TUHH

Technische Universität Hamburg-Harburg

# Evaluation of Fusion Operators

Institute for Software Systems

Project Work by Jesus Martinez Caudillo

1st Supervisor: Prof. Dr. rer. nat. habil. Ralf Möller

2st Supervisor: Dipl.-Ing. Sylvia Melzer

Hamburg, 21st June 2013

# Abstract

Information Retrieval Systems which are used to retrieve multimedia documents perform a set of assertions of queries based on previous knowledge and retrieve related documents following the perspective of similar content description of such documents.

The term multimedia documents is referred to those text formated files containing images and video. In order to identify which documents are more similar to others it is necessary to focus on the content of the documents and provide a representation model which provides a set of features to distinguish the similarities and differences among a given collection of documents. There are two main content representation formats, holistic content description and symbolic content description.

Holistic content description is based on similarity measures and it matches the content based, for example, using nearest-neighbor algorithms [MRS08]. On the other hand symbolic content description are based on multimedia elements (images, video, caption and text) which are represented by symbolic content descriptions.

In this project work symbolic content descriptions will be used. The creation of symbolic content descriptions has three sub-processes: analysis, interpretation and fusion. The analysis process consists in generating the symbolic content descriptions using low-level annotation tools. For example, the mapping of an image results in instances and concepts related with each other, these relations are called assertions and the set of assertions is called Abox.

The interpretation process, also called high-level interpretation process, computes interpretations using the Abox abduction algorithm [Mel12]. The abduction algorithm is used to find explanations (causes) for observations (effects). Concept definitions and inclusion axioms are grouped in a Tbox. A Tbox contains intentional knowledge in the form of a terminology and DL-safe rules [Kay11].

Consider forward chaining one of these rules, where a query it is determined to be entailed by a knowledge base of clauses [RN03]. Backward chaining is another rule where the query is implied to be true and the task consists of finding the set of clauses whose implications prove the query to be true. The result of the interpretation is the difference operation of two Aboxes which is another Abox. Another approach is ABox-difference operation where the result is based on the elements in common that two ABoxes have.

Finally the fusion process merges outputs of the interpretation process. Each interpretation Abox resulting of the previous process contains assertions of which can be related with other resulting instances. The fusion consists of relating these Aboxes which refer to the same assertions

The aim of this project work is to evaluate the result of the ABox-difference operation (fusion

algorithm) using the BOEMIE repository (Bootstrapping Ontology Evolution with Multimedia Information Extraction) according to its precision, speedness and performance. The document corpora contains topics of sports events in european cities, the precision of evaluation must be keen since it is likely to have many documents that seem to be identical, therefore the precision is desirable.

Precision is defined as the fraction of retrieved documents that are relevant whereas recall is the fraction of relevant documents that are retrieved [KB96]. The fusion process will be evaluated by comparing its performance with respect to documents with high precision and recall. It is expected to get high quality documents by enhancing high precision and recall on the retrieved documents.

# Contents

# 1 Goals

The aim of this work is to evaluate of the fusion algorithm on multiple multimedia documents from the BOEMIE repository. The results of this project work, will show the viability of the use in terms of computational work of the fusion algorithm.

# 2 Description Logics and Fusion Process

2.1 Multimedia documents

Detecting information in a web page can be difficult if we try to go consciously deeply by looking whether each image contains a person or not, however it it is easy if we look for a name of a person in a text. The task of computing interpretations of multimedia documents can be called multimedia interpretation. [Kay11]

Multimedia documents have embedded video, images, caption text and interactive links (urls) which represent data, one typical example is a web page. Multimedia interpretation processes are used to identify the information contained in documents independently of the systems used for it. [Kay11]

2.2 Symbolic content descriptions

*Symbolic content descriptions* are used to identify the multiple multimedia elements of a document.

A symbolic content description associates *semantic descriptions* with complete images or individual image objects. [Bos08]. In a document where an image of a person is presented, many semantic interpretations can be linked in to. Figure 1 represents a piece of multimedia document, such document contains three multimedia objects. Now when performing a given query it can be specified in such a way that it is returning results in a part of the multimedia document. For instance, if a query like "Kajsa Berqvist images" is preferable to narrow the search only within the symbolic content description "*image"* of this document.


Figure 1 Multimedia document which contains three multimedia objects: image, caption and text.

## 2.3 Description Logics

In order to create content description of multimedia documents, there must be a way of representation of content, the so called *Description Logics (DL)*. Description Logics is the most recent name for knowledge representation formalisms that represent the knowledge of an application domain (the "world") by first defining relevant concepts of the domain (its terminology), and the using these concepts to specify properties of objects and individuals occurring in the domain (the world description) [BN03].

DL makes use basically of logic-based semantics which allows to infer implicit knowledge from a knowledge base. DL provides concepts and relations as structure of this knowledge base. Concepts can have simple properties often called attributes. [NB03].

2.3.1 Syntax and Semantics

Some of the syntax in DL are:

| Syntax | Constructor |
|---|---|
| $A$ | atomic concept |
| $C \sqcap D$ | conjunction |
| $C \sqcup D$ | disjunction |
| $\neg C$ | negation |
| $\exists R.C$ | existential restriction |
| $\forall R.C$ | value restriction |
| $\exists \leq_n R.C$ | qualified minimum restriction |
| $\exists \geq_n R.C$ | qualified maximum restriction |

Table 1. Constructors for building complex concepts. Melzer[34]

For example, the expression:

$$\mathrm{Person} \sqcap \exists \mathrm{hasRanking.Ranking}$$

From the athletics domain $\mathrm{Person}$ and $\mathrm{Ranking}$ are atomic concepts, $\mathrm{hasRanking}$ is called atomic role and $\exists \mathrm{hasRanking.Ranking}$ is a complex concept.

A knowledge base comprises two components, a *TBox* and a *ABox*. Nardi and Brachman refer to TBox as the *intentional knowledge* and ABox *extensional knowledge* while Baader and Nutt state that a TBox introduces *terminology* and an ABox contains *assertions*. Thus, we can resume that a TBox contains a set of concept definitions for a given application domain and ABox are assertions to specific individuals (*instances*) of the application domain

A generalized *concept inclusion axiom* (GCI) is a terminological axiom of the from:

$C \sqsubseteq D$

where $C$ and $D$ are concepts. An interpretation $\mathfrak{J}$ satisfies a CGI axiom $C^{\mathfrak{J}} \sqsubseteq D^{\mathfrak{J}}$ if $C \sqsubseteq D$. An interpretation $\mathfrak{J}$ is a model of a TBox $\mathcal{T}$ if it satisfies all axioms in $\mathcal{T}$. For example $\mathrm{Athlete} \sqsubseteq \mathrm{Person}$ states that the concept $\mathrm{Athlete}$ is a specialization of the concept $\mathrm{Person}$, and thus every domain object in the interpretation of the concept $\mathrm{Athlete}$ must also be in the interpretation of the concept $\mathrm{Person}$ [Kay11]

An example of TBox is:

$$\mathrm{Person} \equiv \mathrm{PersonBody} \sqcap \mathrm{PersonFace}$$

which intentionally states the equality of the symbolic content description $\mathrm{Person}$ and other two $\mathrm{PersonBody} \sqcap \mathrm{PersonFace}$. $C \equiv D$ is another terminological axiom, also known as concept definition used as an abbreviation for two CGI axioms $C \sqsubseteq D$ and $C \sqsupseteq D$.

An Abox represents assertional knowledge and is a set of assertions of the form $i : C$ or $(i; j) : R$ where $C$ is a concept description, $R$ is a role description and $i; j$ are individuals.

A concept assertion $i : C$ is satisfied w.r.t. a TBox $\mathcal{T}$ if for all models $\mathfrak{J}$ of $\mathcal{T}$ it holds that $i^{\mathfrak{J}} \in C^{\mathfrak{J}}$. A role assertion $(i; j) : R$ is satisfied w.r.t. a TBox $\mathcal{T}$ if $(i^{\mathfrak{J}},j^{\mathfrak{J}}) \in R^{\mathfrak{J}}$ for all models $\mathfrak{J}$ of $\mathcal{T}$. [Mel12].

An example of ABox is:

$$\begin{gathered} \mathit{body1}: \mathrm{PersonBody} \\ \mathit{face1}: \mathrm{PersonFace} \\ \mathit{name1}: \mathrm{PersonName} \\ (\mathit{body1}, \mathit{face1}): \mathrm{isAdjacent} \end{gathered}$$

which states that $\mathit{body1}$, $\mathit{face1}$ are instances of the concepts $\mathrm{PersonBody}$ and $\mathrm{PersonFace}$ and have the relation $\mathrm{isAdjacent}$.

In this project work the knowledge base is formulated in TBoxes which map the intentional knowledge of relations and concepts. Aboxes contain the interpretation results of specific multimedia objects

### 2.3.2 Logic programing

DL reasoners introduce a set of inference services to interpret results by exploiting the knowledge base. In many cases what DL reasoners do is to query the knowledge base and to prove whether the stored knowledge is contradictory by means of inference algorithms [Kay11]. If the reasoners prove that the knowledge base is not contradictory, then the ABox is consistent w.r.t the TBox.

### 2.3.3 Query

A *conjunctive query* consists of head and body. The head contains *distinguished variables* and defined the format of the query result, the body is a sequence of query atoms [Esp11][Kay11]. This is an example of a query:

$$\{ (\mathrm{X}) \mid \mathrm{Sport(Y)}, \mathrm{hasPerformance(Y,X)}, \mathrm{PersonName(X)} \}$$

Where $\mathrm{X}$ is denotes the head and $\mathrm{Sport(Y)}$, $\mathrm{hasPerformance(Y,X)}$, $\mathrm{PersonName(X)}$ is the set of atomic queries, $\mathrm{X}$ is the result of the query.

