Formalizing Multimedia Interpretation based on Abduction over Description Logic Aboxes *

Sofia Espinosa Peraldi, Atila Kaya, Ralf Möller

Hamburg University of Technology, Germany,
{sofia.espinosa, at.kaya, r.f.moeller}@tuhh.de

Abstract. The paper describes how interpretations of multimedia documents can be formally derived using abduction over domain knowledge represented in an ontology. The approach uses an expressive ontology specification language, namely description logics in combination with logic programming rules, and formalizes the multimedia interpretation process using a combined abduction and deduction operation. We describe how the observables as well as the space of abducibles can be formally defined. The approach is evaluated using examples from text processing, but can also be applied to interpret content in other modalities.

1 Introduction

Multimedia interpretation can be defined as the process of extracting deep-level semantics from content. Deep-level semantics represent the abstract meaning of the content and they are the result of reasoning processes over background knowledge. Automatic interpretation of multimedia content for information extraction (IE) purposes is becoming a major issue of research due to requirements to reduce costs in the areas of knowledge management and information retrieval. The goal is to automate interpretation based on formal models tailored towards a particular domain.

The basic idea of the multimedia interpretation approach that we pursue in this paper is that content descriptions derived by low-level analysis processes should be supported by constructing one or multiple high-level "explanations". The appraoch has been explored in the EU-funded projects BOEMIE (http://www.boemie.org) and CASAM (http://www.casam-project.eu).

Since the goal is to derive high-level descriptions with *new objects and assertions among new and "old" objects*, deduction alone is not an appropriate formalization. In addition to deduction, an abduction step is required, as has been realized several times in the literature.

- First techniques for finding abductive explanations have been discussed long ago [1, 2]. In [3] Mayer and Pirri present a semantic tableaux for abduction in the context of first-order logic using 'reversed skolomization'. Until now only the formalization of this problem is shown, the algorithms are not practical.

* This paper has been partially supported by the projects BOEMIE and CASAM (FP6-027538 and FP7-217061, respectively).

- in [4] Hobbs et al. formalize text interpretation as abduction. In his work Hobbs also argues that deduction is not enough and describes how high-level text interpretation can be realized as abduction over predicate logic formulae. Since predicate logic reasoning is undecidable in general, and the performance of reasoning engine is rather low (if they terminate for a particular problem), we pursue an approach that uses a less expressive but decidable formalism for which highly optimized reasoners exist (e.g., RacerPro).
- In [5], Shanahan presents a formal theory of robot perception as a form of abduction. In this work, low-level sensor data is transformed into a symbolic representation using first-order logic and abduction is used to derive explanations. Logic-based modeling is used to describe the behavior of procedural programs in [5], however.
- In the context of scene interpretation, recently, Möller and Neumann proposed the use of DLs for the representation of aggregates that can be used by reasoning services as building blocks for the scene interpretation process [6, 7]. However, in these works, description logics are used to theoretically describe multimedia interpretation processes only.

The contribution of this paper is the first declarative way to formalize multimedia interpretation that is directly operational in the sense of executable specifications. The paper describes how the "observables requiring explanation" as well as the "space of abducibles" can be formally defined using logic programming techniques. In contrast to approaches such as [8], which use abduction in the context of rules in logic programming only, or appraoches which deal only with concept abduction or [9], our approach combines existing DL reasoning mechanisms and rules in a coherent framework, i.e., abduction is considered as a new type of non-standard Abox retrieval inference service, which is integrated into an existing DL reasoner. The approach is evaluated using examples from text processing, but can also be applied to interpret multimedia documents in other modalities.

2 Preliminaries

For studying interpretation as abduction over ontologies we focus on the description logic $\mathcal{ALCQ}$. We assume that the reader is familiar with description logics, and we only introduce specific operators necessary for Abox abduction.

2.1 Sequences, Variable Substitutions and Transformations

For the introduction of the interpretation algorithm, we need some additional definitions. A *variable* is a name of the form ?*name* where name is a string of characters from $\{a..z\}$. In the follow definitions, we denote places where variables can appear with uppercase letters.

Let V be a set of variables, and let $\underline{X}, \underline{Y_1}, \ldots, \underline{Y_n}$ be sequences $\langle \ldots \rangle$ of variables from V. $\underline{Z}$ denotes a sequence of individuals. We consider sequences of length 1 or 2 only, if not indicated otherwise, and assume that $(\langle X \rangle)$ is to be

read as (X) and $(\langle X, Y\rangle)$ is to be read as (X, Y) etc. Furthermore, we assume that sequences are automatically flattened. A function *as_set* turns a sequence into a set in the obvious way.

