Identity and Reference in web-based Knowledge Representation (IR-KR)

IJCAI-09 workshop – Pasadena (CA) – 11 July 2009
`http://ir-kr.okkam.org/`

1 Introduction

The Semantic Web initiative has brought forward the idea that the web may become a space not only for publishing and interlinking documents (through HTML hyperlinks), but also for *publishing and interlinking knowledge bases (e.g. in the form of RDF graphs) in an open and fully decentralized environment*. This is how Tim Berners-Lee expressed this idea in a note from 1998:

> The Semantic Web is what we will get if we perform the same globalization process to Knowledge Representation that the Web initially did to Hypertext[1]

Even though models and languages to achieve this goal have been taken from long-standing research in AI, it is important to remark that the priorities are different. While traditionally the focus has been on theories to support sound and complete reasoning, *web-oriented KR primarily aims at dealing with issues of web-wide information interoperability and integration*. With respect to this, perhaps the most central issues is **Principle of Global Identifiers**: ”global naming leads to global network effects” (see Architecture of the World Wide Web, Volume One, 2004, at `http://www.w3.org/TR/2004/REC-webarch-20041215/`). In other words, if a resource (where a resource may range from concrete to abstract objects, from particulars to universals) is globally identified through a uniform identifier in any knowledge repository exposed on the web (e.g. in an RDF store), then any knowledge about it would be much easier to gather and integrate, distributed reasoning becomes practically possible, and knowledge-based navigation across interlinked knowledge sources can be enabled. As it happened for the web of documents, the overall value of such an open and distributed network of interlinked knowledge sources would be immensely bigger than the sum of the value of the components.

Technically, URIs (Uniform Resource Identifiers, see http://www.w3.org/Addressing/) are used to identify entities on the Semantic Web, but how to achieve shared URI understanding and reuse is object of research. This central role of identity and reference for a web-scale KR poses new challenges to traditional KR, and many researchers have suggested that the concept of URI may deeply affect the notions of language (e.g. the semantics of using the ”same” URI in different models), reference (e.g. rigid vs. non rigid designation), interpretation (e.g. the meaning of ”links” across knowlkedge bases) & reasoning (e.g. distributed reasoning across theories) in traditional logic-based KR in AI.

The goal of the workshop on Identity and Reference in web-based Knowledge Representation workshop, which in its past editions was mainly restricted to the Web and Semantic Web communities (see past editions at WWW2006[2], WWW2007[3] and ESWC2008[4]), is to open the debate on the impact and the challenges that web-oriented KR poses to some of the core concepts of traditional AI.

These working notes collect the papers which have been selected for presentation at the workshop, which was held in conjunction with IJCAI-09 at Pasadena (CA) in July 2009. The papers provide different perspectives on the issue of identity and reference, and are also an illustration of the relevance of the problem and on the diversity of views which exist on it.

We’d like to thank all the authors and the participants for their contribution to the success of the workshop.

The Organizing Committee

Paolo Bouquet, University of Trento, Italy
Marko Grobelnik, IJS, Slovenia
Harry Halpin, University of Edinburgh, UK
Frank van Harmelen, VU Amsterdam, NL
Heiko Stoermer, University of Trento, Italy
Giovanni Tummarello, DERI Galway, Ireland
Michael Witbrock, Cycorp Inc

[1] See `http://www.w3.org/DesignIssues/RDFnot.html`.

[2] `http://www.ibiblio.org/hhalpin/irw2006/`

[3] `http://okkam.dit.unitn.it/i3/`

[4] `http://www.okkam.org/IRSW2008/`

Table of content

Wojciech M. Barczynski, Falk Brauer, Alexander Löser, and Adrian Mocan
Algebraic Information Extraction of Enterprise Data Methodology and Operators … 3

David Booth
Denotation as a TwoStep Mapping in Semantic Web Architecture … 8

David Booth
The URI Lifecycle in Semantic Web Architecture … 12

Paolo Bouquet, Chiara Ghidini, and Luciano Serafini
Identity and Reference on the Global Giant Graph … 17

Francois Scharffe, Yanbin Liu, and Chunguang Zhou
RDF-AI: an Architecture for RDF Datasets Matching, Fusion and Interlink … 23

Tadej Štajner
From unstructured to linked data: entity extraction and disambiguation by collective similarity maximization … 29

Michael Witbrock, Elizabeth Coppock, and Robert Kahlert
Uniting a priori and a posteriori knowledge: A research framework … 35

Xiaoshu Wang and Arlindo L. Oliveira
Is the Web a Web of Documents or Things? … 40

Algebraic Information Extraction of Enterprise Data: Methodology and Operators

Wojciech M. Barczyński[†] **Falk Brauer**[†] **Alexander Löser**[‡] **Adrian Mocan**[†]

† SAP Research CEC Dresden
SAP AG
Dresden, Germany
{firstname.lastname}@sap.com

‡ TU Berlin
Database System and Information Management Group
Berlin, Germany
alexander.loser@tu-berlin.de

Abstract

Accurate information extraction program is a key prerequisite for the correct identification of entities on the Web, but their development is not at all a trivial task. Moreover, the maintenance, optimization, and customization of such programs require significant effort and resources. Recent trends in the Information Extraction (IE) research introduce the vision of algebraic information extraction. An important aspect of this vision is the declarative description of the extraction flows, outside of the monolithic IE program, by using a small set of generic operators. In our approach, we follow this vision and address the three main requirements that have never been addressed together in the same system before. First, we introduce a new methodology for efficient entity extraction from unstructured data, which involves the mapping of the extracted entities to already existing structured or semi-structured data. Second, we propose a set of operators addressing a comprehensive set of IE tasks, such as extracting atomic elements and aggregating them to complex real world objects, identifying relationships. Third we propose operators for leveraging global identifier providers, such as OKKAM. To verify our approach, we have implemented an information extraction system and evaluated our operators on a real example extraction flow for retrieving product information from forum pages.

1 Introduction

Today an increasing number of applications offer support for efficient, high quality aggregation and consolidation of information. These systems mainly support structured data, while numerous business domains also require analysis, aggregation and consolidation of unstructured data available in the enterprise. In particular, business domains, such as *Business Intelligence*, *Customer Relationship Management*, *Help Desk Solutions*, *Product Information Management*, are only some of the areas that require support for management and understanding of the unstructured data.

A classical example in this context is software product manager, who needs to consolidate comments, report errors and feature requests from customers of a given product (e.g. in a forum of SAP Community Network). Nowadays most of such tasks are performed mostly manually by browsing and scanning the data produced by thousands of customers each day and collected, e.g., via the company forum channels. This effort could be significantly reduced, if analysts could rely on a system able to identify and filter the data for relevant product entities, such as customers, products or error messages. The system should first recognize relevant product specific attributes, such as vendor's name, product's name, version, language, and related error codes. Next, it would aggregate recognized atomic attributes to complex product entities, e.g., to *software_product (vendor, product name, version)* and *error_message (message, error code)*. Moreover, the system should recognize the relationship between a software product and corresponding error codes in the text. Finally, it should match the recognized complex product entities, error codes and their relations against the structure and instances of the enterprise metadata systems in place.

In the past Information Retrieval, Database and Natural Language Processing communities have developed several IE systems to address the task of recognizing entities from unstructured text. For the task of identifying atomic attributes using named entity recognition (NER), tutorials like [Doan *et al.*, 2006] introduce the state-of-the-art in machine learning techniques and rule, or list based approaches and their combinations. On the other side, the database community has developed powerful systems to create semi-automatic mappings based on instances [Do and Rahm, 2007].

However, a complete solution for integrating unstructured and structured data has not been developed so far. For example, aggregation and consolidation of extracted data from unstructured text in a logical model that can be mapped to existing relational data remains an open issue. In this paper we propose a methodology to decompose the IE task into atomic operators in order to improve the reusability and to simplify the development of the tools placed on the top of the IE layer. Furthermore, the unique and persistent identification of entities of interest inside the company and even beyond its borders is still challenging. In our work, we address this issue by using the OKKAM [Bouquet *et al.*, 2008], a large scale infrastructure, which provides services to find, create and manage unique identifiers.

The structure of our paper is the following: Section 2, describes a methodology for consolidating unstructured enterprise data. In Section 3 we present a small set of generic operators allowing a developer to define complex IE. Section 4 discusses the related work and Section 5 concludes the paper.

2 Methodology for Consolidating Unstructured Enterprise Data

Before describing proposed set of operators, some basic concepts of Enterprise Resource Planning (ERP) systems are introduced. The business model of ERP systems is based on advance knowledge about relevant business objects in customers' environment. Such knowledge includes most important business objects and its attributes, but also the relations of these business objects to each other within the scope of a business process. Basically, this knowledge and its semantics are standardized and stored in a relational model. This model is often customized to the specific requirements of a customer. For integrating unstructured knowledge, system integrators also customize NER products to the specific semantics, needs of the company, and the existing ERP system. This cost intensive process includes an analysis of the specific semantics of the customer documents, an analysis of the ERP semantics, and their integration. Often such projects are very expensive and the code as well as the semantics are not reusable in other projects, because of the complex monolithic code generated in the project.

To overcome these shortcomings we focus on the aspect of lowering the effort of the extraction steps. We provide the following concepts:

1. Separate the extraction logic and flow from the extraction program code and keep it updated.
2. Use a entity identifier management systems like OKKAM to store the type system and metadata about entity with their unique identifiers. The metadata contains information needed to recognize entity independently from the format and link it to its instances.
3. Use state of the art matching and mapping algorithms to identify matches between the extracted data and existing relational ERP data, at instance level by the entity identifier management system.
4. Run complex services, such as semantic search and topical aggregation on top of the integrated unstructured data.

In contrast to a monolithic approach each step is separated from the other. Thus it allows different vendors to provide solutions for each step independently, e.g., for basic extraction, information binding and information mapping, and information search. Figure 1 shows an example for implementing these steps in a real world scenario where forum pages are analyzed for a search of comments about products. The necessary steps are:

Extracting Basic Document Features and Entities. This step extracts relevant features from the structured data and basic entities from the document corpus. Basically, each operator works on one or more annotation tuples and produces further annotation tuples. Annotation are extracted parts (or fragments of text) from a document, such as the title of a HTML page or a recognized product. They contain semantic metadata: the entity type, e.g., SAP Product; and the extracted entity itself e.g., NetWeaver 2004s. Each operator takes input as a set of annotations and returns new ones.

• **Extraction of basic document structure.** In this step 'relevant' parts of the document's structure are recognized and extracted. For example, rules can be used to extract titles, anchors, body, etc. from web documents. In the example given in Figure 1, we extract anchors from a web document using the rule based operators proposed in Section 3.1. The authors of [Reiss *et al.*, 2008; Shen *et al.*, 2007] and we have investigated that an iterative combination of simple declarative operators address most of the basic information tasks. However, our approach allows the further enrichment of this step with more complex operators if necessary.