Given the following ABox.

$$\mathit{performance1}: \mathrm{Performance}$$
$$\mathit{sport1}: \mathrm{Sport}$$
$$\mathit{name1}: \mathrm{PersonName}$$
$$(\mathit{name1, performance1}): \mathrm{hasPerformance}$$

If we compute the previous query the result has the following form:

$$\mathrm{X?\ name1}$$
$$\{ \mathrm{Sport}(\mathit{sport1}), \mathrm{hasPerformance}(\mathit{name1, performance1}), \mathrm{PersonName}(\mathit{name1}) \}$$

## 2.4 Interpretation Process

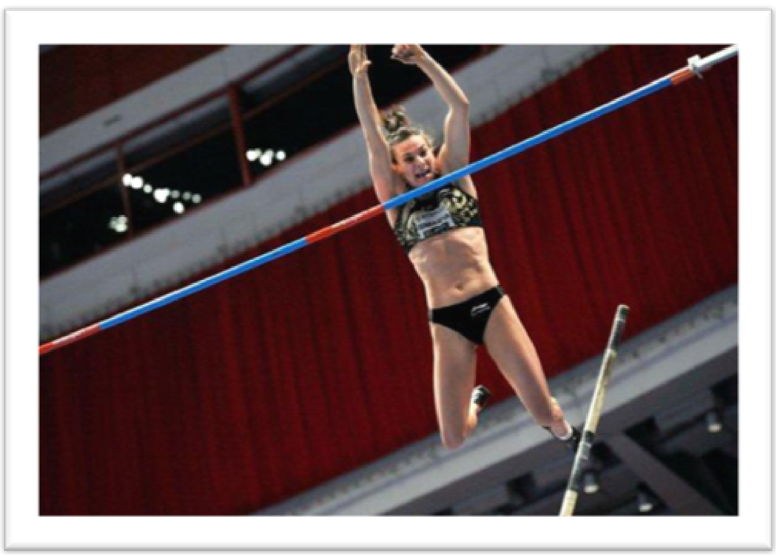
So far we have discussed that the background knowledge is compose of TBoxes. It is still needed to obtain ABoxes from the multimedia documents. Multimedia interpretation is a process based on reasoning about analysis of ABoxes with respect to background knowledge [Kaya, 60].

It can be said that one ABox represents an interpretation of a multimedia document or part of it and contains assertions about surface-level objects. An ABox is a possible content description of the image, such interpretation serves for boolean queries.

A multimedia object contains information of one section of the multimedia document. In figure we can see an image embedded, afterwards this object is interpreted.

23 FEB 2012 GENERAL NEWS 24 FEBRUARY 2012 – STOCKHOLM, SWEDEN

## ISINBAYEVA CLEARS 5.01M WORLD RECORD IN STOCKHOLM



Stockholm, SwedenMost of the nearly 10,000 spectators gathered in the Ericsson Globe Arena in Stockholm on Thursday (23) to experience the 2012 edition of "XL-Galan" chose to stay on when the meet was all but over at 10 p.m. with just one athlete in one event remaining. The minority that did leave will certainly live to regret that decision.

Figure 2. Yelena Isinvayeva clears 5.01m world record. Adapted from [INT]

After the result analysis of the image displayed in figure 2, we obtain the resulting following ABox Δ:

| | | |
|---|---|---|
| *body1* | : | PersonBody |
| *face1* | : | PersonFace |
| *bar1* | : | HotizontalBar |
| *pole1* | : | Pole |
| *(body1, face1)* | : | isAdjacent |
| *(body1, bar1)* | : | isAdjacent |
| *(pole1, body1)* | : | isAdjacent |

Table 2. Image Abox

Let's suppose that the multimedia document displayed in figure 2 is extracted from the BOEMIE repository and that the following TBox is part of the background knowledge:

| | | |
|---|---|---|
| Person | $\sqsubseteq$ | $\exists\leq_1$hasPart.PersonFace $\sqcap$ $\exists\leq_1$hasPart.PersonBody |
| Athlete | $\sqsubseteq$ | Person |
| PoleVaulter | $\sqsubseteq$ | Athlete |
| JumpingEvent | $\sqsubseteq$ | $\exists\leq_1$hasParticipant.Athlete |
| PoleVault | $\sqsubseteq$ | JumpingEvent $\sqcap$ $\exists\leq_1$hasPart.HorizontalBar $\sqcap$ $\exists\leq_1$hasPart.Pole $\sqcap$ $\forall$hasParticipant.PoleVaulter |

Table 3 . TBox

And the following set of DL-safe rules:

isAdjacent(Y, Z) $\leftarrow$ Person(X), hasPart(X, Y), PersonFace(Y), hasPart(X, Z), PersonBody(Z)

isAdjacent(Y, Z) $\leftarrow$ PoleVault(X), hasParticipant(X, Y), PoleVaulter(Y),

$$\text{hasPart}(X, Z), \text{Pole}(Z)$$

$$\text{isAdjacent}(Y, Z) \leftarrow \text{PoleVault}(X), \text{hasParticipant}(X, Y), \text{PoleVaulter}(Y), \text{hasPart}(X, Z), \text{HorizontalBar}(Z)$$

The symbolic content descriptions obtained from the analysis result of the image *body1*, *face1*, *bar1* and *pole1* are instances necessary for the interpretation process. The set of rules are considered whether are applicable using the forward-chaining and backward-chaining algorithms. In order to do this we perform a query expansion, for example for the first rule shown before an explanation Γ must be generated in the following way:

$$\text{isAdjacent}(\text{face1}, \text{body1}) \leftarrow \text{Person}(\text{IND1}), \text{hasPart}(\text{IND1}, \text{face1}), \text{PersonFace}(\text{face1}), \text{hasPart}(\text{IND1}, \text{body1}), \text{PersonBody}(\text{body1})$$

$\Gamma_1$ = { Person(IND1), hasPart(IND1, face1), PersonFace(face1), hasPart(IND1, body1), PersonBody(body1) }

Exaplanation $\Gamma_1$ is consistent w.r.t the TBox. From the other rules we can obtain the following explanations:

$\Gamma_2$ = { PoleVault(IND2), hasParticipant(IND2, IND1), PoleVaulter(IND1), hasPart(IND2,pole1), Pole(pole1) }

$\Gamma_3$ = { PoleVault(IND2), hasParticipant(IND2, IND1), PoleVaulter(IND1), hasPart(IND2,bar1), HorizontalBar(bar1) }

Since these three explanations are consistent the interpretation process is complete by adding the assertions resulted from to $\Gamma_1$, $\Gamma_2$, and $\Gamma_3$ the ABox Δ :

$$\Delta' = \Delta \cup \Gamma_1 \cup \Gamma_2 \cup \Gamma_3$$

IND1 and IND2 are the new generated assertions for Δ'. The background knowledge states that Person, PoleVault, PoleVaulter, hasPart, and hasParticipant are entailed by the rules.

| | | |
|---|---|---|
| *IND1* | : | Person |
| *IND1* | : | PoleVaulter |
| *(IND1, body1)* | : | hasPart |
| *(IND1, face1)* | : | hasPart |
| *IND2* | : | PoleVault |
| *(IND2, IND1)* | : | hasParticipant |
| *(IND2, bar1)* | : | hasPart |
| *(IND2, pole1)* | : | hasPart |

Table 4. Image ABox with new generated assertions.

The result yields a new Abox after the addition, where the assertions become part of the content descriptions of the ABox.:

| | | |
|---|---|---|
| *body1* | : | PersonBody |
| *face1* | : | PersonFace |
| *bar1* | : | HotizontalBar |
| *pole1* | : | Pole |
| *IND1* | : | Person |
| *IND1* | : | PoleVaulter |
| *IND2* | : | PoleVault |
| *(body1, face1)* | : | isAdjacent |
| *(body1, bar1)* | : | isAdjacent |
| *(pole1, body1)* | : | isAdjacent |

| | |
|---|---|
| *(IND1, body1)* | : hasPart |
| *(IND1, face1)* | : hasPart |
| *(IND2, bar1)* | : hasPart |
| *(IND2, pole1)* | : hasPart |
| *(IND2, IND1)* | : hasParticipant |

Table 5. Image ABox after the interpretation process.

### 2.5 ABox Difference

An ABox difference is a reasoning service for the detection of semantic differences in Aboxes. [Mel12]. This operation yields another ABox containing the concepts and relations that two ABoxes have in common. It is defined as follow:

Given two ABoxes and $A$, $A'$ with a TBox $T$ such that $(T,A) \not\models \bot$ and $(T,A') \not\models \bot$ the difference $\Delta_{A,A'}$ has the following properties:

1. there exists a (not necessarily total) mapping $\varphi$: $\mathrm{inds}(A') \mapsto \mathrm{inds}(A)$ such that $(T, \Phi(A' \cup \Delta_{A,A'})) \models A$ (w.r.t. $T$)
2. $A' \cup \Delta_{A,A'}$ is satisfiable (consistent),
3. $\Delta_{A,A'}$ is minimal (i.e., there is, w.r.t. $\subseteq$, no smaller $\Delta'_{A,A'}$ with the same properties),

where $\Phi(A)$ with $A= \{a_1, \ldots, a_n\}$ is defined as follows:

$$\Phi(A) =_{\mathrm{def}} \{\Phi(a_1), \ldots, \Phi(a_n)\},$$
$$\Phi(i : C) =_{\mathrm{def}} \varphi(i) : C,$$
$$\Phi((i, j) : R) =_{\mathrm{def}} (\varphi(i), \varphi(j)) : R.$$

Example 1. Let the two ABoxes:

$$A= \{i\text{: } (\mathrm{PersonBody} \sqcap \mathrm{PersonFace})\}$$
$$A'= \{i\text{: } (\mathrm{Person})\}$$

and the TBox :

$$T = \{\mathrm{Person} \equiv \mathrm{PersonBody} \sqcap \mathrm{PersonFace}\} .$$

the result of:

$$\Delta_{A,A'} \text{ is empty,}$$

because of the TBox $T$ establishes the equality between $\mathrm{Person}$ and $\mathrm{PersonBody} \sqcap \mathrm{PersonFace}$ .