A *variable substitution* $\sigma = [X \leftarrow i, Y \leftarrow j, \ldots]$ is a mapping from variables to individuals. The application of a variable substitution σ to a sequence of variables $\langle X\rangle$ or $\langle X, Y\rangle$ is defined as $\langle\sigma(X)\rangle$ or $\langle\sigma(X), \sigma(Y)\rangle$, respectively, with $\sigma(X) = i$ and $\sigma(Y) = j$. In this case, a sequence of individuals is defined. If a substitution is applied to a variable X for which there exists no mapping $X \leftarrow k$ in σ then the result is undefined. A variable for which all required mappings are defined is called *admissible* (w.r.t. the context).

2.2 Grounded Conjunctive Queries

Let $\underline{X}, \underline{Y_1}, \underline{Y_n}$ be sequences of variables, and let $Q_1, \ldots, Q_n$ denote atomic concept or role descriptions.

A query is defined by the following syntax.

$$\{(\underline{X}) \mid Q_1(\underline{Y_1}), \ldots, Q_n(\underline{Y_n})\}$$

The sequence $\underline{X}$ may be of arbitrary length but all variables mentioned in $\underline{X}$ must also appear in at least one of the $\underline{Y_1}, \cdots, \underline{Y_n}$: $as\_set(\underline{X}) \subseteq as\_set(\underline{Y_1}) \cup \cdots \cup as\_set(\underline{Y_n})$.

Informally speaking, $Q_1(\underline{Y_1}), \ldots, Q_n(\underline{Y_n})$ defines a conjunction of so-called *query atoms* $Q_i(\underline{Y_i})$. The list of variables to the left of the sign $\mid$ is called the *head* and the atoms to the right of are called the query *body*. The variables in the head are called distinguished variables. They define the query result. The variables that appear only in the body are called non-distinguished variables and are existentially quantified.

Answering a query with respect to an ontology Σ means finding admissible variable substitutions σ such that $\Sigma \models \{(\sigma(\underline{Y_1})) : Q_1, \ldots, (\sigma(\underline{Y_n})) : Q_n\}$. We say that a variable substitution $\sigma = [X \leftarrow i, Y \leftarrow j, \ldots]$ introduces *bindings* $i, j, \ldots$ for variables $X, Y, \ldots$. Given all possible variable substitutions σ, the *result* of a query is defined as $\{(\sigma(\underline{X}))\}$

Note that the variable substitution σ is applied before checking whether $\Sigma \models \{(\sigma(\underline{Y_1})) : Q_1, \ldots, (\sigma(\underline{Y_n})) : Q_n\}$, i.e., the query is *grounded* first.

For a query $\{(?y) \mid Person(?x), hasParticipant(?y, ?x)\}$ and the Abox $\Gamma_1 = \{ind_1 : HighJump, ind_2 : Person, (ind_1,\ ind_2) : hasParticipant\}$, the substitution $[?x \leftarrow ind_2, ?y \leftarrow ind_1]$ allows for answering the query, and defines bindings for $?y$ and $?x$.

A *boolean* query is a query with $\underline{X}$ being of length zero. If for a boolean query there exists a variable substitution σ such that $\Sigma \models \{(\sigma(\underline{Y_1})) : Q_1, \ldots, (\sigma(\underline{Y_n})) : Q_n\}$ holds, we say that the query is answered with *true*, otherwise the answer is *false*.

Later on, we will have to convert query atoms into Abox assertions. This is done with the function *transform*. The function *transform* applied to a set of query atoms $\{\gamma_1, \ldots \gamma_n\}$ is defined as $\{transform(\gamma_1, \sigma), \ldots, transform(\gamma_n, \sigma)\}$ where

$transform(P(\underline{X}), \sigma) := (\sigma(\underline{X})) : P$.

2.3 Rules

A rule r has the following form $P(\underline{X}) \leftarrow Q_1(\underline{Y_1}), \ldots, Q_n(\underline{Y_n})$ where P, $Q_1, \ldots, Q_n$ denote atomic concept or role descriptions with the additional restriction (safety condition) that $as\_set(\underline{X}) \subseteq as\_set(\underline{Y_1}) \cup \cdots \cup as\_set(\underline{Y_n})$.