• **Extraction and normalization of basic entities** cover the recognition of basic entities. In our example we use rules (regular expressions) for recognizing the version, a dictionary for vendors, and another one for the product names. Section 3 gives an overview of the defined operators, but this set can be extended with other state-of-the art approaches mentioned in [Doan *et al.*, 2006].

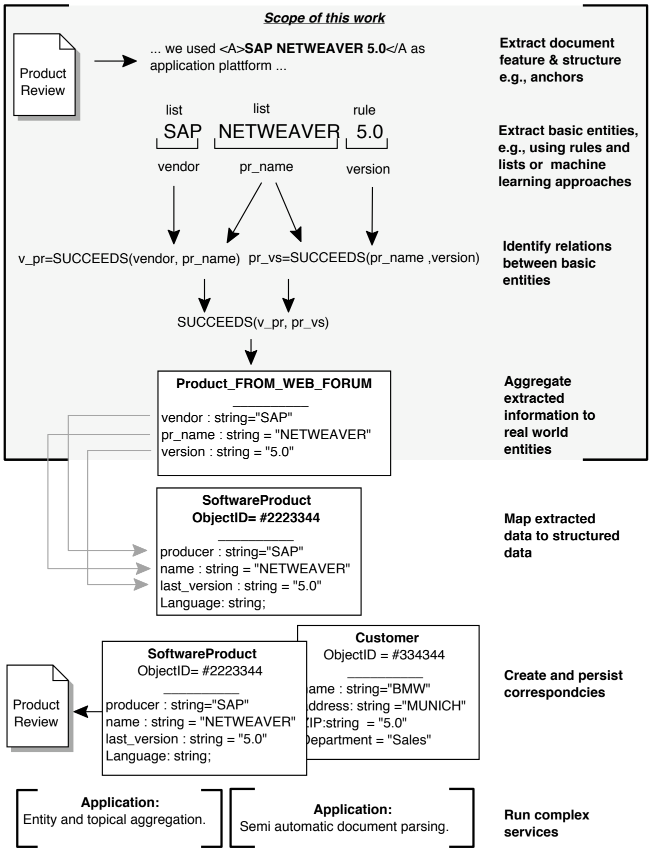

Scope of this work
Product Review
... we used <A>SAP NETWEAVER 5.0</A as application plattform ...
Extract document feature & structure e.g., anchors
list
list
rule
SAP
NETWEAVER
5.0
vendor
pr_name
version
Extract basic entities, e.g., using rules and lists or machine learning approaches
v_pr=SUCCEEDS(vendor, pr_name)
pr_vs=SUCCEEDS(pr_name ,version)
Identify relations between basic entities
SUCCEEDS(v_pr, pr_vs)
Product_FROM_WEB_FORUM
vendor : string="SAP"
pr_name : string = "NETWEAVER"
version : string = "5.0"
Aggregate extracted information to real world entities
SoftwareProduct
ObjectID= #2223344
producer : string="SAP"
name : string = "NETWEAVER"
last_version : string = "5.0"
Language: string;
Map extracted data to structured data
Customer
ObjectID = #334344
ame : string="BMW"
ddress: string ="MUNICH"
IP:string = "5.0"
epartment = "Sales"
SoftwareProduct
ObjectID= #2223344
producer : string="SAP"
name : string = "NETWEAVER"
last_version : string = "5.0"
Language: string;
Product Review
Create and persist correspondcies
Application:
Entity and topical aggregation.
Application:
Semi automatic document parsing.
Run complex services

Figure 1: Methodology for information extraction.

Bind Extracted Information to Semantic Model. Each document contains complex entities, such as products, customer signatures, address fields, and organizations, which in their turn consist of basic recognized entities. The goal of this step is twofold; first identify relationships between the basic entities within the text and bind both the basic entities and then the relationships to an existing semantic model.

• **Identify relations between basic entities.** Identifying relationships between entities from unstructured text is an open issue in information extraction (e.g. [Suchanek *et al.*, 2007] presents a solution for extracting relationships from Wikipedia).We have also defined several such operators for extracting basic relations (see Section 3.2). Figure 1 exemplifies the usage of the *SUCCEED* operator to detect the relationships between the basic entities *vendor(SAP)*, *pr_name(NETWEAVER)*, and *version(5.0)* by applying this operator three times.

• **Algebraic operations on annotation tuples.** All generic operators generate annotation tuples, which could be stored as sets in a standard SQL data base. Thus, we apply basic set operations from standard SQL, such as these operators are executed on tuples of the same annotation type and leverage the full potential of existing SQL databases.

• **Aggregate basic entities and relations to logical entities.** The overall goal of the previous information extraction steps is to generate a logical structure, where the attributes of a complex entity are bound to a logical entity structure, e.g., stored in a relational table. In Figure 1 we see a product, which is described by a vendor, a name and a version number.

Map Extracted Data to Existing Structured Data. This step maps the semantic model of the extracted data to existing ERP models, recognizes the instances in the ERP model and enriches the document with the relevant recognized ERP instances.

- **Create correspondences.** Often the extracted knowledge stored in a relational table structure does not exactly match with existing enterprise data stored in a ERP system. Several existing research prototypes address such problems, e.g., the data cleansing and de-duplication approach of [Naumann *et al.*, 2006]. Schema and instance matching has been investigated in [Do and Rahm, 2007]. This step has the role of finding the appropriate matches with the ERP business objects. As the found candidates can be either unique or ambiguous, we propose as a support for this task the usage of an company-internal OKKAM Node, which stores the most relevant data to identify an entity and allows via unique identifiers to further look up the ERP system directly for additional metadata.
- **Persist correspondences.** All the extracted logical entities, their span values, types, and mappings to business objects are linked to the document. In our example business objects corresponding to a product and to a customer were recognized in the ERP data and linked to the documents from where they have been extracted.

Running Complex Services on Top of Extracted Entities and ERP Data. Although this is an area of active research, we foresee two main scenarios for entity and topical aggregation and semi-automatic document analysis.

- **Entity and topical aggregation.** Product managers are interested in resolving complex queries over forums threads, such as "SELECT all costumers (and their commentaries), which have been recently arguing about *SAP NETWEAVER 5.0*" or "SELECT all products about data mining mentioned by customer XXX during this year". The first example is an aggregational query over a product entity, while the second example is a topical query and an aggregation over several products. Answering such queries clearly represent a typical data mining problem.
- **Semi-automatic document analysis.** So called information workers, such as specialists in a call center, are required to identify the issue raised in an incoming document (email text, attachment or scanned document) including which customer sent the received document, which products are of concern, what is the type of the document (inquiry, order, rejection etc.). Further, they needs to check whether the customer is already registered in the ERP system and what other problems the customer had before.

In consequence of using identifier provider, such as OKKAM, user or application can use entities' identifiers calling a service. Thus the service doesn't need to disambiguate entities, e.g., cope with the fact that many products can have the same product name. Moving issues of identification to separate metadata node helps to modularize client application around ERP system.

3 Algebraic Information Extraction

In the this section we introduce a set of generic, domain independent IE operators. The chosen abstraction level for this generic operators is based on our current understanding and experience with the most common information extraction tasks. Before we describe the set of operators, we briefly describe the concept of an *annotation*.

Annotations are extracted parts (fragments of text) of a document, such as the title of a HTML page or a recognized product. The extracted text is denoted as the *value* of annotation. They also contain semantic metadata: the entity type (e.g., SAP Product) and the extracted entity itself (e.g., NetWeaver 2004s with its unique identifier if it is available). An operator takes as input a set of annotations and returns new ones.

We distinguish six types of operators (Table 1):

- **Basic extraction**. Operators for document features and entities: *ErcDS*, *ErcRegEx*, *RxR*, *LC*, and *RC*.
- **Relation operators**. They identify relationships by creating complex annotations that represent complex entities. We define following relation operators: *BETWEEN* (*BET*), *SUCCEED* (*SUCC*), and *xRy*.
- **Aggregate to logical entities**. By using the *BIND* operator, we can link complex annotation (representing a complex entity) with other existing annotations. *BIND* can also create a new, logical entity, if entity type parameter is set.
- **Map to structured data operator**. After creating complex annotations, we use the *ErcRel* operator to link them to structured data for further processing. *ErcRel* creates a new annotation that points to the matched tuple in the database and links it to the input annotations.
- **Set operations** include the known set operators from SQL: *DISTINCT*, *UNION*, *JOIN*, and *GROUP BY*. These operators are executed on annotation tuples and by this leverage on the full potential of any existing SQL database.
- **Id retrieve operator**. There are situations, when entities can be disambiguated by retrieving one or more of their identifiers assigned to them by an identifier provider (for example OKKAM). Of course, this becomes a trivial task if the documents are already annotated with OKKAM identifiers that can be directly accessed. Operator *GET_UID* provides this functionality. The identifier can be used later to integrate recognized entity with ERP.

In our approach annotations are placed into a graph, which as the root has document annotation. Annotations are linked by one of four kinds relations: *LINEAGE*, *CONTAIMENT*, *IS_ATTR*, and *RELATE*, where every operator creates links between input and output annotation. The lineage information (*LINEAGE*) can be used for debugging, while *CONTAIMENT* informs that one annotation is contained by another. The containment information is derived from the span of the annotations, if we found product name in HTML title, there is a *CONTAIMENT* relation between product name annotation and HTML title annotation. Annotations are related by *IS_ATTR*, if one annotation represent complex entity C and another represent an entity E (attribute of C), C and E are linked by *IS_ATTR*. For example, product version is an attribute of a product. *RELATE* is a generic relation for dependencies between two annotations without strong semantic. For example *BETWEEN* creates an annotation, which represents the text between two annotations, and uses this kind of relation (as default) to link results with input annotations.

We would like also to introduce the concept of *context*. By *context* we denote the text boundaries for executing an operator. We allow three types of context boundaries: a given window of characters, a given window consisting of a number of terms and sentence boundaries determined by using NLP techniques.