Example 2. Let the two ABoxes:

$$A = \{i\text{: Person}\}$$

$$A' = \{j\text{: PersonBody}, k\text{: PersonFace})\}$$

and the TBox :

$$T = \{\mathrm{Person} \equiv \mathrm{PersonBody} \sqcap \mathrm{PersonFace}\} .$$

yields two results:

$$\Delta_{A,A'} = \{i\text{: Person}\}$$

$$\Delta_{A',A} = \{j\text{: PersonBody}, k\text{: PersonFace}\}$$

The first result $\Delta_{A,A'}$ is because the ABox $A'$ contains two different instances $j$ and $k$, one mapping to the content descripcion $\mathrm{PersonBody}$ and the other to $\mathrm{PersonFace}$. Therefore $\Delta_{A,A'}$ yields the content descriptions that $A$ has and $A'$ has not. The second result occurs exactly vice versa.

## 2.6 Fusion

The goal of fusion is to find concept instances within results obtained from different media objects, where such instances refer to the same individual in the real world and consequently add same_as statements between them. [Esp11]

Fusion is the next step after the interpretation process for a multimedia document. In the last interpretation example we obtained an ABox for one segment of the web page, nevertheless other segments may not yield the same symbolic content descriptions.

In order to fuse the interpretation of the multimedia document, the structural composition has to be taken into account so as their segment are redundant and complementary [Kay11]. The fusion of two ABoxes can yield one ABox containing the same assertions common in both ABoxes (figure 3).

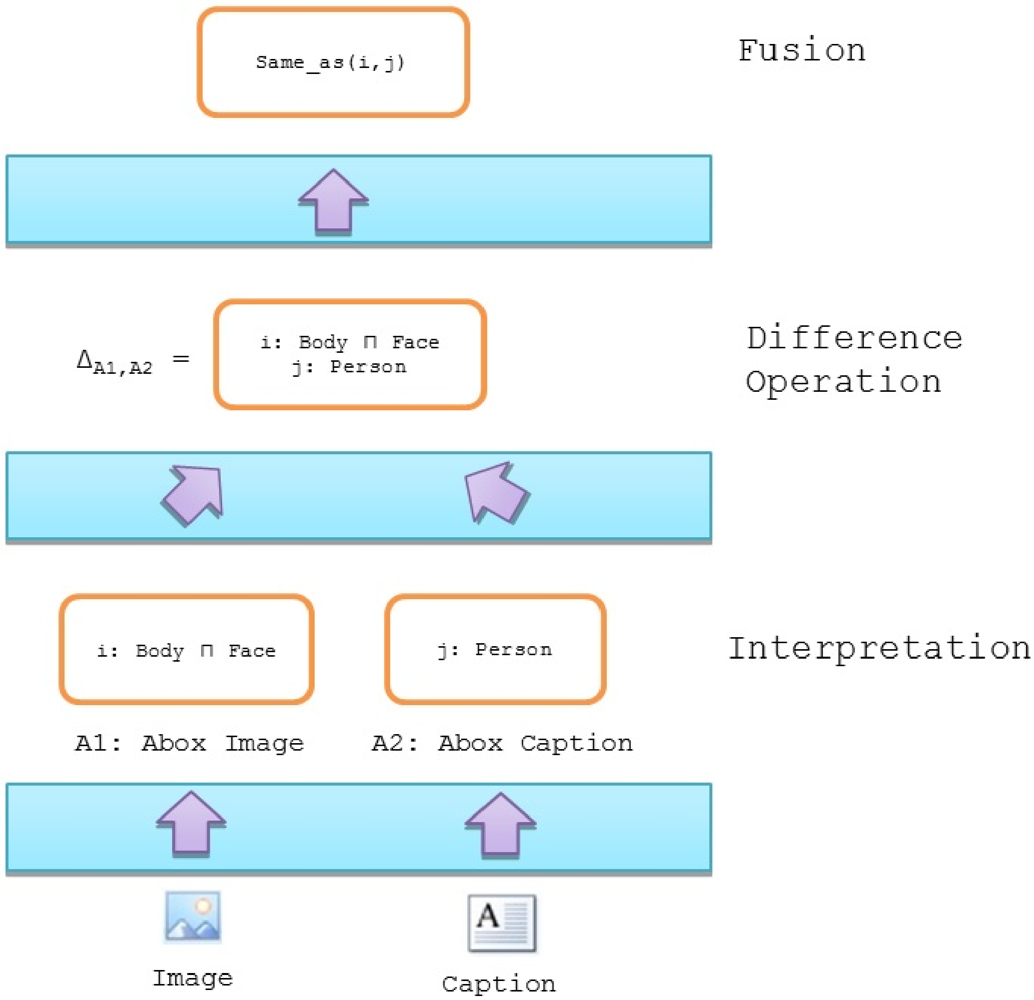



Figure 3. Fusion approach.

The fusion process consists of an algorithm in which having a set of ABoxes, their elements in common are fused in a new Abox, this is done by the *A-box difference*, the new created Abox contains derived rules stating the relation between the elements of two Aboxes.

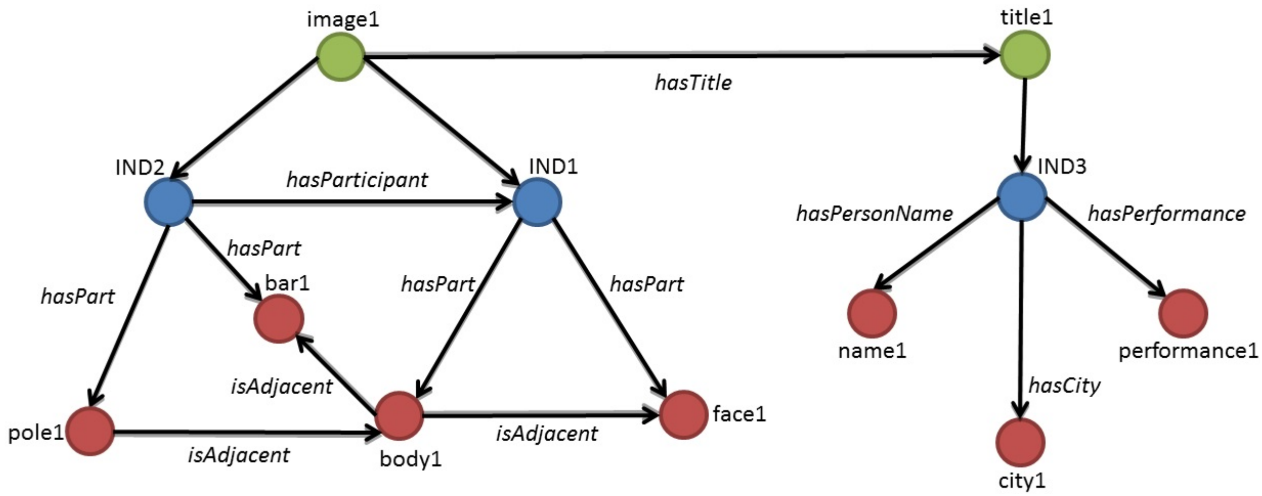



Figure 4. Interpretation of ABoxes for image and title

Figure 4 displays the interpretation of the ABoxes for image and title of figure 2. We can observe that if $hasTitle(\mathrm{image1,title1})$ then these two ABoxes can be fused if we have the following rule:

$$\mathrm{hasTitle(A, X)} \;\; \leftarrow \;\; \mathrm{Image(A), depicts(A, B), Title(X), depicts(X, Y), same\_as(Y, B)}$$

where:

$$\begin{aligned}\mathrm{hasTitle(image1, title1)} \;\; \leftarrow \;\; & \mathrm{Image(image1), depicts(image1, IND1), Title(title1),} \\ & \mathrm{depicts(title1, IND3), same\_as(IND3, IND1)}\end{aligned}$$

Which yields the new relation $\mathrm{same\_as(IND3,\ IND1)}$, therefore this is the decisive step of the fusion operation and this means that the assertions $\mathrm{IND3}$ and $\mathrm{IND1}$ yield the same symbolic content description. Now we can establish the relation displayed in Figure 5:

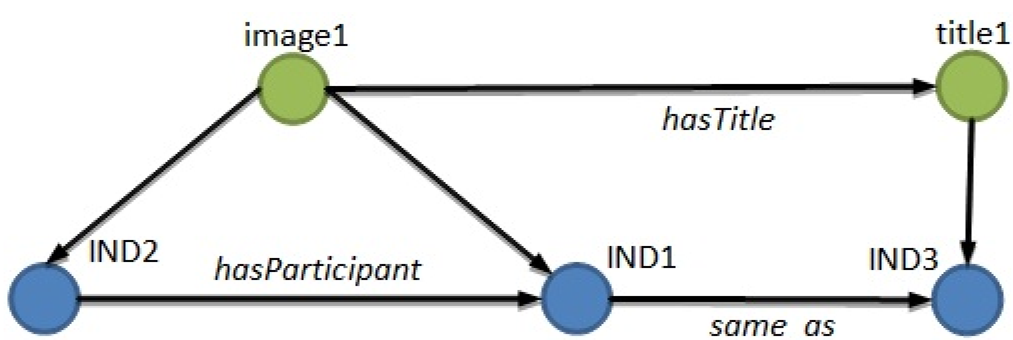


Figure 5. Illustration of fusion process.

As a result ABoxes for the image and the title result in a new Abox which is consistent w.r.t the TBox.

# 3 Software and Data

## 3.1 BOEMIE

BOEMIE stands for Bootstrapping Ontology Evolution with Multimedia Information. It is an specific background knowledge which contains multimedia content such as web pages, videos and images about athletics events news. These repository was gathered from television broadcasters and websites from sports federations and associations. The data display real events which took place mainly in european cities.