Rules are used to derive new Abox assertions, and we say that a rule r is *applied* to an Abox $\mathcal{A}$. The function call $apply(\Sigma, P(\underline{X}) \leftarrow Q_1(\underline{Y_1}), \ldots, Q_n(\underline{Y_n}), \mathcal{A})$ returns a set of Abox assertions $\{(\sigma(\underline{X})) : P\}$ if there exists an admissible variable substitution σ such that the answer to the query

$$\{() \mid Q_1(\underline{Y_1}), \ldots, Q_n(\underline{Y_n})\}$$

is *true* with respect to $\Sigma \cup \mathcal{A}$.[1] If no such σ can be found, the result of the call to $apply(\Sigma, r, \mathcal{A})$ is the empty set. The application of a set of rules $\mathcal{R} = \{r_1, \ldots r_n\}$ to an Abox is defined as follows.

$$apply(\Sigma, \mathcal{R}, \mathcal{A}) = \bigcup_{r \in \mathcal{R}} apply(\Sigma, r, \mathcal{A})$$

The result of $forward\_chain(\Sigma, \mathcal{R}, \mathcal{A})$ is $\emptyset$ if $apply(\Sigma, \mathcal{R}, \mathcal{A}) \cup \mathcal{A} = \mathcal{A}$ and $apply(\Sigma, \mathcal{R}, \mathcal{A}) \cup forward\_chain(\Sigma, \mathcal{R}, \mathcal{A} \cup apply(\Sigma, \mathcal{R}, \mathcal{A}))$ otherwise.

3 Multimedia Interpretation

The multimedia interpretation process aims to compute interpretations (sets of descriptions) of a multimedia document based on low-level descriptions and background knowledge. Beside low-level descriptions about objects and their relations, an interpretation also contains high-level descriptions about abstract objects like events and their relations with low-level descriptions. High-level descriptions cannot directly be inferred from low-level descriptions but they have to be hypothesized w.r.t. some background knowledge.

In the rest of this section, we first formalize abduction as the key inference service for explaining observations, and we then describe the multimedia interpretation process that exploits both abduction and deduction to compute interpretations.

3.1 Computing Explanations via Abduction

In general, abduction is formalized as $\Sigma \cup \Delta \models \Gamma$ where background knowledge (Σ), and observations (Γ) are given and explanations (Δ) are to be computed. In terms of DLs, Δ and Γ are Aboxes and Σ is a TBox.

Abox abduction is implemented as a non-standard retrieval inference service in DLs, in contrast to standard retrieval inference services where answers are found by exploiting the ontology, Abox abduction has the task of acquiring what should be added to the ontology in order to answer a query. Therefore, the

[1] If $\Sigma \cup \mathcal{A}$ is inconsistent the result is well-defined but useless. It will not be used afterwards.

result of Abox abduction is a set of hypothesized Abox assertions. To achieve this, the space of abducibles has to be previously defined and we do this in terms of rules.

We assume that a set of rules $\mathcal{R}$ as defined above (see Section 2.3) are specified, and define a non-deterministic function *compute_explanation* as follows.

- $compute\_explanation(\Sigma, \mathcal{R}, \mathcal{A}, (\underline{Z}) : P) = transform(\Phi, \sigma)$ if there exists a rule $r = P(\underline{X}) \leftarrow Q_1(\underline{Y_1}), \ldots, Q_n(\underline{Y_n}) \in \mathcal{R}$ that is applied to an Abox $\mathcal{A}$ such that a set of query atoms Φ and an admissible variable substitution σ with $\sigma(\underline{X}) = \underline{Z}$ can be found, and the query $Q := \{() \mid expand(P(\underline{X}), r, \mathcal{R}) \setminus \Phi\}$ is answered with *true*.
- If no such rule r exists in $\mathcal{R}$ it holds that $compute\_explanation(\Sigma, \mathcal{R}, \mathcal{A}, (\underline{Z}) : P) = \emptyset$.

The goal of the function *compute_explanation* is to determine what must be added (Φ) such that an entailment $\Sigma \cup \mathcal{A} \cup \Phi \models (\underline{Z}) : P$ holds. Hence, for *compute_explanation*, abductive reasoning is used. The set of query atoms Φ defines what must be hypothesized in order to answer the query Q with *true* such that $\Phi \subseteq expand(P(\underline{X}), r, \mathcal{R})$ holds. The definition of *compute_explanation* is non-deterministic due to several possible choices for Φ.