Table 1: Algebraic operators for common IE tasks (Operator's result is bold out)

ErcDS	$(ANNOT[]{:}annots, URI{:}dataSource, eType{:}eT)$ -> $ANNOT[]$
Example:	**SAP** NetWeaver 5.0 ErcDS(ANCHOR[], VENDOR_LIST, VENDOR)
ErcRegEx	$(ANNOT[] : anns, URI : dataSource, eType{:}eT)$ -> $ANNOT[]$
Example:	SAP NetWeaver **5.0** ErcRegEx(ANCHOR[], $\backslash d\backslash .(\backslash d1,+)\backslash z$, VERSION)
RxR	$(ANNOT[] : anns, RegEx : rgx1, RegEx : rgx2, eTYPE : eT)$ -> $ANNOT[]$
Example:	<a>**SAP Netweaver 5.0**< /a> RxR(HTML_DOC[], "<a>", "< /a>", ANCHOR)
LC/RC	$(ANNOT[] : anns, eType : eT, Cntxt : c)$ -> $ANNOT[]$
Example:	SAP **Netweaver 5.0** RC(VENDOR[], PROD_NAME_CANDIDATE, 30_{chars})
BETWEEN	$(ANNOT[]{:}a_1, ANNOT[]{:}a_2, eType{:}eT, Cntxt{:}c)$ -> $ANNOT[]$
Example:	SAP **NetWeaver** 5.0 BETWEEN ($VENDOR[]$, $VERSION[]$, BE_VEN_VER, 30_{chars})
SUCCEED	$(ANNOT[] : a_1, ANNOT[] : a_2[], RegEx : rule, eType : eT, Cntxt : c)$ -> $ANNOT[]$
Example:	SAP **NetWeaver** 5.0 SUCCEED ($VENDOR[]$, $VERSION[]$, NETWEAVER_RULE, NETWEAVER,30)
BIND	$ANNOT[] : parent, (ANNOT[] : a_1 ... ANNOT[] : a_n), eType : eT$ -> $ANNOT[]$
Example:	bind vendor , product line and version extracted from anchor text BIND($ANCHOR[]$,[$PROD\_NAME[]$, $PROD\_VER[]$, $VENDOR[]$],PRODUCT)

3.1 Basic Extraction

For our basic set of operators, the extraction logic is based either on rules or dictionaries. However, the interface provided by our framework allow the plug-in of "third party" named entity recognition and named entity normalization operators. Table 1 introduces our set of basic extraction operators and gives examples on their usage. Since the semantic of most of the operators is fairly intuitive, therefore we only give a brief introduction here.

List based extraction with *ErcDS*. This operator extracts annotations using a single list of domain terms as an input. Instead of a list, our implementation does also support a column of a database as an input. In the example shown in Table 1, we use the operator to extract a new annotation of the type *VENDOR*.

Rule based extraction with *ErcRegEx*. This operator extracts annotations using a single regular expression (see Table 1).

Rule based extraction with *RxR*. This operator expects two regular expressions, which determine the left and right boundary of the text to be extracted. One common usage of this operator is the extraction of features from HTML, such as titles and anchors. The example for RxR in Table 1 shows the extraction of the highlighted text *SAP NETWEAVER 5.0* between two HTML elements *<a>* and *</a>*.

Extract left and right context with *LC* and *RC*. This operator extracts the text left or right of a given annotation. The second boundary is defined by the context parameter. The example in Table 1 shows the extraction of the text *NETWEAVER 5.0* right of annotated vendor *SAP*.

3.2 Relations Operators

After recognizing and normalizing the named entities the next important challenge is identifying possible relationships between them. The operators in this section require at least two already extracted annotations, from the same document. They use these annotations as boundaries to identify and process text between them. Similar to *LC* and *RC*, each annotator executes its operation if two boundary annotations are within a specified context window. Such operators create a $RELATE$ link between the source annotation and the newly created annotation. Table 1 and the following list give a brief introduction of these operators functionality.

Text between two annotations with *BETWEEN*. *BETWEEN* takes as an input two annotations and the text between them is used as a value for the new annotation. *BETWEEN* is normally used to list the text elements between two annotation types (Table 1).

Text between two annotations with *SUCCEED*. This is a variant of *BETWEEN* operator. It emphasis that one annotation has to be before second annotation and only then regular expression is checked. In the example shown in Table 1.

3.3 Aggregation to Logical Entities

The BIND operator is used to establish correspondences from one or more attribute to a complex real world entity. Table 1 introduces the parameters of *BIND* operation: it expects an input as a set of annotations of one or more types, in our case: *vendor*, *product line*, and *version*. These annotations will be bound to the output annotation, which denotes a logical complex entity of specified type, e.g. *product*. The complex type can represent a relational table structure. That is, the *BIND* operator creates an IS_ATTR relation between the source annotations used as input and the complex annotation created as output.

Figure 3.3 shows an example of extraction and data binding. In the first step, we extract from each HTML documents all anchors. Since we have a dictionary of vendors, we are able to identify version information using regular expressions. For this purpose we use *ErcRegEx* operator and regular expression *version_rule*. We store the version information in an annotation of type *version*. Similarly, we extract the vendor information using the *ErcDS* operator with a product vendors dictionary in annotations of type *vendor*. To obtain the text between *vendors* and *versions* we run *BETWEEN* operator on each product annotation. As left boundary type we use annotations of type *vendor* and as right boundary type we use annotations of type *version*. We limit the context window to 30 characters. Next, we create the annotation of type *product line* and finally, all three annotations (*vendor*, product line and *version*) are processed by the *BIND* operator to form a complex annotation object.

4 Related Work

There are three broad research areas that are relevant to our work: *Text analytics*, *Information extraction frameworks*, and *Declarative and algebraic information extraction*.

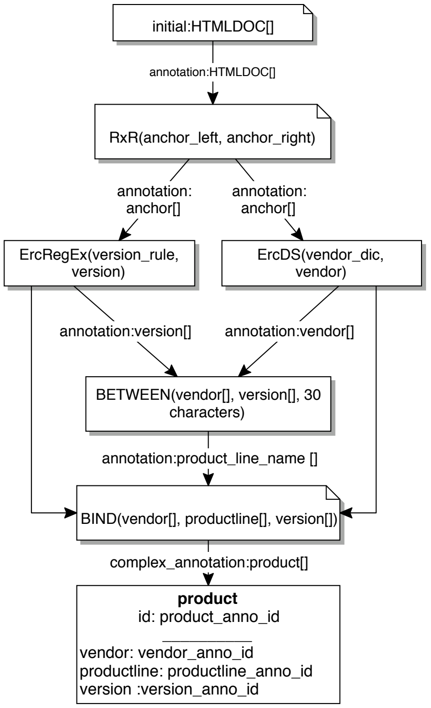

initial:HTMLDOC[]
annotation:HTMLDOC[]
RxR(anchor_left, anchor_right)
annotation: anchor[]
annotation: anchor[]
ErcRegEx(version_rule, version)
ErcDS(vendor_dic, vendor)
annotation:version[]
annotation:vendor[]
BETWEEN(vendor[], version[], 30 characters)
annotation:product_line_name []
BIND(vendor[], productline[], version[])
complex_annotation:product[]
product
id: product_anno_id
vendor: vendor_anno_id
productline: productline_anno_id
version :version_anno_id

Figure 2: Product line names from product list.

Text Analytics is a mature research area dealing with the automatic analyzes of text in order to extract structured information. Examples of common text analytic tasks include *entity identification* (e.g., identifying persons, locations, organizations, etc.), *relationships detection* (e.g., person X works in company Y) [Suchanek *et al.*, 2007] and *co-reference resolution* (identifying different variants of the same entity either in the same or different documents) [McCarthy and Lehnert, 1995]. Text analytic programs used in information extraction are called annotators and the objects extracted by them are called annotations. Traditionally, such annotations have been directly absorbed into applications. A prominent example is the AVATAR, which tackles some of these challenges [T.S.Jayram *et al.*, 2006]. In our work we focus on defining comprehensive set of operators, which could be applied across an enterprise. Furthermore we propose first operator, which leverage identifier provider, such as OKKAM, in information extraction.

Information extraction frameworks. IE developers commonly combine information extraction solutions, often as off-the-shelf IE "blackboxes", glued with additional procedural code into larger IE programs. Such programs are rather difficult to implement, understand, and debug. Recent works have presented compositional IE frameworks, such as UIMA [uim, 2009]. Extraction tasks are modeled as objects and it standardizes object APIs to enable *plug and play*. However, UIMA does not propose an algebraic approach for information extraction and therefore it is hard to optimize. By providing a set of well defined operators, we make a base for applying many optimization techniques, such as reordering.

Declarative and algebraic information extraction. One popular example of an application using this approach is the LIXTO system [Baumgartner *et al.*, 2001]. It provides to the user a graphical user interfaces to extract data from the web. The user can click on elements in a Web page and the system proposes XPath-like queries for the extraction process. It lacks integration of external data sources, like database systems and does not provide a solution for relation finding. In the area of logic programming; [Shen *et al.*, 2007] and [Chu *et al.*, 2007] describe internal representations for operators with the goal of optimization. However, they do not focus on binding the annotation objects to existing relational data.

5 Conclusions

In order to implement the vision of integrating unstructured data with structured data we addressed three key requirements which have not been addressed all together in the same framework before. We propose a new methodology for an efficient information extraction from unstructured data and new techniques on how to map the extracted data to existing structured data. Based on our knowledge and experience in the area of IE, we proposed a set of seven operators addressing most common information extraction tasks for: extracting atomic attributes, identifying relationships, and composing atomic attributes to complex real world objects.

Acknowledgments

This work is partially supported by the FP7 EU Large-scale Integrating Project OKKAM - Enabling a Web of Entities (contract no. ICT-215032). For more details, visit http://fp7.okkam.org.

References

[Baumgartner *et al.*, 2001] Robert Baumgartner, Sergio Flesca, and Georg Gottlob. Visual web information extraction with lixto. In *VLDB*, 2001.

[Bouquet *et al.*, 2008] Paolo Bouquet, Heiko Stoermer, Claudia Niederee, and Antonio Ma na. Entity name system: The backbone of an open and scalable web of data. In *ICSC '08:*, 2008.

[Chu *et al.*, 2007] Eric Chu, Akanksha Baid, Ting Chen, AnHai Doan, and Jeffrey F. Naughton. A relational approach to incrementally extracting and querying structure in unstructured data. In *VLDB*, 2007.

[Do and Rahm, 2007] Hong Hai Do and Erhard Rahm. Matching large schemas: Approaches and evaluation. *Inf. Syst.*, 32(6):857–885, 2007.

[Doan *et al.*, 2006] AnHai Doan, Raghu Ramakrishnan, and Shivakumar Vaithyanathan. Managing information extraction: state of the art and research directions. In *SIGMOD '06*, 2006.

[McCarthy and Lehnert, 1995] Joseph F. McCarthy and Wendy G. Lehnert. Using decision trees for coreference resolution. In *IJCAI*, 1995.

[Naumann *et al.*, 2006] Felix Naumann, Alexander Bilke, Jens Bleiholder, and Melanie Weis. Data fusion in three steps: Resolving schema, tuple, and value inconsistencies. *IEEE Data Eng. Bull.*, 29, 2006.

[Reiss *et al.*, 2008] Frederick Reiss, Shivakumar Vaithyanathan, Sriram Raghavan, and Rajasekar Krishnamurthyand Huaiyu Zhu. An algebraic approach to rule-based information extraction. In *ICDE*, 2008.

[Shen *et al.*, 2007] Warren Shen, AnHai Doan, Jeffrey F. Naughton, and Raghu Ramakrishnan. Declarative information extraction using datalog with embedded extraction predicates. In *VLDB*, 2007.

[Suchanek *et al.*, 2007] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *WWW '07*, 2007.