Dealing with the multimedia data contained in the documents is not the priority of this work since it is already provided in the corpus of BOEMIE. It is assumed that the semantic annotations already map the symbolic content description.

Such annotations are added to a document a posteriori and they reside separately in an environment which is linked to the document [Bos08]. This is the background knowledge present in BOEMIE which is composed by DL-based rules already designed, formalized and structured in the OWL language. Such rules can be handled by the DL RacerPro reasoner.

The *Web Ontology Language OWL* is a knowledge representation language that is widely used in the Semantic Web context. OWL has an XML-based syntax [kay11]. Basically BOEMIE can be seen using a web browser, the Racer-Pro reasoner supports this type of format.

See Fig 3 [Mel12] embedded in a web page one example of the BOEMIE content. Special information extraction (IE) tools were used to produce *low-level annotations*. Low-level annotation means that visual descriptors are extracted via annotated region, and associated with the corresponding concept [Mel12]. These annotations are added after the creation of the document and they are under the Web Ontology Language (OWL), a language used for ontology representations.

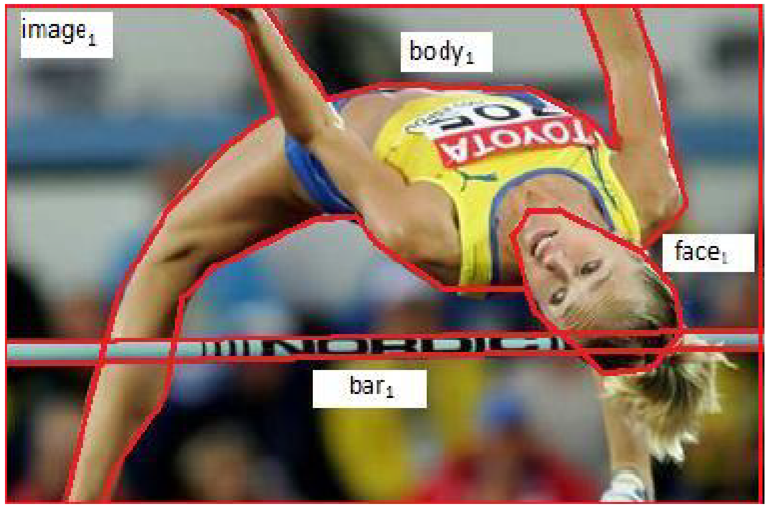



Figure 3: Image with four semantic annotations: horizontal
bar *bar1*, body *body1*, face *face1* and image *image1* [Mel12]

The DL of the BOEMIE background knowledge consist of three domain ontologies: Athletics Event Ontology, Multimedia Content Ontology and Geographics Information Ontology [Kay11]. An *ontology* determines what kind of things exist, but does not determine their specific properties and relationships [RN03]. Basically an ontology enables the concept-based modeling of a knowledge base.

The BOEMIE repository present these ontologies as merely TBoxes. We use these background knowledge to obtain new assertions in the fusion step and finally we conclude whether the generated ABoxes are accurate or not.

3.2 RacerPro

RacerPro stands for Renamed ABox and Concept Expression Reasoner Professional. This DL reasoner allows to interpret results by exploiting the knowledge base and can also be used for managing web ontologies based on OWL.

RacerPro is used to process the BOEMIE knowledge base which allows to check consistency of the TBoxes, retrieve resources, find implicit relationships, process OWL-queries and many other functionalities. When testing an ABox or a TBox, the individual object is called instance of the concept.

RacerPro is based in the XML protocol, the format of the TBoxes are given in BOEMIE in XML format and can be opened in any web browser.

In the BOEMIE scenario, a set of RacerPro instances is managed by the scenario interpretation engine. This semantic interpretation engine offers the interpretation in image, text, caption, audio and video OCR modalities [Kay11].

# 4 Experiments

## 4.1 Abox Difference

RacerPro allows to compute ABox difference using the following syntax:

```
compute-abox-difference1 A1 A2
```

The results can vary depending on the order of the ABoxes, RacerPro distinguish what ABox `A1` has that ABox `A2` does not and vice versa. So in one case one ABox might contain the other completely, partly or it might be completely different. In this experiment we have three tests.

Test 1

```
[1] ? (in-tbox TB)
[1] > TB

[2] ? (equivalent Person (and PersonBody PersonFace))
[2] > :OKAY

[3] ? (in-abox A1 TB)
[3] > A1

[4] ? (instance i (and PersonBody PersonFace))
[4] > :OKAY

[5] ? (in-abox A2 TB)
[5] > A2

[6] ? (instance j Person)
[6] > :OKAY

[7] ? (compute-abox-difference A1 A2)
[7] > (NIL)
```

TBox = {Person ≡ PersonBody ⊓ PersonFace}

A1 = {*i*: (PersonBody ⊓ PersonFace)}

A2 = {*j*: Person}

$\Delta_{A1,A2}$ = empty

This test yields empty between ABoxes A1 and A2 because their concepts map to the same content descriptions, the TBox states the equality between Person and PersonBody ⊓ PersonFace, therefore the result is empty, both ABoxes are equal.

Test 2

```
[1] ? (in-tbox TB)
[1] > TB

[2] ? (equivalent Person (and PersonBody PersonFace))
[2] > :OKAY

[3] ? (in-abox A1 TB)
[3] > A1

[4] ? (instance i Athlete)
[4] > :OKAY

[5] ? (in-abox A2 TB)
[5] > A2

[6] ? (instance j (and PersonBody PersonFace))
[6] > :OKAY

[7] ? (compute-abox-difference A1 A2)
[7] > (((instance i Athlete)))

[8] ? (compute-abox-difference A2 A1)
[8] > (((instance j Person)))
```

TBox = {Person ≡ PersonBody ⊓ PersonFace}

A1 = {*i*: Athlete}

A2 = {*j*: (PersonBody ⊓ PersonFace)}

$\Delta_{A1,A2}$ = {*i*: Athlete}

$\Delta_{A2,A1}$ = {*i*: Person}

We can see that the instruction `(compute-abox-difference A1 A2)` outputs `(instance i Athlete)` because is clear that `Athlete` is not the same as `(and PersonBody PersonFace)` and it's the only element of `A1`. Now the operation `(compute-abox-difference A2 A1)` outputs `(instance j Person)` since it is the equivalence to `(and PersonBody PersonFace)` written in the Tbox `TB`. This test shows that both ABoxes map to different content descriptions.

Test 3

```
[1] ? (in-tbox TB)
[1] > TB

[2] ? (equivalent Person (and PersonBody PersonFace))
[2] > :OKAY

[3] ? (in-abox A1 TB)
[3] > A1

[4] ? (instance i (and PersonBody PersonFace))
[4] > :OKAY

[5] ? (in-abox A2 TB)
[5] > A2

[6] ? (instance j PersonFace)
[6] > :OKAY

[7] ? (compute-abox-difference A1 A2)
[7] > (((instance i Person)))

[8] ? (compute-abox-difference A2 A1)
[8] > (NIL)
```

TBox = {Person ≡ PersonBody ⊓ PersonFace}

A1 = {*j*: (PersonBody ⊓ PersonFace)}

A2 = {*i*: PersonFace }

$\Delta_{A1,A2} = \varnothing$

$\Delta_{A2,A1} = \{i\text{: Person}\}$

In the first operation even when `A1` only has `PersonFace`, the operation retrieves an ABox containing the instances that `A1` has and `A2` has not, which is empty. `PersonFace` (`A1`) is also in `A2`. The second operation results in `Person`. Even if `PersonFace` of `A1` is contained in `A2`, they contain different instances and therefore the result states clearly that they are not equal.

4.2 Inconsistence difference operation

```
[1] ? (in-tbox tb)
[1] > tb

[2] ? (disjoint athlete not athlete)
[2] > :OKAY

[3] ? (in-abox a1)
[3] > a1

[4] ? (instance i athlete)
[4] > :OKAY

[5] ? (in-abox a2)
[5] > a2

[6] ? (instance j not athlete)
[6] > :OKAY

[7] ? (compute-abox-difference1 a1 a2)
[7] > (:ERROR "ABox a1 is incoherent.")

[8] ? (compute-abox-difference1 a2 a1)
[8] > (:ERROR "ABox a1 is incoherent.")
```

TBox = {Athlete ≠ ~Athlete}

A1 = {$i$: Athlete}

A2 = {$j$: ~Athlete}

$\Delta_{A1,A2}$ = ERROR

This test there cannot be an ABox difference operation because both ABoxes are different and they are inconsistent.