The function application $expand(P(\underline{X}), P(\underline{X}) \leftarrow Q_1(\underline{Y_1}), \ldots, Q_n(\underline{Y_n}), \mathcal{R})$ is also defined in a non-deterministic way as

$$expand'(Q_1(\underline{Y_1}), \mathcal{R}) \cup \cdots \cup expand'(Q_n(\underline{Y_n}), \mathcal{R})$$

with $expand'(P(\underline{X}), \mathcal{R})$ being $expand(P(\underline{X}), r, \mathcal{R})$ if there exist a rule $r = P(\underline{X}) \leftarrow \ldots \in \mathcal{R}$ and $\langle P(\underline{X}) \rangle$ otherwise. We say the set of rules is backward-chained, and since there might be multiple rules in $\mathcal{R}$, backward-chaining is non-deterministic as well. Thus, multiple explanations are generated.

3.2 The Media Interpretation Process

In the following we devise an abstract computational engine for "interpreting" Abox assertions in terms of a given set of rules. Interpretation in this sense is not to be confused with the interpretation of a concept description (which is defined as a set of objects from the domain). Interpretation of Abox assertions w.r.t. a set of rules is meant in the sense that using the rules some high-level explanation is constructed such that the Abox assertions are entailed. The interpretation of an Abox is again an Abox. For instance, the output Abox might represent results of a content interpretation process (see below for an example). The presentation in this paper slghtly extended the one in [10].

Let Γ be an Abox of observations whose assertions are to be explained. The goal of the interpretation process is to use a set of rules $\mathcal{R}$ to derive "explanations" for elements in Γ. The interpretation algorithm implemented in the interpretation engine works on a set of (possible) interpretations $\mathfrak{I}$, i.e., a set of Aboxes.

Initially, $\mathfrak{I} \Leftarrow \{\Gamma\}$, i.e. $\{(pName_1,\ country_1) : personNameToCountry,$ $(hjName_1,\ city_1) : sportsNameToCity\}$, at this stage, the interpretation is

just the input Abox Γ.[2] The complete multimedia interpretation process is implemented by the *interpret* function:

function $interpret(\Omega, \Xi, \Sigma, \mathcal{R}, S, \Gamma)$:
  $\mathfrak{I}' := \{\Gamma\}$
  repeat
    $\mathfrak{I} := \mathfrak{I}'$
    $(\mathcal{A}, \alpha) := \Omega(\mathfrak{I})$ // $\mathcal{A} \in \mathfrak{I}$, $\alpha \in \mathcal{A}$ s.th. $requires\_fiat(\alpha)$ holds
    $\mathfrak{I}' := (\mathfrak{I} \setminus \{\mathcal{A}\}) \cup interpretation\_step(\Sigma, \mathcal{R}, S, \mathcal{A}, \alpha)$.
  until $\Xi(\mathfrak{I})$ or no $\mathcal{A}$ and α can be selected such that $\mathfrak{I}' \neq \mathfrak{I}$
  return $\mathfrak{I}$

It takes as parameters a strategy function Ω, a termination function Ξ, a background knowledge Σ, a set of rules $\mathcal{R}$, a scoring function S and an Abox Γ of observations. It applies the strategy function Ω in order to decide which assertion to interpret, uses a termination function Ξ in order to check whether to terminate due to resource constraints and a scoring function S to evaluate an explanation.

The function Ω for the interpretation strategy and Ξ for the termination condition are used as an oracle and must be defined in an application-specific way. In our multimedia interpretation scenario we assume that $requires\_fiat$ function is defined in an application-specific way. The function *interpretation_step* is defined as follows.

$interpretation\_step(\Sigma, \mathcal{R}, S, \mathcal{A}, \alpha)$:

$$\bigcup_{\Delta \in compute\_all\_explanations(\Sigma, \mathcal{R}, S, \mathcal{A}, \alpha)} consistent\_completed\_explanations(\Sigma, \mathcal{R}, \mathcal{A}, \Delta).$$

We need two additional auxiliary functions.

$consistent\_completed\_explanations(\Sigma, \mathcal{R}, \mathcal{A}, \Delta)$:

$$\{\Delta' \mid \Delta' = \Delta \cup \mathcal{A} \cup forward\_chain(\Sigma, \mathcal{R}, \Delta \cup \mathcal{A}), consistent_{\Sigma}(\Delta')\}$$

$compute\_all\_explanations(\Sigma, \mathcal{R}, S, \mathcal{A}, \alpha)$:

$$maximize(\{\Delta \mid \Delta = compute\_explanation(\Sigma, \mathcal{R}, \mathcal{A}, \alpha)\}, S).$$

The function $consistent_{(\mathcal{T}, \mathcal{A}_0)}(\mathcal{A})$ determines if the Abox $\mathcal{A} \cup \mathcal{A}_0$ has a model which is also a model of the Tbox $\mathcal{T}$.