[T.S.Jayram *et al.*, 2006] T.S.Jayram, Rajasekar Krishnamurthy, Sriram Raghavan, Shivakumar Vaithyanathan, and Huaiyu Zhu. Avatar information extraction system. *IEEE Data Eng. Bull.*, 2006.

[uim, 2009] Uima sdk. http://incubator.apache.org/uima, 2009.

Denotation as a Two-Step Mapping in Semantic Web Architecture

David Booth
Cleveland Clinic
david@dbooth.org

Latest version: http://dbooth.org/2009/denotation/

Views expressed herein are those of the author and do not necessarily reflect those of Cleveland Clinic.

Abstract

In RDF, URIs are used to denote resources -- things in the universe of discourse. According to RDF semantics, an *interpretation* defines the mapping from a URI to a resource. Many interpretations may be consistent with a given RDF graph, and RDF semantics does not specify how to select a suitable interpretation from among the possible candidates. In other writings the author has advocated that in semantic web architecture, such denotation should be viewed as a *two-step mapping*: from the URI to a set of core assertions specified in a *URI declaration*, and thence to the resource. The reason for this view is that it permits a consistent resource identity to be associated with a URI: the constraints expressed in the URI declaration represent a common identity for that URI. This paper shows how this view of denotation corresponds to established RDF semantics.

Key words: Semantic Web, RDF, identity, URI declaration, URI definition, denotation, RDF semantics

1 Introduction

In RDF[Klyne 2004] URI references (hereinafter called *URIs*) are used to denote *resources* -- things in the universe of discourse. In some cases, these resources are web pages -- what the Architecture of the World Wide Web[Jacobs 2004] calls *information resources* -- but in many cases they are not: they are things like people, proteins and cars. This discussion will focus on non-information resources, but the reasoning can be extended to cover information resources.

In other writings, Booth[Booth 2007][Booth 2008] has advocated the view that in RDF assertions, the use of a URI to denote a resource involves a two-step mapping: from the URI to a set of assertions, and to thence to the resource, as illustrated in Figure 1.

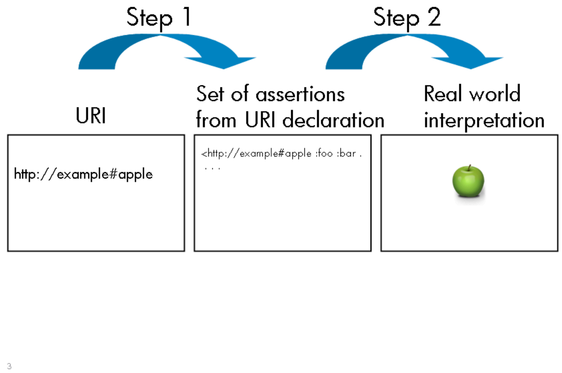

Step 1
Step 2
URI
Set of assertions from URI declaration
Real world interpretation
http://example#apple
<http://example#apple :foo :bar .
. . .
3

Figure 1: Denotation as a two-step mapping from URI to resource

This view is based on the idea that each URI is associated with a particular set of core assertions, specified in a *URI declaration*, that should be used both by statement authors writing RDF and by applications consuming and interpreting that RDF. The purpose of this view is to establish a more stable notion of resource identity by constraining the interpretations of that URI in a consistent, well-defined way.

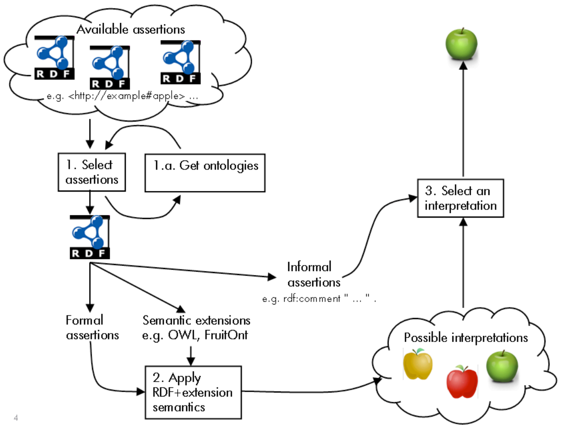
At first glance, this view of denotation as a two-step mapping may appear to deviate from established RDF semantics[Hayes 2004] (and classic logic theory). To dispel any such misunderstanding, this paper explains the correspondence between this view and RDF semantics.

1.1 RDF semantics in the context of a semantic web application

Consider a semantic web application that applies RDF semantics to draw conclusions about the resources denoted by URIs in a set of RDF assertions. In the RDF semantics, an *interpretation* specifies a mapping from URIs to a set of resources and properties. However, RDF semantics is intentionally silent about two questions that are critically important to the application:

- Given that many sets of RDF assertions may be available from many sources, how should the application decide *which* assertions to use? For example, if Abby, Bob and Carol all offer RDF documents that may be relevant to Sam's application, which ones should Sam use? Clearly this question involves complicated issues of trust, provenance and relevance, and for this reason it is typically left to human judgement.
- Once a set of assertions has been selected for a particular application, how should a suitable interpretation be selected? In other words, how should the application decide which mapping of URIs to resources should be used? The RDF semantics limits the set of possible interpretations, but typically it does not completely constrain them to a unique interpretation. For example, the RDF semantics of Sam's chosen assertions may constrain the interpretation of URI http://example#apple to denote some kind of apple, but which kind? One interpretation may map http://example/apple to a red apple, and another may map it to a green apple. Which one should Sam use?

Figure 2: RDF semantics for an application


Available assertions
RDF
RDF
RDF
e.g. <http://example#apple> ...
1. Select assertions
1.a. Get ontologies
RDF
Informal assertions
e.g. rdf:comment " ... " .
3. Select an interpretation
Formal assertions
Semantic extensions e.g. OWL, FruitOnt
2. Apply RDF+extension semantics
Possible interpretations
4

These questions correspond to steps 1 and 3 in Figure 2, which illustrates the broad process by which the application makes use of RDF assertions. In step 1, assertions are selected that are deemed relevant to the application. This is often an iterative or recursive process, as illustrated by the additional step 1.a: when an RDF document is selected for use, it may refer to ontologies that are defined in other documents, using mechanisms such as owl:imports[Dean 2004], and hence the assertions in those documents may also be merged with the set of assertions that have already been selected for use by the application.

After a set of RDF assertions has been selected, the selected assertions are often used in three ways:

- The *formal assertions* form the RDF graph whose entailments will be determined in step 2, by applying RDF semantics (and any extension semantics).
- Particular URIs -- typically namespaces -- may be recognized and trigger the inclusion of particular *semantic extensions* [Hayes 2004, section 6] in step 2. For example, the namespace URI http://www.w3.org/2002/07/owl# signals that the semantic extensions defined by OWL[Dean 2004] should be used. Although such semantic extensions are often associated with well known vocabularies such as OWL, any URI may signal the use of semantic extensions. For example, http://example#FruitOnt might signal that some special entailment rules related to fruits should be used.
- Embedded *informal assertions*, such as prose contained in rdfs:comment[Brickley 2004] statements, may be used later in step 3 to help the user select the most appropriate interpretation corresponding to a particular URI.

In step 2, entailment rules defined by the RDF semantics and any semantic extensions are applied to the formal assertions selected in step 1 to produce entailments that constrain the set of possible interpretations for the URIs in use. RDF semantics does not require entailment rules defined by semantic extensions to be used, but if the application wishes to extract the most benefit from the selected assertions, typically they will be desired. Note that according to the RDF semantics, semantic extensions must be monotonic, such that any entailments that hold without the use of the semantic extensions must also hold if the semantic extensions are used.

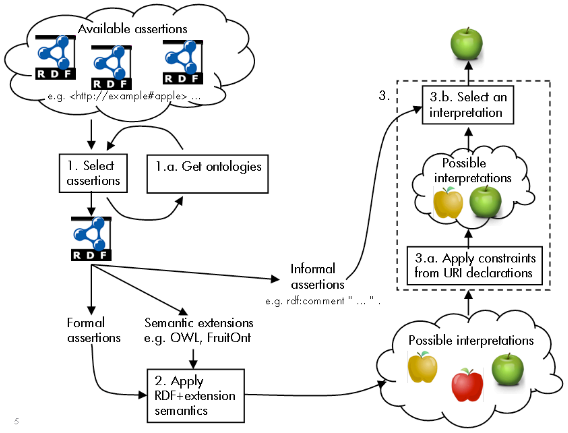
In step 3, an interpretation is selected from the set of possible interpretations, perhaps with the aid of informal assertions. The selected interpretation maps a URI used in the RDF selected in step 1, such as http://example#apple, to a resource, such as a particular green apple.

1.2 Denotation as a two-step mapping in RDF semantics

There are two ways that the two-step mapping of Figure 1 can be described in terms of RDF semantics as illustrated in Figure 2. The first is that the act of selecting an interpretation (i.e., step 3 of Figure 2) can be decomposed into two sub-steps corresponding to a two-step mapping, as shown in Figure 3:

- In step 3.a the set of possible interpretations determined by step 2 is further constrained by the core assertions from the URI declarations of the URIs used in the RDF selected in step 1, thus resulting in a (presumably) smaller set of possible interpretations.
- In step 3.b an interpretation is selected from this smaller set of possible interpetations, perhaps with the aid of informal assertions, as previously described.

Figure 3: Selecting interpretations in two steps

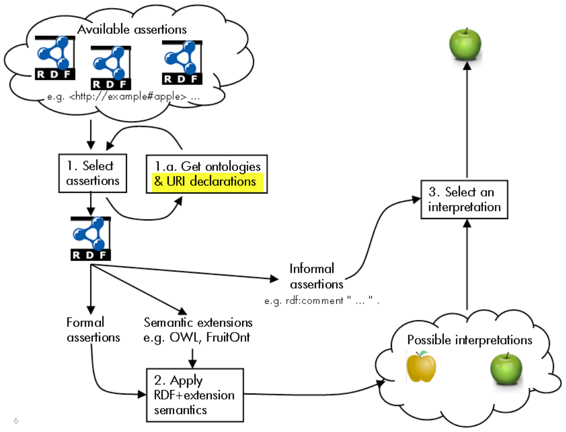

Available assertions
RDF
RDF
RDF
e.g. <http://example#apple> ...
1. Select assertions
1.a. Get ontologies
RDF
Formal assertions
Semantic extensions e.g. OWL, FruitOnt
2. Apply RDF+extension semantics
Informal assertions
e.g. rdf:comment " ... " .
Possible interpretations
3.
3.a. Apply constraints from URI declarations
Possible interpretations
3.b. Select an interpretation

Although this is the simplest way to conceptualize the correspondence, in practice the additional assertions introduced by the URI declarations are likely to be processed in a manner that is very similar to the way ontologies are processed. In Figure 4, step 1.a is expanded to perform the iterative or recursive inclusion of both ontologies and URI declarations: when an RDF document is selected for use, both ontologies and URI declarations that it uses are obtained and merged with the set of selected assertions.