### 4.3 Queries in RacerPro

```
[1] ? (in-tbox TB)
[1] > TB

[2] ? (implies Person Body)
[2] > :OKAY

[3] ? (implies Person Face)
[3] > :OKAY

[4] ? (implies Person PersonName)
[4] > :OKAY

[5] ? (in-abox A2 TB)
[5] > A2

[6] ? (instance performance1 Performance)
[6] > :OKAY

[7] ? (instance name1 PersonName)
[7] > :OKAY

[8] ? (instance person1 Person)
[8] > :OKAY

[9] ? (related person1 performance1 hasPerformance)
[9] > :OKAY

[10] ? (related person1 name1 hasPersonName)
[10] > :OKAY

[11] ? (in-abox A1 TB)
[11] > A1

[12] ? (instance body1 Body)
```

```
[12] > :OKAY

[13] ? (instance face1 Face)
[13] > :OKAY

[14] ? (instance bar1 Bar)
[14] > :OKAY

[15] ? (instance person1 Person)
[15] > :OKAY

[16] ? (related body1 face1 isAdjacent)
[16] > :OKAY

[17] ? (related person1 bar1 isAdjacent)
[17] > :OKAY

[18] ? (racer-answer-query (?x ?y) (?x ?y isAdjacent))
[18] > (((?x person1) (?y bar1)) ((?x body1) (?y face1)))
```

TBox = {

Person ⊃ PersonBody,

Person ⊃ PersonFace,

Person ⊃ PersonName

}

A1 = {

*body1* : PersonBody,

*face1* : PersonFace,

*bar1* : Bar,

*person1*: Person,

(*body1*,*face1*): isAdjacent,

(*person1*,*bar1*): isAdjacent

}

A2 = {

*performance1*: Performance,

*name1* : PersonName,

*person1*: Person,

(*person1*,*performance1*): hasPerformace,

(*person1*,*name1*): hasPersonName,

}

Where the query defined is:

```
(racer-answer-query (?x ?y) (?x ?y isAdjacent))
```

is Adjacent(X,Y) ← ?(X), ?(Y)

We want to obtain the elements which satisfy the previous rule. RacerPro outputs:

```
(((?x person1) (?y bar1)) ((?x body1) (?y face1)))
```

RacerPro found two explanations w.r.t. the ABox `A1`.

```
?x body1 ?y face1
?x person1 ?y bar1
```

4.4 Fusion

Test 1

TBox = { HighJumper ≠ PoleVaulter }

AboxImage = {

*performance1* :Performance,

*p1* : HighJumper,

*p1* : Person,

*p4* : PoleVaulter,

*p4* : Person

}

AboxCaption = {

*city1* : City,

*p3* : PoleVaulter,

*p3* : hasPersonName “Yelena”,

*p3* : Person,

*p2* : HighJumper,

*p2* : Person

}

This ABoxes both have two instances which refer to the same symbolic content descriptions, we could say that p2 and p1 map to the same entity as well as p3 and p4. The image could display two athletes in a single shot or two images put together, however this is not relevant, what it's relevant is that the caption provides additional information about the name of the pole vaulter in the image (Yelena).

Now when computing the ABox difference we can obtain either city or performace1. But we are more interested in the information in common in these ABoxes, RacerPro computes these differences using `(compute-abox-difference1 AboxImage AboxCaption :only-difference-p nil)`, and creates one new ABox containing what is in common in these two ABoxes, the result is:

AboxFusion = {

*p1* : HighJumper,

*p4* : PoleVaulter,

*p2* : HighJumper,

*p3* : PoleVaulter,

(*p2*,*p1*): same_individual_as,

(*p3*,*p4*): same_individual_as,

(*p1*,*p2*): same_individual_as,

(*p4*,*p3*): same_individual_as

}

RacerPro states that two concepts map to the same content descriptions by attaching the relation `same-individual-as`. The result is an ABox called AboxFusion.

### 4.5 Fusion in the BOEMIE repository

BOEMIE already provides a set of ABoxes of a web page that contains text, captions and images, however these elements are vast and therefore the description logics yield elements in the web page separately independent. This means that a single text might have one or more ABoxes depending on its size and treat them as if they were completely different even when their description logics refer to the same entity. This issue depends on the low-level annotation tools used to produce the ABoxes.

There are cases where the ABoxes of a text might refer to the same content in two or more ABoxes of captions, this is where the fusion prove the mapping of equal content descriptions.

In this experiment it is used *java* programing language and an API to control a Racer client locally and run multiple ABox difference operations across the BOEMIE repository.

Test 1

The interpretation ABoxes are contained in a folder called `\BoemieRepository\Analysis\` where there exists a set of interpreted ABoxes of multimedia documents.
Inside of each interpretation folder of a multimedia document (`..\Interpretation\`) there are subfolders for image, caption and text ABox interpretations in owl file format. The fusion experiment needs two files, one ABox of an image and other ABox of a caption for example:

- `Caption-boemie_html_ncsr-skel_seg3emb_interpretation1.owl` : in beginning of the name of the file `Caption` stands for the part of the document that was interpreted, `seg3` has a number to identify this file.
- `Image-boemie_html_ncsr-skel_seg2emb_interpretation1.owl` : this is a ABox from an image and is identified by the number two.

`Fusion_Img2_Cap3.owl` : is the name of the generated fusion ABox file.

The content descriptions of the ABox of the caption are:

AboxCaption = {

| | |
|---|---|
| *seg3emb* | : Caption, |
| *IND-43567* | : Person, |
| *name_1756* | : PersonName, |
| *city_1758* | : City, |
| *performance_1757* | : Performance, |
| (*city_1758*, *“Monaco”*) | : hasCityNameValue, |
| (*IND-43567*, *name_1756*) | : hasPersonName, |
| (*performance_1757*, *"6.01"*) | : hasPerformanceValue, |

(*name_1756*, *performance_1757*) : personNameToPerformance,
(*name_1756*, *"Tim Mack"*) : hasPersonNameValue,
(*seg3emb*, *IND-43567*) : depicts,
(*seg3emb*, *name_1756*) : depictsMLC,
(*seg3emb*, *"seg3emb_interpretation1"*) : hasInterpretationABox
}

The content descriptions of the ABox of the image are:

AboxImage = {
*seg2emb* : Image,
*IND-43548* : Person,
*IND-43550* : Person,
*mlc01* : PersonBody,
*mlc03* : HorizontalBar,
*mlc03* : PersonBody,
*mlc02* : PersonFace,
*mlc04* : Pole,
*mlc04* : PersonFace,
(*mlc01*, *mlc03*) : isBelow,
(*mlc01*, *mlc02*) : isBelow,
(*mlc03*, *mlc04*) : isBehind,
(*mlc03*, *mlc04*) : isAboveRight,
(*mlc02*, *mlc04*) : isRight,
(*mlc02*, *mlc03*) : isBelow,
(*seg2emb*, *IND-43548*) : depicts,
(*seg2emb*, *IND-43550*) : depicts,
(*seg2emb*, *mlc01*) : depictsMLC,
(*seg2emb*, *mlc02*) : depictsMLC,
(*seg2emb*, *mlc03*) : depictsMLC,
(*seg2emb*, *mlc04*) : depictsMLC,
(*seg2emb*, *“..W400XH600.jpg"*) : hasURL,
(*seg2emb*, *"seg2emb_interpretation1"*) : hasInterpretationABox
}

Tim Mack clears 6.01 in Monaco (Getty Images)

Figure 4: Web-page section with image and caption.

Figure 4 is the image that corresponds to the ABoxes' interpretations. We observe that the interpretation ABox of the image is not completely accurate, the tools used in the semantic extraction process to generate the BOEMIE repository gave that output, that is the reason why the interpretation ABox states two individuals of type Person. It is not relevant in the project work to evaluate the quality of the information in the repository.

The result of the ABox difference operation is:

AboxDifference = {

(*IND-43567, IND-43548*): same_individual_as,
(*IND-43567, IND-43550*): same_individual_as,
(*IND-43550, IND-43567*): same_individual_as,
(*IND-43548, IND-43567*): same_individual_as

}

The ABox difference yields the three individuals *IND-43548, IND-43567, IND-43550* are equal. The three of them are the same person, two are from the image and one from the caption.

The result of this experiment of fusion carried out on files of the BOEMIE repository shows that the ABox difference operation works efficiently.

Test 2.

This experiment tests a set of larger data, the objective is to know the speed of RacerPro to process multiple ABox-difference operations concurrently.

For this experiment a java program uses the client for setting an interface that communicates with RacerPro and provides the path of the ABox files (owl format). RacerPro gives as output many fusion ABoxes also in owl format.

The java program takes as input the root folder where are stored all the interpretation files (`\BoemieRepository\Analysis\`) and iterates continuously over hundreds of ABoxes in order to measure the time that takes to get the fusion ABoxes.

These tests were done in a computer with the following characteristics.

| | |
|---|---|
| Processor: | Intel Core i5 3.2GHz |
| RAM: | 6 GB |
| Operative System: | Windows 7 Enterprise 64-bit |

Table 6. Characteristics of the computer.

When processing the ABox difference of two 1.5 Kbytes average files the computation time is reasonable, between 4213 and 7404 milliseconds as shown in the table 7.

| KBytes | Milliseconds |
|---|---|
| 3,16113 | 4213 |
| 3,17382813 | 7404 |
| 3,83398 | 8435 |
| 3,94531 | 7734 |
| 6,16016 | 16940 |
| 6,34375 | 4702 |
| 6,64746094 | 3891 |
| 7,27051 | 3640 |
| 7,56445 | 19605 |
| 7,71777 | 8748 |
| 8,04589844 | 8140 |
| 8,26270 | 3445 |
| 9,4140625 | 7798 |
| 9,86816 | 17058 |
| 9,91796875 | 64935 |
| 10,11621 | 41279 |
| 10,56055 | 13619 |
| 11,42969 | 114236 |
| 12,8144531 | 570820 |
| 13,61523 | 3436459 |
| 15,31250 | 453505 |

Table 7. Size of ABoxes and computation time for ABox difference operation.

We also observe that the bigger the file the longer the computation time for generating the fusion ABox. When one of the ABox files is over 13 KByte the computational time increases drastically, 3436459 milliseconds of processing (57,2 minutes) and with files over 17 KBytes the computation might take hours.