Depending on the application context, some of the observations can be taken for granted (bona-fide assertions) whereas others as requiring explanations (we call them fiat assertions for brevity). The $requires\_fiat$ function is used to split Γ into bona-fide assertions Γ_1 and fiat assertions Γ_2. Assertions for which the $requires\_fiat$ function returns true constitute Γ_2, whereas $\Gamma_1 = \Gamma \setminus \Gamma_2$.

We impose restrictions on the choice of the explanations (Δs) computed during the interpretation process. In particular, a scoring function S evaluates an

[2] $\Leftarrow$ denotes the assignment operator

explanation Δ according to the two criteria proposed by Thagard for selecting explanations [11], namely simplicity and consilience. According to Thagard, the less hypothesized assertions an explanation contains (simplicity) and the more ground assertions (observations) an explanation involves (consilience), the higher its preference score. The following function can be used to compute the preference score for a given explanation[3]: $S(\Sigma, \Gamma_1, \Delta) := S_f(\Sigma, \Gamma_1, \Delta) - S_h(\Delta)$. The function S_f represents the number of assertions in the explanation (Δ) that follow from $\Sigma \cup \Delta$, and the function S_h represents the number of assertions in the explanation. Thus, S_f and S_h can be defined as follows:

$$S_f(\Sigma, \Gamma_1, \Delta) := \sharp\{\alpha \in \Gamma_1 \mid \Sigma \cup \Delta \models \alpha\}$$
$$S_h(\Delta) := \sharp\{\alpha \in \Delta\}$$

The function *maximize* selects those explanations (Δs) for which the score $S(\Sigma, \Gamma_1, \Delta)$ is maximal, i.e., there exists no other $\Delta' \in \mathfrak{I}$ such that $S(\Sigma, \Gamma_1, \Delta') > S(\Sigma, \Gamma_1, \Delta)$.

4 An Interpretation Example

We proceed with the interpretation of a text excerpt to discuss the details of the interpretation process. Figure 1 shows a text from a web page with athletics news, the underlined words are tokens which have to be detected by text analysis processes. The extraction result obtained by using the BOEMIE shallow text processing technology is depicted in Figure 2 as an Abox. In Figure 3 a small part of the ontology (Σ) relevant for our discussion is shown. Additionally, a small excerpt of the set of rules ($\mathcal{R}$) used for the interpretation of texts about the athletics example is shown in Figure 4. The rules shown in Figure 4 define the space of abducibles.

'13 August 2002 - Helsinki. Russia's newly crowned European champion Jaroslav Rybakov won the high jump with 2.29 m. Oskari Fronensis from Finland cleared 2.26 and won silver.'

Fig. 1. A small excerpt from a web page.

The Abox in Figure 2 constitutes the set of observations Γ for the *interpret* function (see Section 3.2). The strategy function Ω selects the assertions that are fiat and therefore require the computation of explanations. In the current implementation, the strategy function selects all binary predicates shown in Figure 2 in the beginning of the process. Next, each assertion selected by the strategy function is transformed into a corresponding query and the abductive retrieval inference service is asked for explanations. For example, from the role assertion $(hjName_1, date_1) : sportsNameToDate$ the following query is derived:

[3] For the sake of brevity the parameters of S are not shown in the previous functions.

$$
\begin{array}{rl}
date_1 : Date & (date_1,\ \text{`13 } August\ 2002'\text{)} : hasValue \\
city_1 : City & (city_1,\ \text{`}Helsinki\text{'}) : hasValue \\
country_1 : Country & (country_1,\ \text{`}Russia\text{'}) : hasValue \\
country_2 : Country & (country_2,\ \text{`}Finland\text{'}) : hasValue \\
perf_1 : Performance & (perf_1,\ \text{`2.29'}) : hasValue \\
perf_2 : Performance & (perf_2,\ \text{`2.26'}) : hasValue \\
rank_1 : Ranking & (rank_1,\ \text{`}silver\text{'}) : hasValue \\
hjName_1 : HighJumpName & (hjName_1,\ \text{`high jump'}) : hasValue \\
pName_2 : PersonName & (pName_1,\ \text{`}Jaroslav\ Rybakov\text{'}) : hasValue \\
pName_1 : PersonName & (pName_2,\ \text{`}Oskari\ Fronensis\text{'}) : hasValue \\
(pName_1,\ country_1) : personNameToCountry & \\
(pName_2,\ country_2) : personNameToCountry & \\
(pName_1,\ perf_1) : personNameToPerformance & \\
(pName_2,\ perf_2) : personNameToPerformance & \\
(hjName_1,\ perf_1) : sportsNameToPerformance & \\
(hjName_1,\ date_1) : sportsNameToDate & \\
(hjName_1,\ city_1) : sportsNameToCity &
\end{array}
$$