The *ontological closure* is obtained if all such referenced ontologies (and URI declarations) are recursively merged. However, RDF semantics does not require an application to obtain the ontological closure: it is free to stop chasing references at any point it chooses. However, if the application does not obtain the ontological closure:

- the application may forego some entailments that it otherwise could have been obtained; and
- the application runs the risk that it may fail to detect a logical inconsistency that otherwise would have been exposed.

Figure 4: Getting URI declarations as ontologies


Available assertions
RDF
RDF
RDF
e.g. <http://example#apple> ...
1. Select assertions
1.a. Get ontologies & URI declarations
RDF
Formal assertions
Semantic extensions e.g. OWL, FruitOnt
2. Apply RDF+extension semantics
Informal assertions
e.g. rdf:comment " ... " .
Possible interpretations
3. Select an interpretation

From the perspective of semantic web architecture, this means that, although the application is free to make this choice, the *quality* of the application may suffer if it fails to obtain the ontological closure.

2 Related Work

Okkam[Bouquet 2008] is an ambitious project that seeks to establish common URI identity by providing a service for mapping from a resource description to a URI, such that multiple users who wish to refer to that resource can determine what URI to use. In principle the Okkam approach seems orthogonal (and compatible) with the two-step mapping described here, since its purpose is to map in the opposite direction.

3 Conclusions

This view of denotation as a two-step mapping from URIs to resources is entirely consistent with established RDF semantics (and classic logic theory). It merely seeks to partially specify the step of selecting a suitable interpretation for a URI -- a step that is unspecified in RDF semantics. In partially specifying this step, the range of possible interpretations for a URI is constrained by the core assertions contained in its URI declaration. This approach enables the URI to have a stable resource identity across applications: the resource identity is always constrained to a set of interpretations that is delimited by the URI declaration.

Acknowledgements

Thanks to Jonathan Rees for his description of ambiguity from classic logic theory[Rees 2009], which inspired this document.

References

[Booth 2007] David Booth. *URI Declaration in Semantic Web Architecture.* 25-Jul-2007. http://dbooth.org/2007/uri-decl/

[Booth 2008] David Booth. *Why URI Declarations? A comparison of architectural approaches.* ESWC-08 workshop on Identity and Reference on the Semantic Web 2008, 2-Jun-2008. http://dbooth.org/2008/irsw/

[Bouquet 2008] Paulo Bouquet, Heiko Stoermer and Claudia Niederee. *Entity Name System: The Backbone of an Open and Scalable Web of Data.* Proceedings of the IEEE International Conference on Semantic Computing, ICSC 2008, number CSS-ICSC 2008-4-28-25, pages 554-561, IEEE Computer Society, August 2008. http://www.okkam.org/publications/stoermer-EntityNameSystem.pdf

[Brickley 2004] Dan Brickley, R. V. Guha. *RDF Vocabulary Description Language 1.0: RDF Schema.* W3C Recommendation 10-Feb-2004. http://www.w3.org/TR/rdf-schema/

[Dean 2004] Mike Dean, Guus Schreiber, editors. *OWL Web Ontology Language Reference.* W3C Recommendation 10-Feb-2004. http://www.w3.org/TR/owl-ref/

[Hayes 2004] Patrick Hayes, editor. *RDF Semantics.* W3C Recommendation 10-Feb-2004. http://www.w3.org/TR/rdf-mt/

[Jacobs 2004] Ian Jacobs, Norman Walsh, editors. Architecture of the World Wide Web, Volume One. W3C Recommendation 15-Dec-2004. http://www.w3.org/TR/webarch/

[Klyne 2004] Graham Klyne, Jeremy J. Carroll and Brian McBride, editors. *Resource Description Framework (RDF): Concepts and Abstract Syntax.* W3C Recommendation 10-Feb-2004. http://www.w3.org/TR/rdf-concepts/

[Rees 2009] Jonathan Rees, *Learning from other disciplines.* W3C public email archive, 26-Feb-2009. http://lists.w3.org/Archives/Public/public-awwsw/2009Feb/0027.html

---

Change log
14-May-2009: Editorial fixes and added mention of related work.
16-Mar-2009: Initial version.

The URI Lifecycle in Semantic Web Architecture

David Booth
Cleveland Clinic
david@dbooth.org

Latest version: http://dbooth.org/2009/lifecycle/

Views expressed herein are those of the author and do not necessarily reflect those of Cleveland Clinic.

Abstract. Various parties are typically involved in the creation and use of a URI, including the URI owner, an RDF statement author, and a consumer of that RDF statement. What principles should these parties follow, to ensure that a consistent resource identity is established and (to the extent possible) maintained throughout that URI's lifetime? This paper proposes a set of roles and responsibilities for establishing and determining a URI's resource identity through its lifecycle.

Key words: Semantic Web, RDF, identity, URI declaration, URI definition

1 Introduction

Semantic web applications are based both on formal logic and web architecture. The Architecture of the World Wide Web (AWWW) [Jacobs 2004] describes some of the most important architectural principles underlying web applications, but additional architectural principles are needed that have not yet been well established for *semantic* web applications. Some of these pertain to the creation of URIs and the association of a URI to a resource, i.e., the URI's resource identity. This paper proposes some architectural responsibilities pertaining to resource identity and the lifecycle of a URI. They are intended as a starting point for discussion.

The AWWW defines the notion of *information resources*, which roughly correspond to web pages. But semantic web applications routinely use URIs to denote non-information resources: things such as people, proteins and cars. This paper will focus on the lifecycle of URIs that are used to denote non-information resource.

Note that the lifecyle of a URI is independent of the lifecycle of the resource that it denotes. For example, a URI that denotes the Greek philosopher Plato may be minted long after Plato has died. Similarly, one could mint a URI to denote one's first great-great-grandson even though such a child has not been conceived yet.

Words such as "MUST", "SHOULD" and "MAY" that are written in all capitals are used in the sense of RFC 2119 [Bradner 1997].

2 Roles in the URI lifecycle

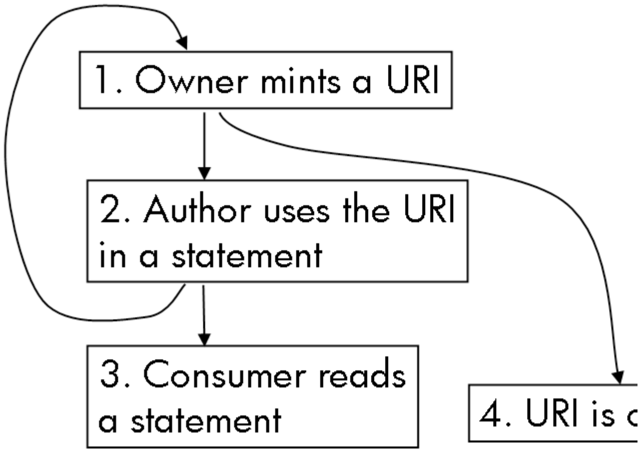
Three roles seem critically important to the URI lifecycle:

- **URI owner.** This is the person or social entity that has the authority to establish an association between a URI and a resource, as defined in AWWW. Normally it is the owner of the domain from which the URI is minted, however, the owner may delegate minting authority for all or portions of a URI space.
- **Statement author.** This is a person or agent that decides to use the URI in an RDF statement to denote a resource.
- **Consumer.** This is a person or application that reads an RDF statement and wishes to know what resource the URI was intended to denote.

3 Events in the URI lifecycle

Four common events in the URI lifecycle are illustrated in Figure 1 and described below.

Figure 1: URI Lifecycle


1. Owner mints a URI
2. Author uses the URI in a statement
3. Consumer reads a statement
4. URI is c

3.1 Event 1: Owner mints a URI

Minting a URI is the act of establishing the association between the URI and the resource it denotes. A URI MUST only be minted by the URI's owner or delegate. Minting a URI from someone else's URI space is known as *URI squatting*.[Swick 2006]

> ***URI owner responsibility 1:*** *When minting a URI, the URI owner (or delegate) SHOULD publish a URI declaration [Booth2007] at the follow-your-nose (f-y-n) location, containing core assertions whose purpose is to constrain the set of permissible interpretations [Hayes 2004] for this URI. These core assertions SHOULD NOT be changed after their publication.*

Note that a single document can serve as a URI declaration for many URIs: the correspondence between URIs and URI declarations is many-to-one.

In essence, publication of a URI's declaration creates a social expectation that the URI will be used in a way that is consistent with its declaration. This is analogous to the social expectation created when a standards organization publishes a definition for a term such as "Foo Compliant". If a party later claims that their widget is "Foo Compliant", yet that widget is not actually consistent with the "Foo Compliant" definition, that party will be seen as violating this social expectation.

Ideally, a URI declaration should also include other information (either directly or by reference) that will help statement authors and consumers make use of this URI, such as:

- Date written, author, copyright, revision history and other metadata.
- The relationship between this URI declaration and other URI declarations. For example, this URI declaration may be *broader* or *narrower* than another URI declaration: permitting a URI's set of interpretations that is a superset or subset of the other URI's set of possible interpretations, as described in Splitting Identities in Semantic Web Architecture [Booth 2009].
- Change policy for the core assertions. Some ontologies, such as SKOS [Miles 2009], have intentionally chosen to permit the definitions of their terms to be changed without minting new URIs for them. Although such a policy could be disastrous for some applications, for others it may be the most cost effective. Although changing the core assertions may change the set of permissible interpretations for a URI -- thus changing the URI's resource identity -- such changes are okay if the change policy has set expectations appropriately.
- Pointers to ancillary assertions that are believed to be compatible with this URI declaration.
- Pointers to related ontologies or data.

Although this additional information may be included directly in a URI declaration, information that is likely to need updating independent of the core assertions would be better to include by reference, so that updating this additional information will not cause consumers to think that the core assertions had changed when they did not.

Cool URIs for the Semantic Web [Sauermann 2009] describes best practices for minting URIs and hosting associated URI declarations (though it does not use the term "URI declaration").

Avoiding URI proliferation and near aliases

> ***URI owner responsibility 2****: A URI owner SHOULD NOT mint a new URI if a suitable alternate URI already exists.*

The AWWW points out that URI aliases -- multiple URIs that denote the same resource -- impose a cost on users. However, the cost of dealing with multiple URIs that denote similar but not identical resources -- near aliases -- is even greater than the cost of direct aliases, because users are forced to understand the relationships and differences between the URI declarations. Therefore, even if a new URI is deemed necessary for administrative reasons, it would be better to write the new URI declaration in terms of an existing URI's declaration than to create a new, slightly different declaration. Properties such as owl:sameAs, owl:equivalentClass and owl:equivalentProperty [Dean 2004] may be useful in some circumstances, but because they require *use* (rather than *mention* [Anonymous 2009]) of the old URI they may not be desirable in the new URI's declaration.