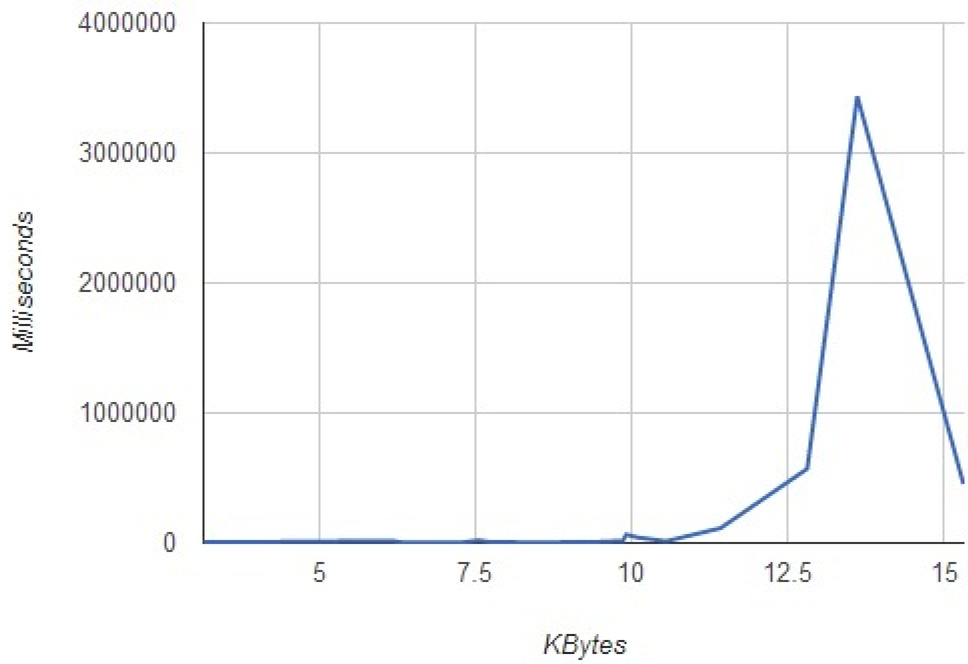



Figure 5. Graph of computation time and size of the ABox files

In the figure 5 we observe the sum of size of the two files. In a processing input of 9,9 KBytes the time is significantly small compared to an input of 12 or 13 KBytes. The tendency were the difference is less than 4 KBytes shows a huge difference of computation time. Further sizes over 17 KBytes were computed also, however the computer performance was not good.

From this we can conclude that RacerPro processes fine small files, however when working with big files it doesn't maintain a linear tendency in terms of resources needed.

4.6 Evaluation with AllegroGraph

AllegroGraph is a graphical user interface for exploring, querying, and managing meta-data as triples. AllegroGraph runs over HTTP to provide services through a web browser. With this tool we can load owl files into the AllegroGraph internal repository.

It supports SPARQL and Prolog queries, which can be saved and reused. What is useful in AllegroGraph is the visualization of individuals because the string names in the BOEMIE repository are too large. AllegroGraph forms a triple by subject, predicate and object what is the same in description logics to two things:

1) Definition of an individual

- In Description Logics

| | |
|---|---|
| *city_1758 :* | City |

- In AllegroGraph

| subject | predicate | object |
|---|---|---|
| city_1758 | type | City |

2) Relation between two individuals

- In Description Logics

| | |
|---|---|
| (*performance_1757, "6.01"*) : | hasPerformanceValue, |

- In AllegroGraph

| subject | predicate | object |
|---|---|---|
| performance_1757 | hasPerformanceValue | "6.01" |

Basically we use AllegroGraph to evaluate the ABox fusion files generated in the last experiment. Once obtained a ABox-difference operation it is necessary to compare whether different queries retrieve the same data without including the ABox fusion file. The main objective is to get something different once that the fusion was done.

AllegroGraph can also be controlled using a java client to load, delete and create repositories, this client also allows us to perform queries over the desired knowledge base of TBoxes and ABoxes.

Test 1.

One of the main goals of this project work consists of proving that the fusion ABox resulted from the ABox-difference operation helps to increase the recall in a query.

If we perform a query in the two ABoxes from where we obtain the fusion ABox, without including the fusion ABox, the output only retrieves results of one ABox, where the result is present. However if both ABoxes are related by the fusion ABox, the result should include both ABoxes. This is basically what we pretend to prove in this test.

same_individual_as is the name of the relation which is included in the fusion ABox owl file. This relation states that two individuals of two ABoxes are equal, nevertheless the amount of relations can be more than one, for instance:

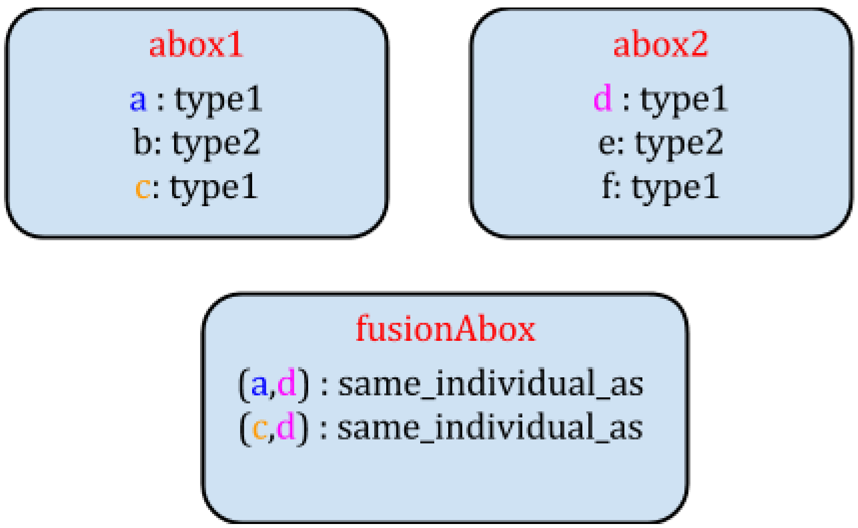



Figure 6. Fusion ABox same_individual_as relations

In figure 6 we observe that there are two same_individual_as relations, $a$, $c$ and $d$ refer to the same individual. If we perform a query to look for *type1*, we will get two documents, however $f$ is not part what we are interested for. Let's suppose that the value of *a* is "*Tim Mack*", if we look for this string in the query, the result should include also the value of $c$ and $d$.

If we look for $d$, $c$ or $a$ the result must retrieve the correct elements in both documents. Thus there are six queries that can point to two documents. Without the fusion ABox, the same queries lead to just one document. In resume three individuals increase the recall in the queries

In order to know the individuals that will retrieve more that one result in a query, we use a process that iterates over all the individuals, executes a query and counts the number of results. Such query contains *same_individual_as*. This query looks like:

```
select ?x ?y where{ { <INDIVIDUAL> owl:sameAs ?y} }
```

The test was executed using a java client that performs the following steps:
1) Connect to the repository over HTTP
2) Clean the repository
3) Load the two ABoxes
4) Count the number of individuals and store them in a list
5) Load the fusion ABox
6) Iterate over the list of individuals and perform the query
7) Sum up the number of results
8) Compare the number of results against the number of the individuals
9) The difference is the gain including the fusion ABox

From the folder:

`\BoemieRepository\Analysis\00e46af4-a6ef-431b-a34b-41e54f96fc45\_aux\interpretation`

The following files were used.

- `Caption\Caption-boemie_html_ncsr-skel_seg3emb_interpretation1.owl` : ABox ofa caption.
- `Image\Image-boemie_html_ncsr-skel_seg2emb_interpretation1.owl` : ABox if an image.
- `Fusion_Img2_Cap3.owl` : ABox-difference operation.

The result was 4 as the gain.

Test 2.

Taken the last test as initial point we iterated the same over 200 ABoxes-difference operations, giving the following graph.

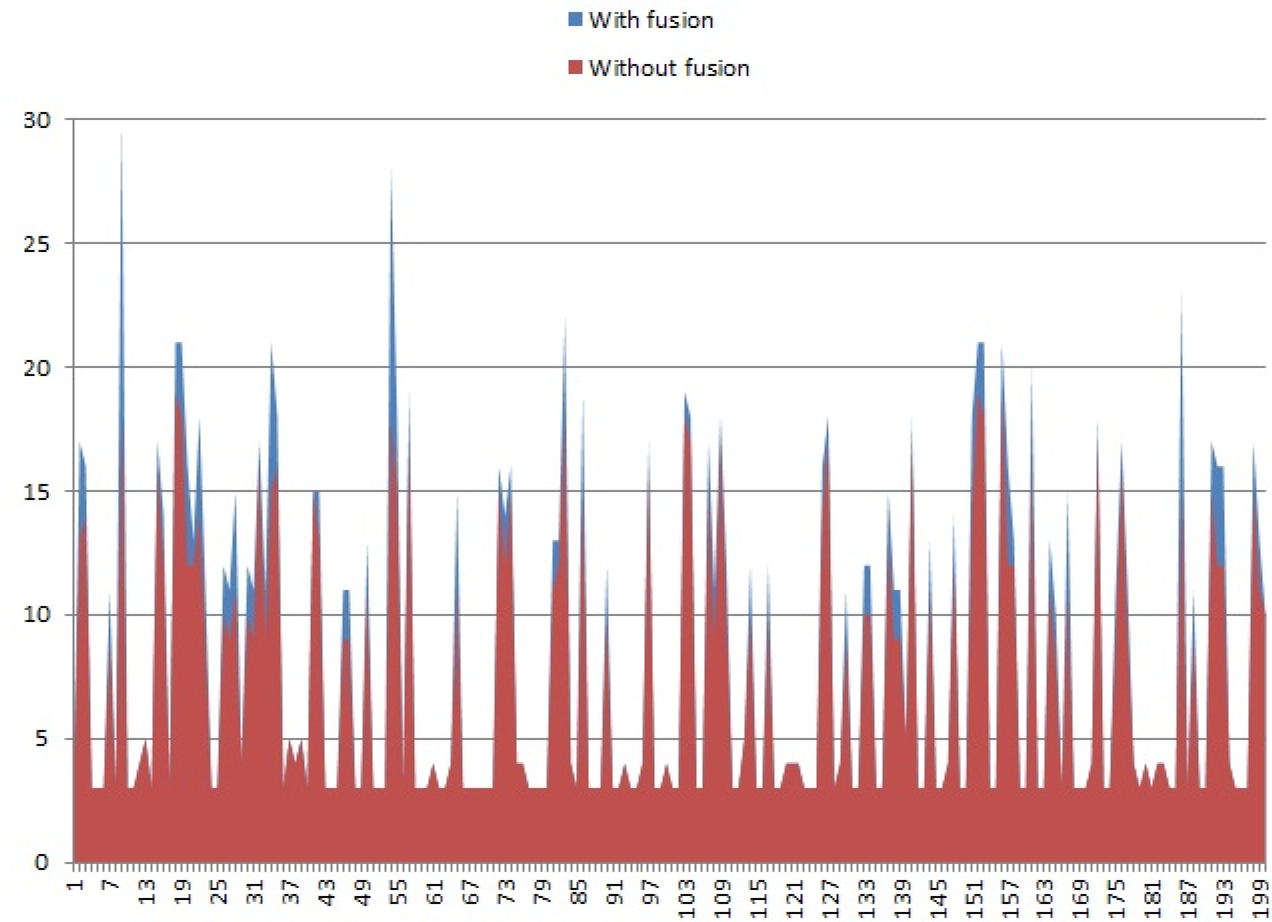


Figure 7. Graph comparison of result with fusion and without fusion

As we can see in figure 7 the blue part behind the red part represents the gain when we include the fusion ABoxes. From the 200 sets of ABoxes we obtain that 1432 individuals are retrieved without fusion and 1630 with the fusion ABox. This means 13,82 percent recall increased.