Fig. 2. Abox representing the results of text analysis (Γ).

$$
\begin{array}{rl}
Person & \sqsubseteq \exists hasName.PersonName \sqcap \exists hasNationality.Country \\
Athlete & \sqsubseteq Person \\
HighJumper & \sqsubseteq Athlete \\
PoleVaulter & \sqsubseteq Athlete \\
HighJumpName & \sqsubseteq SportsName \sqcap \neg PoleVaultName \\
PoleVaultName & \sqsubseteq SportsName \\
SportsTrial & \sqsubseteq \exists_{\leq 1} hasParticipant.Athlete \sqcap \exists_{\leq 1} hasPerformance.Performance \sqcap \\
 & \quad \exists_{\leq 1} hasRanking.Ranking \\
HighJump & \sqsubseteq SportsTrial \sqcap \forall hasParticipant.HighJumper \sqcap \neg PoleVault \\
PoleVault & \sqsubseteq SportsTrial \sqcap \forall hasParticipant.PoleVaulter \\
SportsRound & \sqsubseteq \exists hasName.RoundName \sqcap \exists hasDate.Date \sqcap \exists hasPart.SportsTrial \\
HighJumpRound & \sqsubseteq SportsRound \sqcap \forall hasPart.HighJump \sqcap \neg PoleVaultRound \\
PoleVaultRound & \sqsubseteq SportsRound \sqcap \forall hasPart.PoleVault \\
SportsCompetition & \sqsubseteq \exists hasPart.SportsRound \sqcap \exists hasName.SportsName \sqcap \exists takesPlace.City \\
HighJumpCompetition & \sqsubseteq SportsCompetition \sqcap \forall hasPart.HighJumpRound \sqcap \\
 & \quad \forall hasName.HighJumpName \sqcap \neg PoleVaultCompetition \\
PoleVaultCompetition & \sqsubseteq SportsCompetition \sqcap \forall hasPart.PoleVaultRound \sqcap \\
 & \quad \forall hasName.PoleVaultName
\end{array}
$$

Fig. 3. An example Tbox for the athletics domain (see http://www.boemie.org/ for an extended version).

$$Q_1 := \{() \mid sportsNameToDate(hjName_1,\ date_1)\}$$

In the given set of rules $\mathcal{R}$ (see Figure 4), two rules have the predicate *sportsNameToDate* in the rule head. Therefore, both rules are applied in a backward chaining way (i.e. from left to right) and corresponding terms are unified and we get variable bindings for X and Y. The unbound variable Z is instantiated with a fresh individual (e.g. *new_ind_1*). Notice that for one of these rules, namely for the one that hypothesizes a pole vault competition, all bindings that are found for Y produce explanations (Δs) that are inconsistent w.r.t. Σ. This is caused by the disjointness axioms in the Tbox (e.g. the concepts *HighJumpName* and *PoleVaultName* are disjoint). The abductive retrieval ser-

$$
\begin{aligned}
personNameToCountry\,(X,Y) \leftarrow\ & Person(Z),\\
& hasPersonName(Z,X), PersonName(X),\\
& hasNationality(Z,Y), Country(Y).\\
sportsNameToDate\,(X,Y) \leftarrow\ & HighJumpCompetition(Z),\\
& hasSportsName(Z,X), HighJumpName(X),\\
& hasDate(Z,Y), Date(Y).\\
sportsNameToDate\,(X,Y) \leftarrow\ & PoleVaultCompetition(Z),\\
& hasSportsName(Z,X), PoleVaultName(X),\\
& hasDate(Z,Y), Date(Y).\\
sportsCompetitionToPerformance\,(X,Y) \leftarrow\ & SportsCompetition(X),\\
& hasSportsName(X,Z), SportsName(Z),\\
& sportsNameToPerformance(Z,Y).\\
sportsCompetitionToPerformance\,(X,Y) \leftarrow\ & SportsCompetition(X),\\
& hasPart(X,Z), SportsRound(Z),\\
& hasPart(Z,W), SportsTrial(W)\\
& hasPerformance(W,Y).
\end{aligned}
$$

Fig. 4. Text interpretation rules for the athletics domain ($\mathcal{R}$).

vice discards inconsistent explanations. Therefore, the generated explanation to answer Q_1 with true is:

$$
\begin{aligned}
\Delta = \{&new\_ind_1 : HighJumpCompetition,\ (new\_ind_1,\ hjName_1) : hasSportsName,\\
&(new\_ind_1,\ date_1) : hasDate\}
\end{aligned}
$$

The assertions in Δ are then added to the Abox $\mathcal{A}$ and the rules are applied in a forward chaining way to find out whether new assertions can be inferred and therefore have to be added to $\mathcal{A}$ as well. In our example there are no new assertions that can be inferred at this stage.