We do not yet have well established conventions for indicating that one URI's declaration is equivalent to another URI's declaration, though properties such as s:isBroaderThan and s:isNarrowerThan [Booth 2009] which are designed to be asserted between URIs themselves (rather than between the resources they denote), are a step in this direction.

3.2 Event 2: Author uses the URI in a statement.

An RDF statement author has a choice about whether to use a given URI in a statement. The guiding principle is:

> ***Statement author responsibility 3:*** *Use of a URI implies agreement with the core assertions of its URI declaration.*

Hence, the statement author is responsible for ensuring that he/she does indeed agree with those assertions and must NOT use the URI if he/she does not agree. However, this is not intended to represent a legal commitment. Rather it is an *identity* commitment: it indicates that the set of interpretations for that statement is intended to be constrained by the core assertions of the URI's declaration, thus constraining the resource identity of the URI.

Transitive closure of the URI declaration
Determining the complete identity commitment would involve computing the transitive closure of the URI declaration's core assertions: for each URI used in the core assertions, obtain the core assertions of that URI's declaration, etc., recursively.

> ***Statement author responsibility 4:*** *The statement author making new assertions SHOULD compute the transitive closure of the URI declarations for all URIs used, to ensure that they are consistent with the author's new assertions.*

There is a risk if the does not: a logical contradiction may go undetected until a consumer attempts to process the statement.

Identity commitment and time
What if a URI's declaration is changed after a statement author has published a statement using that URI? Should consumers assume that the statement author agrees with the new core assertions? Clearly not, since, when the statement was written, the statement author had no way of looking into the future to know what those changes would be. Hence, a more precise way of stating the identity commitment that a statement author makes by using a URI would be something like:

> ***Statement author responsibility 3a:*** *Use of a URI in a statement implies agreement with the core assertions of the URI declaration that existed at the time the statement was written.*

For this reason, RDF documents and URI declarations should indicate the date when they were written or updated. This will allow a consumer reading an RDF document later to determine whether any associated URI declarations are obsolete, and, if so, the consumer can make an informed choice about whether to seek out the original URI declaration or try using the latest.

3.3 Event 3: Consumer reads a statement.

A consumer attempting to interpret an RDF graph wishes to know what resource each URI denotes.

> ***Consumer responsibility 5:*** *The set of possible interpretations for the graph SHOULD be constrained to those that are consistent with the merge of that graph and the transitive closure of the core assertions from all of that graph's URI declarations.*

> ***Consumer responsibility 6:*** *In selecting these URI declarations, the consumer SHOULD use the URI declaration that is believed to be current for that URI (preferably from a local cache, for efficiency).*

However, the consumer MAY select a different declaration. For example:

- If the consumer wishes to be assured of most accurately following the statement author's intent, then the consumer might select the declaration that existed at the time the statement was made.
- If the consumer believes that the current declaration has been compromised (for example, by a management or ownership change of the URI domain -- see community expropriation of a URI)

then the consumer might select an older declaration.

3.4 Event 4: URI is deprecated.

> ***Statement author responsibility 7:*** *Statement authors SHOULD NOT use a URI in new RDF statements if its URI declaration has been compromised such that use of the URI is likely to cause confusion among consumers.*

This can happen, for example, if the URI declaration has been modified in violation of its published change policy or if it becomes inaccessible. In such cases, consumers may be confused about what URI declaration (or version) they should use to interpret the URI. If this occurs, a statement author should either find a different URI to use (preferably) or, if no suitable substitute is found, mint a new URI If no other URI, a new URI should be minted and its declaration should indicate that it deprecates the old URI.

3.5 Other events in the URI lifecycle

Other, less common events in the URI lifecycle may also be of interest.

Community expropriation of a URI.

In some cases, the resource identity of a URI may become so entrenched in the community that, even when its declaration is compromised, statement authors still wish to use the URI according to its original declaration. For example, the original URI owner may have gone bankrupt, and the domain name may have been sold to an unscrupulous company that proceeds to publish a new, misleading declaration for the URI.

In such cases, the community MAY temporarily expropriate that URI by continuing to write RDF statements based on the URI's original declaration, if:

- the cost of changing to new URI would be unreasonably high;
- the original URI declaration is widely known and copies are easily located by consumers;
- sufficient community discussion has taken place to make this decision;
- the decision is widely publicized and documented;
- a new URI is minted, based on the original URI declaration, with a URI declaration that indicates that the new URI deprecates the old URI, specifies a cut-off date by which all new RDF statements SHOULD use the new URI, and provides a link to the community discussion and decision.
- Each new use of the expropriated URI in an RDF document includes an rdf:isDefinedBy statement that indicates the location of the new URI declaration. *Issue: Is this the right requirement?*

Such cases should be rare. The reason to make the expropriation temporary is to avoid the indefinite accumulation of URIs that require special processing.

4 Conclusions

In understanding resource identity -- the association of a URI to a particular resource -- it is helpful to look at the roles, events and responsibilities involved in the lifecycle of a URI. This paper proposes a set of roles and responsibilities for establishing and determining a URI's resource identity through its lifecycle.

References

[Anonymous 2009] Anonymous, *Use-mention distinction*, Wikipedia, retrieved 23-Mar-2009, http://en.wikipedia.org/wiki/Use%E2%80%93mention_distinction

[Booth 2007] David Booth. *URI Declaration in Semantic Web Architecture.* 25-Jul-2007. http://dbooth.org/2007/uri-decl/

[Booth 2009] David Booth. *Splitting Identities in Semantic Web Architecture*, 26-Feb-2009, http://dbooth.org/2007/splitting/

[Bradner 1997] S. Bradner. *RFC2119 - Key words for use in RFCs to Indicate Requirement Levels,* March 1997. http://www.faqs.org/rfcs/rfc2119.html

[Dean 2004] Mike Dean, Guus Schreiber, editors. *OWL Web Ontology Language Reference.* W3C Recommendation 10-Feb-2004. http://www.w3.org/TR/owl-ref/

[Hayes 2004] Patrick Hayes, editor. *RDF Semantics.* W3C Recommendation 10-Feb-2004. http://www.w3.org/TR/rdf-mt/

[Jacobs 2004] Ian Jacobs, Norman Walsh, editors. *Architecture of the World Wide Web, Volume One.* W3C Recommendation 15-Dec-2004. http://www.w3.org/TR/webarch/

[Miles 2009] Alistair Miles and Sean Bechhofer, editors. *SKOS Simple Knowledge Organization System Reference*, W3C Candidate Recommendation 17-Mar-2009, http://www.w3.org/TR/skos-reference

[Sauermann 2009] Leo Sauermann and Richard Cyganiak. *Cool URIs for the Semantic Web*, W3C Working Draft 21-Mar-2009, http://www.w3.org/TR/cooluris

[Swick 2006] Ralph Swick. *URI squatting; please don't.* 10-Mar-2006, public email message archived at http://lists.w3.org/Archives/Public/public-swbp-wg/2006Mar/0036.html

Change log
14-May-2009: Editorial improvements.
23-Mar-2009: Initial version

Identity and Reference on the Global Giant Graph*.

Paolo Bouquet[1] **and Chiara Ghidini**[2] **and Luciano Serafini**[2]
[1] University of Trento (Italy) – `bouquet@disi.unitn.it`
[2] Fondazione Bruno Kessler (Trento, Italy) – `serafini|ghidini@fbk.eu`

Abstract

In this paper we address the issue of how data on the Global Giant Graph (GGG) can be used to answer global queries. We start with a formal model for the GGG, and then we use it to provide a formal specification of three very general modes for answering a query on the GGG, called *bounded*, *navigational* and *direct access* mode respectively. In the final discussion, we connect our model to recent discussions on URI reference and identity in the Semantic Web community.

1 Introduction

In a note from 1998[1], Tim Berners-Lee depicted the Semantic Web as a space for enabling the globalization of knowledge representation (KR). The idea is introduced through an intriguing analogy: like the Web provided an open and decentralized space for the seamless integration of any number of local hypertexts into a global, open hypertext, in which documents can be stored, interlinked and accessed in a uniform way through their URLs; so the Semantic Web should provide an open and decentralized space for the seamless integration of local knowledge bases into a global, open knowledge base, in which resources of any type (including non informational objects, e.g. people, organizations, events) can be represented, interlinked and accessed in a uniform way through their URIs.

More recently, Tim Berners-Lee has re-stated the original intuition in the idea of the "Giant Global Graph"[2] (GGG):

> So the Net and the Web may both be shaped as something mathematicians call a Graph, but they are at different levels. The Net links computers, the Web links documents.
>
> Now, people are making another mental move. There is realization now, "It's not the documents, it is the things they are about which are important". Obvious, really.

The GGG is a possible implementation of the Semantic Web as a global space for KR which Berners-Lee envisaged in 1998, and thus a concrete instance of web-based KR.

In simple words, the GGG is the result of interlinking resources which are described in different "local" RDF graphs[3], each of which represents a collection of data (or knowledge) which a user wants to share with others. The way a graph g is linked to a graph g' is by making in g a *reference* to (the HTTP URI of) a resource r which is described in g', and use this reference as a "key" to access the information about r in g'. For example, the statement:

```
#i foaf:knows
http://bblfish.net/people/henry/card#me
```

in the RDF graph stored at `http://www.w3.org/People/Berners-Lee/card` is a link to the graph `http://bblfish.net/people/henry/card` and, more precisely, to its fragment `#me`. In this respect, HTTP URI references are the "glue" of the GGG.

This "procedural" interpretation of RDF links through HTTP URIs and their use in applications (for example, in RDF browsers, like the Tabulator[4], Disco[5] or the OpenLink RDF browser[6]) is rooted in the way HTTP works, and therefore in the Web core architecture: mean HTTP URI *refer-*

*This work is partially supported by the by the FP7 EU Large-scale Integrating Project **OKKAM – Enabling a Web of Entities** (contract no. ICT-215032). For more details, visit http://www.okkam.org.

[1] See `http://www.w3.org/DesignIssues/RDFnot.html`

[2] See `http://dig.csail.mit.edu/breadcrumbs/node/215`

[3] In what follows, we assume the definition of RDF triple and RDF graph as they are defined in the document on "*Resource Description Framework (RDF): Concepts and Abstract Syntax*", [`http://www.w3.org/TR/rdf-concepts/`]. We recall here the main intuitions. An RDF graph is a collection of RDF triples, and an RDF triple is a statement of a relationship (called "predicate" or "property") between a `subject` and an `object`. We also recall that the subject of a triple can be either a RDF URI reference or a blank node, the predicate can only be a RDF URI reference, and that the object of a triple can be either a RDF URI reference, a blank node or a literal.