4.7 Comparison between Fusion by ABox-difference and Fusion by Inference

Atyla Kaya's procedure to obtain the fusion ABox by forward and backward chaining was implemented over the BOEMIE repository. The results of his experiment are also stored in the repository as a big ABox per document, as we already explained, the document is divided in zones, from where separated ABoxes are extracted, namely image, caption and text. Nevertheless Kaya's ABox contains all same_individual_as relations in one single ABox already interpreted.

ABox-difference operation on the other hand could not have been implemented completely over all ABoxes of a document, the reason as it was previously explained was the resources that owl files needed in relation of their size. The bigger they are, the more the computational resources they require to be processed. For this reason ABoxes of the text part were excluded as well as some ABoxes from captions and images that were too big.

The experiment to compare both techniques took into consideration the the few processed ABoxes by difference operation. This was not equal given that Kaya's result is much bigger and contains more individuals as well as its amount of same_individual_as relations. The ratio however can still be compared.

Again AllegroGraph was used here together with its java client, the process was:

1) Connect to the repository over HTTP
2) Clean the repository
3) Load the ABoxes of images and captions that were used to obtained the ABox-difference operation
4) Load the ABoxes of ABox-difference operation
5) Read the amount of individuals
6) Iterate over the list of individuals and perform the query to get the number of same_individual_as relations
7) Sum up the number of results
8) Clean the repository
9) Load the ABox from Kaya
10) Read the amount of individuals
11) Iterate over the list of individuals and perform the query to get the number of same_individual_as relations
12) Sum up the number of results

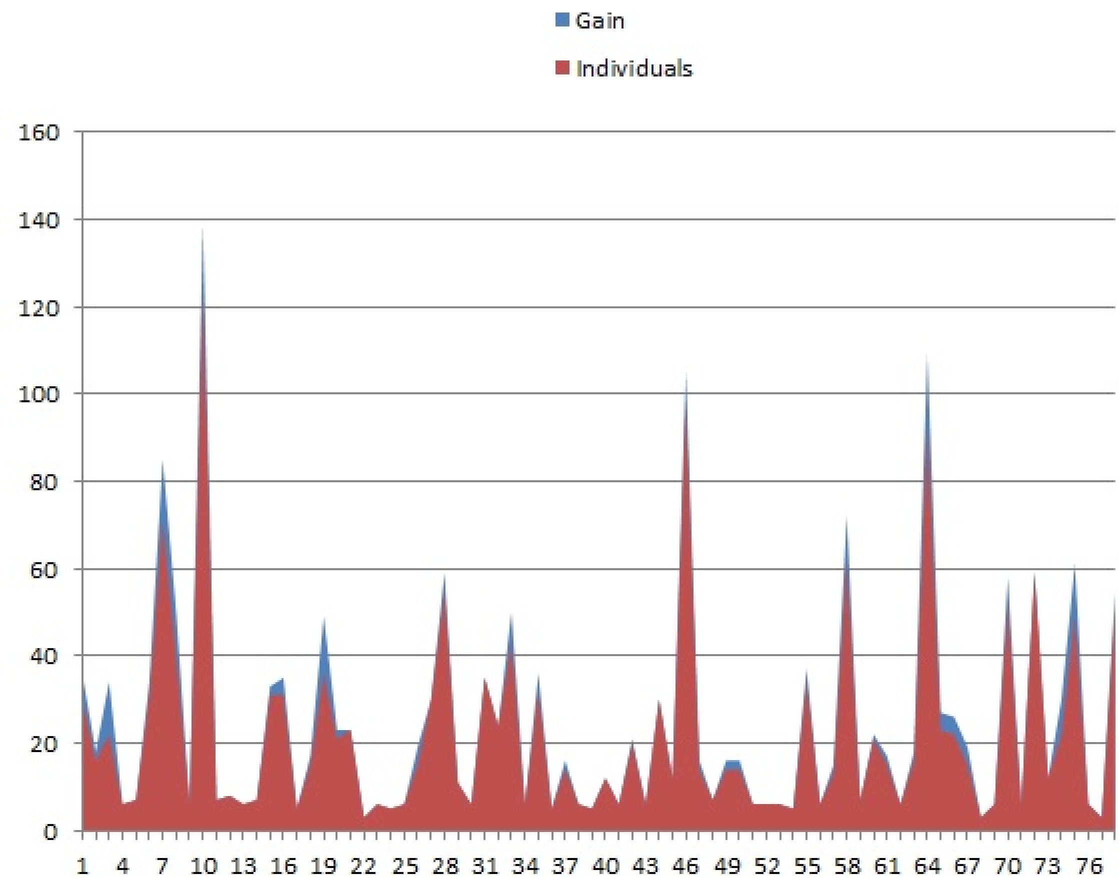


Figure 8. ABox-difference result. Individuals and gain.

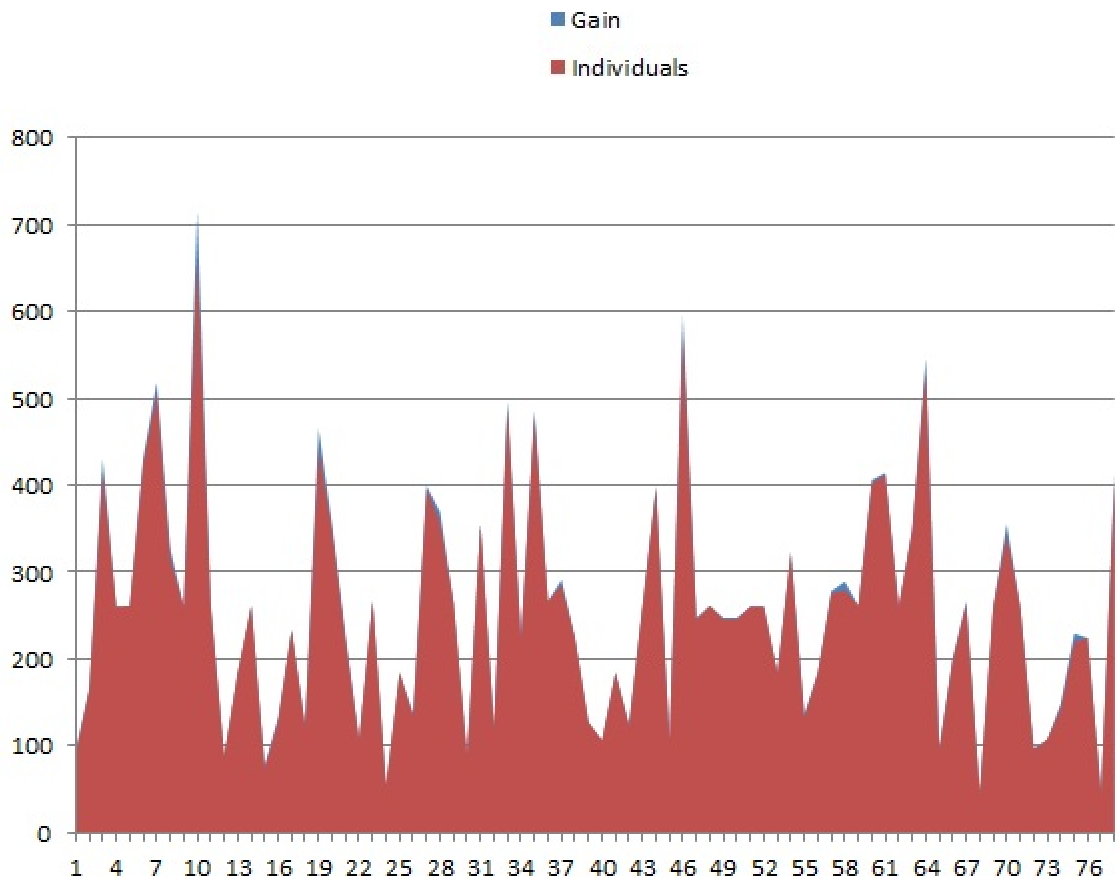



Figure 9. Inference result. Individuals and gain.

Figures 8 and 9 show the performance of both algorithms. The gain was taken as the number of additional relations achieved with same_individual_as. It is noticeable the difference of results because of the number of individuals. The measurements were nonetheless over the same documents.