The same procedure is applied to every assertion selected by the strategy function, until all assertions considered as requiring fiat are explained. Finally, an Abox with the following assertions are returned by the interpretation process:

$$
\begin{aligned}
&new\_ind_1 : HighJumpCompetition,\ new\_ind_2 : Person, new\_ind_3 : Person,\\
&new\_ind_4 : SportsRound,\ new\_ind_5 : SportsTrial, new\_ind_6 : SportsTrial,\\
&(new\_ind_1,\ hjName_1) : hasSportsName,\ (new\_ind_1,\ date_1) : hasDate,\\
&(new\_ind_1,\ new\_ind_4) : hasPart,\ (new\_ind_5,\ new\_ind_2) : hasParticipant,\\
&(new\_ind_4,\ new\_ind_5) : hasPart,\ (new\_ind_5,\ perf_1) : hasPerformance,\\
&(new\_ind_6,\ new\_ind_3) : hasParticipant,\ (new\_ind_6,\ perf_2) : hasPerformance\\
&(new\_ind_2,\ pName_1) : hasPersonName,\ (new\_ind_2,\ country_1) : hasNationality,\\
&(new\_ind_3,\ pName_2) : hasPersonName,\ (new\_ind_3,\ country_2) : hasNationality
\end{aligned}
$$

which represents the semantic description (interpretation) of the multimedia content.

Notice that if interpretations of documents in a multimedia repository are available as Aboxes, DL reasoners can be used to realize semantic multimedia retrieval. For example assume that a query for high jump trials $Q_2 := \{(?X) \mid HighJumpTrial(?X)\}$ is posed. A DL Reasoner (e.g. RacerPro) would return new_ind_5 as a binding for $?X$, even though this information is not explicit in the Abox. Informally speaking, a high jump competition can only have high

jump rounds as parts and a high jump round can only have high jump trials as parts (see the definitions of the concepts $HighJumpCompetition$ and $HighJumpRound$ in the ontology depicted in Figure 3). Therefore it is certain that new_ind_5 has to be a $HighJumpTrial$ instance in all possible worlds. This simple example shows that DL reasoners are useful tools for detecting implicit information in multimedia interpretations and thus can be used to realize semantic multimedia retrieval.

5 Evaluation

In this section, we present an evaluation of an implementation of the multimedia interpretation process presented in Section 3. The utility of the abduction-based multimedia interpretation process is analyzed by means of precision, recall and F-measure through an empirical evaluation of the results obtained for a corpus of web pages.

Experimental Setting and Criteria To test the approach, an ontology about the athletics domain was used as well as a corpus of 104 web pages each, containing daily news about athletics events.

The corpus has been annotated manually with state-of-the-art annotation tool [12]. The manual annotation process has been accomplished in two steps: First, words in the text have been associated with corresponding concepts in the ontology. These concepts are: *PersonName*, *Country*, *City*, *Age*, *Gender*, *Performance*, *Ranking*, *SportsName*, *RoundName*, *Date* and *EventName*. They are called mid-level concepts (MLCs) in the BOEMIE project. Second, the text segments annotated with mid-level concepts as labels are grouped and each group is associated with a high-level concept (HLC) such as *Athlete*, *SportTrial*, *SportsRound*, *SportEvents* and *SportsCompetition*. The outcome of the manual annotation process is a set of annotations for the corpus. They contain not only low-level but also high-level descriptions of the content and serve as ground truth for the evaluation.

Later the set of manually obtained annotations have been used to train the text analysis tools in order to automatically extract concept instances as well as relations between the instances from the corpus. The results of analysis, which are analysis Aboxes, have been automatically interpreted following the multimedia interpretation approach presented. As a result, a set of automatically computed annotations for the same corpus is obtained. This set of annotations, which represent the results of automatic analysis and interpretation of the corpus, will be evaluated in this section.

To set up the evaluation, a set of queries has been defined in order to ask for the number of HLCi (high-level concept instances) in both manual and automatically computed annotations. In this way, names of high-level concepts constitute the parameters to evaluate the precision and recall of the multimedia interpretation framework.

