha00] M. P. Shanahan. Reinventing Shakey. In J. Minker, editor, *Logic-based artificial intelligence*, pages 233–253, New York, 2000. Kluwer Academic. {81}

[Sha03] R. W. Shafer. Rationale and Uses of a Public HIV Drug-Resistance Database. *Journal of Infectious Diseases*, 194:51–58, 2003. {75}

[Sha05] M. P. Shanahan. Perception as Abduction: Turning Sensor Data Into Meaningful Representation. *Cognitive Science*, 1:103–134, 2005. {77, 78, 79, 81}

[SHB07] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis, and Machine Vision.* Thomson Learning, April 2007. {14}

[SLL02] S. Se, D. Lowe, and J. J. Little. Global Localization using Distinctive Visual Features. In *Proceedings of International Conference on Intelligent Robots and Systems (IROS2002)*, pages 226–231, Lausanne, Switzerland, November 2002. {14}

[SP06] E. Sirin and B. Parsia. Pellet System Description. In *Proceedings of the 2006 Description Logic Workshop (DL-2006)*, CEUR Electronic Workshop Proceedings, 2006. {45}

[SP07] E. Sirin and B. Parsia. SPARQL-DL: Sparql query for OWL-DL. In *In Proceedings of the 3rd OWL Experiences and Directions Workshop (OWLED-2007)*, 2007. {48}

[SSS91] M. Schmidt-Schauß and G. Smolka. Attributive Concept Descriptions with Complements. *Artificial Intelligence*, 48(1–26), 1991. {38}

[TBK$^+$06] A.-Y. Turhan, S. Bechhofer, A. Kaplunova, T. Liebig, M. Luther, R. Möller, O. Noppens, P. Patel-Schneider, B. Suntisrivaraporn, and T. Weithöner. DIG 2.0 – Towards a Flexible Interface for Description Logic Reasoners. In B. C. Grau, P. Hitzler, and C. Shankey, editors, *OWL Experiences and Directions 2006*, 2006. {48}

[Tha78] P. R. Thagard. The Best Explenation: Criteria for Theory Choice. *The Journal of Philosophy*, 75(2):76–92, February 1978. {90}

[TLN$^+$08] T.Liebig, M. Luther, O. Noppens, M. Rodriguez, D. Calvanese, M. Wessel, M. Horridge, S. Bechhofer, D. Tsarkov, and E. Sirin. OWLlink: DIG for OWL 2. In *Proceedings of the 5th OWL Experiences and Directions Workshop (OWLED-2008)*, 2008. {48}

[TRP$^+$07] T. Tikwinski, C. Rosche, G. Paliouras, A. Ferrara, A. Kaya, and V. Papastathis. 5.4 Specification of the Architecture. Project Deliverable Version 1.0 Final, The BOEMIE Consortium, April 2007. {106, 109}

[Ull85] J. D. Ullman. Implementation of Logical Query Languages for Databases. In *ACM Transactions on Database Systems*, volume 10(3), pages 289–321, 1985. {56}

[Ull97] J. D. Ullman. Information Integration Using Logical Views. In *Database Theory - ICDT '97, 6th International Conference, Delphi, Greece, January 8-10, 1997, Proceedings*, volume 1186 of *Lecture Notes in Computer Science*, pages 19–40. Springer, 1997. {64}

[USA09] USA Track & Field USATF. The USATF website www.usatf.org, February 2009. {106, 162}

[VJ01] P. Viola and M. Jones. Robust Real-time Object Detection. In *International Journal of Computer Vision*, 2001. {14}

[WM06] M. Wessel and R. Möller. A Flexible DL-based Architecture for Deductive Information Systems. In G. Sutcliffe, R. Schmidt, and S. Schulz, editors, *Proceedings of IJCAR-06 Workshop on Emprically Successful Computerized Reasoning (ESCor)*, pages 92–111, 2006. {45}

[Wor09] World Wide Web Consortium (W3C). The W3C website http://www.w3.org, February 2009. {37}

Index

Curriculum Vitae

Surname	Kaya
First name	Atila
Date of Birth	28. December 1975
Place of Birth	Izmir/Turkey

Education

07/1981 - 06/1986	Şemikler Primary School, Izmir/Turkey
07/1986 - 06/1993	Bornova Anadolu Highschool, Izmir/Turkey
10/1993 - 07/1998	B.Sc. in Mechanical Engineering at the Istanbul Technical University
10/1998 - 05/2001	M.Sc. in Mechatronics at the Hamburg University of Technology
04/2006 - onwards	Ph.D. in Computer Science at the Hamburg University of Technology

Employment

06/2001 - 03/2004	*Software Engineer*, T-Systems Nova GmbH
07/2003 - 03/2006	*Teaching Assistant*, Institute for Software Systems, Hamburg University of Technology
04/2004 - 03/2006	*Software Architect*, T-Systems Enterprise Services GmbH
04/2006 - 02/2009	*Research Assistant*, Institute for Software Systems, TuTech Innovation GmbH
03/2009 - 05/2009	*Research Assistant*, Institute for Software Systems, Hamburg University of Technology
08/2009 - onwards	*IT Consultant & Project Manager*, Vattenfall Europe Information Services GmbH