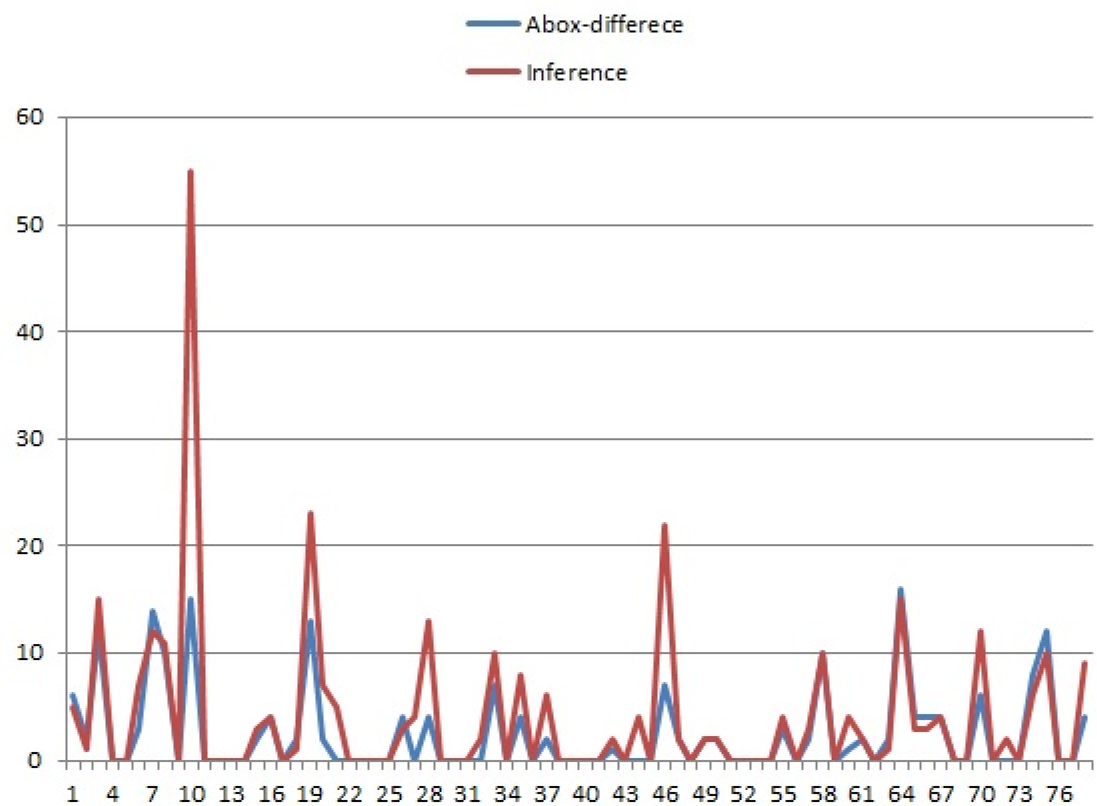


Figure 11. ABox-difference and Inference gains

Figure 11 compares the gain, inference algorithm's is bigger. But referring figure 12 we can notice the big difference between both number of individuals. The sum of the ABoxes generated by Kaya have 19 692 individuals in total, against 1 708 individuals from ABox-difference algorithm, 11,5 times more.

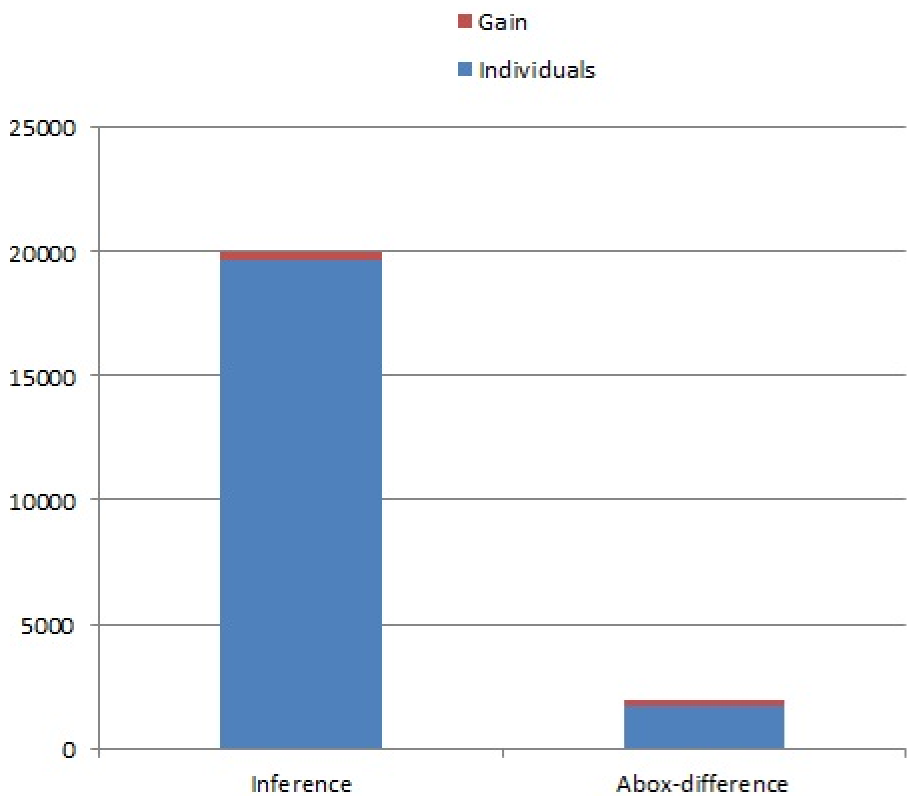



Figure 12. ABox-difference and Inference individuals and gain

Now the total gain from Kaya was 317 against 198 from ABox-difference, 1,6 more. Comparing both ratios 1:11,5 individuals and 1:1,6 gain, figure 13 shows that ABox-difference operation has superior performance. From its total individuals ABox-difference got 10,3 percent of gain, while Inference algorithm got just 1,03 percent.

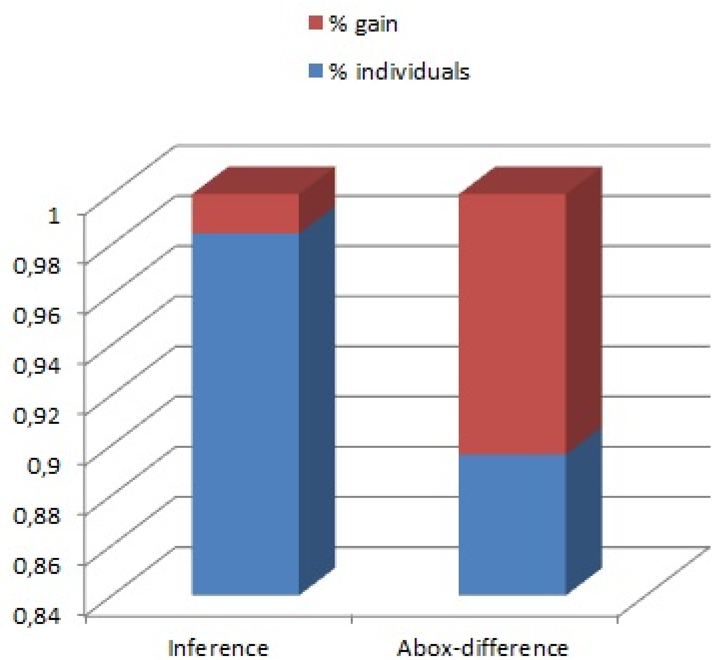



Figure 13. Ratio from Inference and ABox-difference algorithms

# Conclusions

RacerPro calculates the Abox-difference operation between two ABoxes correctly, nevertheless the performance is not optimal when their size is bigger than 13 KBytes. The computational time in a file of 17 KBytes in an average desktop computer takes more than 50 minutes and files over 19 KBytes might delay around an hour and a half. With files of more than 22 KBytes the tests could not be done because the process did not answer. For the sake of the experiments files over 10 KBytes were not considered.

Other issue observed during the ABox-difference operation experiment was that RacerPro found inconsistencies in the ABoxes and therefore it could not finish computing the task, many ABoxes in the BOEMIE repository had errors that did not allow the correct result of the ABox-difference operation, these ABoxes were not taken into consideration in the previous documentation.

The fusion algorithm using the ABox-difference operation is well structured and works correctly in terms of efficiency and operability, nevertheless the computer resources needed for it are rather high.

The completion of this set of experiments showed the detection of semantic differences in Aboxes specified in the ABox-difference properties yields correctly to a fusion approach. I proved through the experiments such evaluation by showing that the ABox-difference increases the recall. It was also observed that ABox-difference approach works better than Inference approach, both algorithms were tested over the same repository and even when the data tested for Inference was bigger, the results from Abox-difference algorithm were more satisfactory.

My personal knowledge about Description Logics, programming languages and software for Information Retrieval increased significantly as well as my experience in that field. The goals of the experiments were achieved by analyzing the data in the BOEMIE repository with the help of suites such as Protegé, AllegroGraph, RacerPro and Eclipse.

# Bibliography


[MRS08] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Matri decompositions and latent semantic indexing*, pages 403–419. Cambridge University Press, 2008.

[RN03] Stuart Russel and Peter Norvig. *Artificial Intelligence a Modern Approach*. Page 301. Prentice Hall 2003. Third edition.

[Mel12] Sylvia Melzer. *Semantic Assets: On the Relationship between Symbolic and Holistic Representations*. Phd thesis, Hamburg University of Technology (TUHH), Hamburg, Germany October 2012. Pages 39.

[Kay11] Atila Kaya. *A Logic-Based Approach to Multimedia* . Phd Thesis, Hamburg University of Technology (TUHH), Hamburg, Germany, February 2011. Page 40

[KB96] S. Koshafian and A. B. Baker. *Multimedia and Imaging Databases*. Morgan Kaufmann Publ. Inc., 1996. Pages 358

[BN03] Franz Baader and Werner Nutt. Basic description logics. In *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003. Chapter 2.

[NB03] Daniel Nardi, Ronald J. Brachman. An introduction to description logics. In Baader et at. [BN03] Chapter 1.

[Esp11] Irma Sofia Espinosa Peraldi. *Content Management and Knowledge Management: Two Faces of Ontology-Based Text Interpretation*. Phd Thesis, Hamburg University of Technology (TUHH), Hamburg, German September 2011.

[Bos08] Sebastian Bossung. *Conceptual Content Modeling - Languages, Applications and Systems.* PhD thesis, Hamburg University of Technology (TUHH), 2008.

[Rac07] Racer Systems GmbH & Co. KG. Racerpro manual 1.9, February 2007.

[INT] Modified from http://www.iaaf.org/news/news/isinbayeva-clears-501m-world-record-in-stock 20.02.2013