[4] `http://dig.csail.mit.edu/2007/tab/`

[5] `http://www4.wiwiss.fu-berlin.de/bizer/ng4j/disco/`

[6] `http://demo.openlinksw.com/DAV/JS/rdfbrowser/index.html`

ence is always dereferenced into the *same* resource, no matter in which location of the web the reference is made. So, for example, the URI `http://www.w3.org/People/Berners-Lee/card#i` is always dereferenced into the appropriate fragment of the RDF graph stored at `http://www.w3.org/People/Berners-Lee/card.rdf`.

This way of connecting local datasets into the GGG by making a direct reference to external resources is distinctive of web-based KR, and raises two interesting issues. On the one hand, the question of how a model for the GGG should look like; indeed, a semantics for RDF (and RDFS) has been proposed in 2001 by W3C[7], but it does not address directly the interpretation of interlinked RDF datasets. On the other hand, the question of how this ecology of interlinked RDF graphs can be used to gather the relevant information for answering a given query. In this paper we address both issues. First we propose a general model for the GGG based on the framework of Distributed First Order Logic or DFOL [Ghidini and Serafini, 1998]; second, we use this model as a tool for formalizing three different ways of exploiting the GGG for answering queries. In the final discussion, we connect these issues to other relevant discussions on URI reference and identity in the Semantic Web community.

2 Formalizing the GGG as a graph space

2.1 Preliminary definitions

In the rest of the paper, we use g, possibly with an index, to denote an RDF graph. URI references contained in g are denoted with $i : x$, where i is a URI, called the *prefix*, and it is used to identify a dataset, and x is the *local reference* of $i : x$ within the dataset i.

Definition 1 (Graph space)**.** Given a set of URIs I, a *graph space* on I is a family of RDF graphs $\mathcal{G} = \{(g_i)\}_{i \in I}$.

The graph space represent a specific state of the Giant Graph, where I is the set of URIs that can be dereferenced into an RDF graph. The *signature of a graph g*, denoted by $\Sigma(g)$, is the set of URIs that occurs in the graph; the signature of a graph space $\mathcal{G}$, denoted with $\Sigma(\mathcal{G})$ is the union $\bigcup_{i \in I} \Sigma(g_i)$ of the signatures of the graphs in $\mathcal{G}$. Finally we use $B(g)$ to denote the set of blank nodes of g. Blank nodes are denoted with x, y, z possibly with indexes, and they are intended to be existentially quantified variables inside a graph. A merge of a set of RDF graphs $g_1, \dots g_n$, denoted as $\mathsf{merge}(g_1, \dots g_n)$ or, $\mathsf{merge}_{i \in \{1,\dots,n\}}(g_i)$, is defined is the the union of the set of triples contained in the graphs $g'_1, \dots, g'_n$, where each g'_i is obtained by renaming the blank nodes of g_i such that $g'_1, \dots, g'_n$ don't share any blank node[8].

Definition 2 (Interpretation of an RDF graph)**.** An interpretation of an RDF graph g is a triple $(\Delta, \mathcal{I}, \mathcal{E})$; where Δ is a non empty set, $\mathcal{I} : \Sigma(g) \to \Delta$ and $\mathcal{E} : \Delta \to 2^{\Delta \times \Delta}$.

As in standard logical formalizations, an interpretation satisfies / does not satisfy statements of a knowledge base according to a formal notion of satisfiability. We therefore introduce the definition of *satisfiability* of a statement. We also define the notion of *model* of an RDF graph g as an interpretation that *satisfies* the statements in g. In the following, we use the symbol "$\equiv$" to denote `owl:SameAs` and the notation $(i : x)^{\mathcal{I}}$ to denote the application of $\mathcal{I}$ to $(i : x)$. With an abuse of notation, we use $(i : x)^{\mathcal{E}}$ to denote $\left((i : x)^{\mathcal{I}}\right)^{\mathcal{E}}$.

Definition 3 (Assignment to blank nodes)**.** Given an interpretation m of g, an *assignment* to the blank nodes of g is a function $\mathbf{a} : B(g) \to \Delta$ such that, for each node $n \in \Sigma(g) \cup B(g)$:

$$(n)^{\mathcal{I}}_{\mathbf{a}} = \begin{cases} (n)^{\mathcal{I}} & \text{if } n \text{ is an URI} \\ \mathbf{a}(n) & \text{if } n \text{ is a blank node} \end{cases}$$

Definition 4 (Satisfiability)**.** Let m be an interpretation of a graph g, $\mathbf{a}$ an assignment to $B(g)$ and $(a.b.c)$ a triple on the signature $\Sigma(g) \cup B(g)$. m *satisfies* $(a.c.b)$ under the assignment $\mathbf{a}$, in symbols, $m \models (a.b.c)[\mathbf{a}]$ if

$$((a)^{\mathcal{I}}_{\mathbf{a}}, (c)^{\mathcal{I}}_{\mathbf{a}}) \in (b)^{\mathcal{E}}$$

Given a set of triples $\Gamma = \{\gamma_1, \dots, \gamma_n\}$, $m \models \gamma_1 \wedge \dots \wedge \gamma_n[\mathbf{a}]$, if $m \models \gamma_k[\mathbf{a}]$ for $1 \leq k \leq n$.

Definition 5 (Model of an RDF graph)**.** An interpretation $m = (\Delta, \mathcal{I}, \mathcal{E})$ of g is a *model* of g, in symbols $m \models g$, if there is an assignment $\mathbf{a}$ such that

1. for any $(a.b.c) \in g$, $m \models (a.b.c)[\mathbf{a}]$
2. $(\equiv)^{\mathcal{E}}$ is the identity relation (formally, $(\equiv)^{\mathcal{E}} = \mathsf{id}(\Delta) = \{(d, d) \mid d \in \Delta\}$).

Definition 6 (Logical consequence in an RDF graph)**.** A triple $(a.b.c)$ in $\Sigma(g)$ is a logical consequence of g, in symbols $g \models (a.b.c)$ if, for any interpretation m, $m \models g$ implies there is an assignment $\mathbf{a}$, such that $m \models (a.b.c)[\mathbf{a}]$. A graph g' is a logical consequence of a graph g, in symbols $g \models g'$ if for any interpretation m, $m \models g$ implies there is an assignment $\mathbf{a}$, such that $m \models (a.b.c)[\mathbf{a}]$ for all $(a.b.c) \in g'$.

Notice that it is possible that $g \models (a.b.c)$ for all $(a.b.c) \in g'$ but $g \not\models g'$. Indeed, $g \models g'$ is true only if all the triple of g' is satisfied by the models of g w.r.t., a unique assignment; while $g \models (a.b.c)$ and $g \models (a'.b'.c')$ can be true w.r.t. different assignments. To emphasise this fact, we use the notation $\bigwedge_{k=1}^{n}(a_k.b_k.c_k)$ to denote the RDF graph composed of the n triples $(a_1.b_1.c_1), \dots, (a_n.b_n.c_n)$.

Query languages, such as SPARQL, are used to access knowledge contained in an RDF graph. In this paper we consider the simplest RDF query language constituted by the class of *conjunctive queries*. Notationally we use $\mathbf{x}$ for an n-tuple $(x_1, \dots, x_n)$ of variables (or blank nodes). Similarly, $\mathbf{c}$ is used to denote an n-tuple $(c_1, \dots c_n)$ of URIs.

Definition 7 (Conjunctive Query)**.** A *conjunctive query*, or simply a query, $q(\mathbf{x})$ on an RDF graph g is an expression of the form

$$q(\mathbf{x}) = \{\mathbf{x} \mid \bigwedge_{i=1}^{k}(a_i.r_i.b_i)\}$$

where $\mathbf{x}$ is a subset of the blank nodes occurring in $\bigwedge_{i=1}^{k}(a_i.r_i.b_i)$, and $(a_i.r_i.b_i)$ is a triple in $\Sigma(g) \cup B(g)$.

If $\mathbf{c}$ is a set of URIs in $\Sigma(g)$, $q(\mathbf{c})$ denotes the conjunction of tuples obtained by uniformly replacing x_1 with $c_1, \dots, x_n$ with c_n, in $\bigwedge_{i=1}^{k}(a_i.r_i.b_i)$.

[7] See `http://www.w3.org/TR/rdf-mt/`

[8] For the definition of merging RDF graphs, refer to `http://www.w3.org/TR/rdf-mt/`

The result of a query $q(\mathbf{x})$ on a graph g is a table with n columns, containing the row $(c_1, \ldots, c_n)$ if the RDF graph (i.e. set of triples) obtained by replacing x_i with c_i in all $(a_i.r_i.b_i)$ with $1 \leq i \leq k$ is entailed by g. We recall the notion of entailment among RDF graphs and RDF triples, as introduced in W3C Recommendation on RDF semantics[9]

Definition 8 (Query answer)**.** The answer of a query $q(\mathbf{x})$ in an RDF graph g is defined as

$$ans(q(\mathbf{x}), g) = \{\mathbf{c} \in \Sigma(g)^n \mid g \models q(\mathbf{c})\}$$

2.2 Semantics of graph spaces

In this section we provide a formal semantics for the GGG viewed as a graph space. We observe that thinking the GGG simply as the RDF graph obtained by merging all the component graphs is not adequate to capture the decentralized nature of the GGG. This approach would not capture the notion of data source, and thus the fact that a certain statement is asserted in a data source and not in another, the fact that different data sources may disagree or be complementary on the properties of the same resource, and so on. We therefore propose a more structured semantics, in which the notion of data resource (identified by a URI) is explicitly modelled.

We see the GGG as a graph space $\mathcal{G}$ composed of a family of graphs $g_1, \ldots, g_n$, and the semantics of $\mathcal{G}$ is given in terms of suitable compositions of the semantics of the component graphs. To do this we exploit the framework of local models semantics and Distributed First Order Logic [Ghidini and Serafini, 1998].

Definition 9 (Interpretation of a graph space)**.** An interpretation M for the graph space $\mathcal{G} = \{g_i\}_{i \in I}$ is a pair $(\{m_i\}_{i \in I}, \{r_{ij}\}_{i,j \in I})$ where $m_i = (\Delta_i, \mathcal{I}_i, \mathcal{E}_i.)$ is an interpretation of g_i, and r_{ij}, is a subset of $\Delta_i \times \Delta_j$. r_{ij} is called the *domain relation from i to j*.

The interpretation of a graph space $\mathcal{G}$ associates to each component graph g_i an interpretation m_i which is defined over the entire set of URIs of $\mathcal{G}$. This is justified by the fact that potentially any URI of the GGG can be reached from any graph. This semantic is consistent with the *open world assumption* usually done in the semantic web. The domain relation represent a form of inter graph equality. Intuitively the fact that $(d, d') \in r_{ij}$ means that from the point of view of g_j d and d' represents the same real world object.

Definition 10 (Model for a graph space)**.** An interpretation M for $\mathcal{G}$ is a *model* for $\mathcal{G}$, in symbols $M \models \mathcal{G}$, if

1. $m_i \models (a.b.c)$ for all $(a.b.c) \in g_i$;
2. $(d, d'), (e, e'), (f, f') \in r_{ij}$ and $(d, e) \in (f)^{\mathcal{E}_i}$ imply $(d', e') \in (f')^{\mathcal{E}_j}$.