Evaluation results Table 1 shows the results of the experiments conducted. The letters M and AC represent manual and automatically computed annotations respectively. The results in Table 1 show that most of the manually annotated high-level concepts (explanations) could also be computed automatically. There are exceptional cases in which no explanations have been computed due to the lack of necessary rules or lack of low-level text analysis results. For example, in the case of *SportsRound*, the interpretation process expects input about *SportsRoundName* and *Date* instances in the relation called *sportsRoundNameToStartDate*, but this structures are rarely found during text analysis. Therefore, the rule necessary to create instances of *SportsRound* is never applied.

Table 1. Evaluation of the multimedia interpretation approach for the text modality

HLC	M	AC	$M \cap AC$	**Precision**	**Recall**	**F-Measure**
Athlete	783	591	496	0.84	0.63	0.72
SportsTrial	729	641	513	0.80	0.70	0.74
SportsCompetition	443	200	188	0.94	0.43	0.80
SportsEvent	304	304	266	0.87	0.87	0.87
SportsRound	375	0	0	0	0	0

Furthermore, we observed that from a total of 200 extracted *SportCompetition*s, five have been further specialized to $HighJumpCompetition$s and ten to $PoleVaultCompetition$s. This is due to low-level analysis results where instances of the concepts $HighJumpName$ and $PoleVaultName$ have been found. As observed in Figure 3 the definition of a $HighJumpCompetition$ ($PoleVaultCompetiton$) requires a $HighJumpName$ ($PoleVaultName$) as a range restriction for the role $hasName$.

6 Conclusions

In this paper we have presented the first declarative way to formalize multimedia interpretation based on abduction that is directly operational in the sense of executable specifications. The interpretation engine specified in this paper is evaluated empirically.

We have shown that given the results of state-of-the-art multimedia analysis tools as low-level descriptions, the whole multimedia interpretation process can be realized by exploiting declarative domain models and high-optimized deductive and abductive reasoning services. We have discussed how existing DL reasoning mechanisms and rules can be combined in a coherent framework. We have also evaluated the approach using examples from the text modality and have shown that the current implementation provides promising results for web pages with news about athletics events.

References

1. Aliseda-Llera, A.: Seeking Explanations: Abduction in Logic, Philosophy of Science and Artifical Intelligence. PhD thesis, University of Amsterdam (1997)
2. Paul, G.: Approaches to Abductive Reasoning–An Overview. AI Review 7 (1993) 109–152
3. Mayer, M.C., Pirri, F.: First-Order Abduction via Tableau and Sequent Calculi. Bulletin of the IPGL **1**(1) (1993) 99–117
4. Hobbs, J.R., Stickel, M., Appelt, D., Martin, P.: Interpretation as abduction. Artificial Intelligence Journal **Vol. 63** (1993) 69–142
5. Shanahan, M.: Perception as Abduction: Turning Sensor Data Into Meaningful Representation. Cognitive Science **29**(1) (2005) 103–134
6. Möller, R., Neumann, B.: Ontology-based Reasoning Techniques for Multimedia Interpretation and Retrieval. In: Semantic Multimedia and Ontologies : Theory and Applications. Springer (2008)
7. Neumann, B., Möller, R.: On Scene Interpretation with Description Logics. In Christensen, H., Nagel, H.H., eds.: Cognitive Vision Systems: Sampling the Spectrum of Approaches. Number 3948 in LNCS. Springer (2006) 247–278
8. Kakas, A., Denecker, M.: Abduction in logic programming. In Kakas, A., Sadri, F., eds.: Computational Logic: Logic Programming and Beyond. Part I. Number 2407 in LNAI. Springer (2002) 402–436
9. Colucci, S., Di Noia, T., Di Sciascio, E., Donini, F.M., Mongiello, M.: A uniform tableaux-based method for concept abduction and contraction in description logics. In: Proc. of the 16th Eur. Conf. on Artificial Intelligence (ECAI 2004). (2004) 975–976
10. Castano, S., Espinosa, S., Ferrara, A., Karkaletsis, V., Kaya, A., Möller, R., Montanelli, S., Petasis, G., Wessel, M.: Multimedia interpretation for dynamic ontology evolution. In: Journal of Logic and Computation, Oxford University Press (2008)
11. Thagard, R.P.: The best explanation: Criteria for theory choice. The Journal of Philosophy (1978)
12. Petasis, G., Karkaletsis, V., Paliouras, G., Androutsopoulos, I., Spyropoulos, C.D.: Ellogon: A new text engineering platform (2002)