Condition 2 in the definition above formalizes the fact that the properties stated in one graph propagate to other graphs through the domain relation. In other words, the domain relation is used to model a weak form of inter-graph identity. It is important to observe that at this stage no general identity condition is imposed on the model for the situation in which the same URI occurs in different graphs. This case will be taken into account in Section 4, where we investigate different ways of using the GGG for building the dataset for answering a query over a graph space.

A graphical representation of condition 2 is given in Figure 1. (e, f) being in the extension of f (represented in the shaded circle) in g_i is represented by a solid line between d and e. This, together with the fact that d, e and f are mapped by the domain relation into d', e' and f', entail that the pair (d', e') is necessarily in the extension of f' in g_j.

Definition 11 (Global logical consequence)**.** Let g be a set of triples in $\Sigma(\mathcal{G})$, $\mathcal{G} \models i : g$ if for all interpretations M of $\mathcal{G}$, $M \models \mathcal{G}$ implies that there is an assignment $\mathbf{a}$ such that $m_i \models g[\mathbf{a}]$, with m_i the i-th model of M.

In a graph space queries are submitted to a specific graph g_i and then propagated through semantic links to retrieve a global answer.

Definition 12 (Global answer)**.** The *global answer*, $g\_ans_i(q(\mathbf{x}), \mathcal{G})$, of a query $q(\mathbf{x})$ submitted at i, is defined as follows:

$$g\_ans_i(q(\mathbf{x}), \mathcal{G}) = \{\mathbf{c} \in (\Sigma(\mathcal{G}))^n \mid \mathcal{G} \models i : q(\mathbf{c})\}$$

Definition 12 provides a logical definition of what the answer to a query is with respect to the most generic class of models for a graph space. We now move to the illustration and formalization of different ways of using the GGG for answering a query, each of which provides a different result set. Our objective is to characterize each modality of query answering in terms of a restricted class of models for a graph space. More precisely, for each modality X which is introduced in Section 3, we will define a restricted class of models for $\mathcal{G}$, called X-models, such that the query answer relative to X is equal to the logical consequence of $\mathcal{G}$ w.r.t., the restricted class of the X-models.

3 Three Ways of Querying the GGG

We now move to the problem of building the dataset for answering a query on the GGG. In the next three short sections, we provide an intuitive descriptions of three general modes which can be adopted, whereas in Section 4 we will provide a formalization.

3.1 The Bounded Mode

A first mode, called the *bounded mode*, is to first isolate the set of RDF graphs to which the query is addressed, and then merge them to build the dataset against which the query is finally processed[10]. So, for example, given a SPARQL query on `http://www.w3.org/People/Berners-Lee/card#i`, the answer will be computed by considering exclusively the triples which are contained in the RDF dataset specified through the `FROM` keyword.

In the bounded mode, the dataset is the subgraph of the GGG explicitly specified in the `FROM` part of the query, and nothing else.

[9] `http://www.w3.org/TR/rdf-mt/`.

[10] See definition in `http://www.w3.org/TR/rdf-sparql-query/` Here we are not concerned with the distinction between RDF graphs and named graphs, so we will not make use of this distinction.

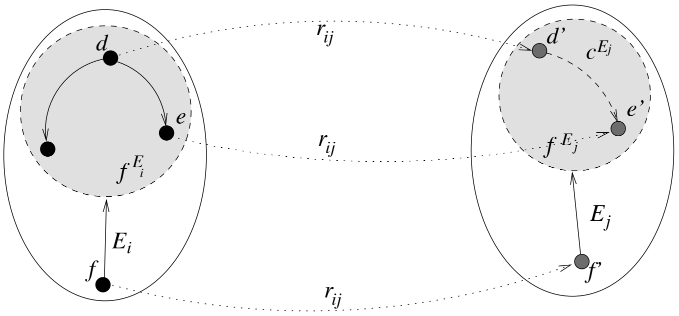


Figure 1: Example of propagation of triples acrross graphs connected by domain relation

3.2 The Navigational Mode

In the second general mode, called *navigational mode*, the query does not define the boundaries, but only the initial graph, from which all the other graphs can be reached via RDF links. This mode can be illustrated by analogy with web navigation. The idea is that, starting from a resource r in a graph g, one navigates the GGG by following the links which are found between resources, and then uses the information found in linked graphs to answer the initial query. When it reaches a fixpoint (namely, all the reachable graphs are collected), the query is evaluated against the resulting merged graph. Imagine `http://bblfish.net/people/henry/card#me` is the only external resource named in `http://www.w3.org/People/Berners-Lee/card.rdf`, and that in `http://bblfish.net/people/henry/card` there are n triples about `http://bblfish.net/people/henry/card#me` (and their object is a literal, and not another URI). Then the navigational mode would fetch and merge the triples about `http://www.w3.org/People/Berners-Lee/card#i` and about `http://bblfish.net/people/henry/card#me` from the two graphs, and then would use it as the dataset for processing the query.

The navigational mode is or attempt of modeling some of the ideas behind the Linked Data approach[11].

3.3 The Direct Access Mode

The third mode is more related to the use of search engines on the Web. In this mode, which we call the *direct access mode*, the query is processed on the graph which results from merging all the relevant graphs which can be found on the Web. Of course, there is the problem of defining what relevant means. But here we can disregard the problem, as the essence of the mode does not change. We can imagine that the relevant graphs are retrieved through a smart request to a search engine, or that we use all the RDF graphs which are available on the Web. In both cases, the dataset is not built by navigation, but by direct access.

In our example, we can imagine to collect all the RDF graphs published on the Web which contain a reference to Tim Berners-Lee (e.g. by searching them through one of the available Semantic Web search engines, like for example Sindice[12] or Swoogle[13]), collect them (notice that some of these graphs can be completely disconnected) merge them and then process the query on the resulting graph.

4 Formalization of the query modes

The formalization we provide is based on the intuition that the RDF graphs composing the GGG, are considered as entry points for the GGG. The knowledge encoded in the GGG can be retrieved by querying the GGG from one of these entry points. The different ways of building the dataset from the GGG described in the previous section represent possible ways in which a query can be evaluated and propagated across the GGG.

More in detail, the remaining of the section provides: (i) the formal definitions, in terms of query execution, of the three different procedural strategies used to answer the query in the three different modes; and (ii) a formal semantics that completely characterizes the query results in the three different modes.

4.1 The Bounded Mode

We focus here on the situation in which the answer of a query is computed on an explicitly mentioned set of datasets identified by a finite set of uris $J \subseteq I$. In this case the query is submitted to the graph g obtained by *merging* the graphs g_j with $j \in J$ and this dataset is not extended with information from graphs with indexes not in J.

Definition 13 (Bounded answer)**.** The *bounded answer* of the query $q(\mathbf{x})$ submitted at J is

$$b\_ans_J(q(\mathbf{x}), \mathcal{G}) = ans(q(\mathbf{x}), \mathsf{merge}_{j \in J}(g_j))$$

From the semantic point of view, the bounded mode, where the bounded determines the subset J of resources, can be modeled by isolating J's resources from the rest of resources (namely the $I \setminus J$-resources). This can be done by imposing that all the domain relation between resources inside J

[11] See `http://www.w3.org/DesignIssues/LinkedData.html` for the description of the approach, and `http://www4.wiwiss.fu-berlin.de/bizer/pub/LinkedDataTutorial/` for a tutorial on how to publish Linked Data on the Web).

[12] `http://sindice.com`.

[13] `http://swoogle.umbc.edu/`.

and resources outside J are empty, while domain relations between resources inside J is an isomrophism.

Definition 14 (Bounded model)**.** M is a *J-bounded model*, for any set $J \subseteq I$, if it is a model for $\mathcal{G}$ and for all $i, j \in I$

- if $i, j \in J$ then $r_{ij}((a)^{\mathcal{I}_i}) = (a)^{\mathcal{I}_j}$
- if $j \in J$ and $i \notin J$, then $r_{ij} = r_{ji} = \emptyset$

$g\_ans_J^B(q(\mathbf{x}), \mathcal{G})$ is defined as the global answer $g\_ans_i(q(\mathbf{x}), \mathcal{G})$ restricted to the J-bounded models, for some $i \in J$.

Theorem 1. $g\_ans_J^B(q(\mathbf{x}), \mathcal{G}) = b\_ans_J(q(\mathbf{x}), \mathcal{G})$

Theorem 1[14] formalises the intuition that the answer of a query submitted on $\mathcal{G}$ in the bounded mode is only computed by using the local information available in the graphs in J.

4.2 The Navigational Mode

In the navigational mode, a query $q(\mathbf{x})$ is submitted at a certain resource i, and it is answered by merging all the graphs in $\mathcal{G}$ that are reachable from g_i by following foreign references.

To define the semantics of the navigational mode we first need to formalise the reachability relation between graphs. This is based on the notion of foreign reference.

Definition 15 (Local and foreign URI reference)**.** The occurrence of $i : x$ in the graph g_j is a *local reference* if $i = j$, a *foreign reference* otherwise.

Definition 16 (Reachable graph)**.** Given a graph space $\mathcal{G}$, g_j is *directly reachable* from g_i, denoted by $i \rightarrow j$ if i contains a foreign reference to j. g_j is *reachable* from g_i, in symbols $i \xrightarrow{*} j$ if there is a sequence $i = h_1, h_2, h_3 \ldots, j = h_n$ such that $h_k \rightarrow h_{k+1}$ for $1 \leq k \leq n-1$. For any $i \in I$, $i^* = \{j | i \xrightarrow{*} j\}$.

Definition 17 (Navigational answer)**.** The *navigational answer* of the query $q(\mathbf{x})$ submitted at i is

$$n\_ans_i(q(\mathbf{x}), \mathcal{G}) = ans(q(\mathbf{x}), \mathsf{merge}_{j \in i^*}(g_j))$$

Intuitively the definition of navigational answer says that to answer a query on a graph g_i, first one needs to collect all the information that can be reached by following the links originating from g_i (i.e., we compute $\mathsf{merge}_{j \in i^*}(g_j)$, and then we submit the query on this extended dataset.

Definition 18 (Navigational model)**.** M is a *navigational model* if it is a model for $\mathcal{G}$ and for all $i, j \in I$ with $i \xrightarrow{*} j$, then

$$r_{ji}((j : x)^{\mathcal{I}_j}) = (j : x)^{\mathcal{I}_i}$$

$g\_ans_i^N(q(\mathbf{x}), \mathcal{G})$ is defined as the global answer $g\_ans_i(q(\mathbf{x}), \mathcal{G})$ restricted to the navigational models.

Theorem 2. $g\_ans_i^N(q(\mathbf{x}), \mathcal{G}) = n\_ans_i(q(\mathbf{x}), \mathcal{G})$

4.3 The Direct Access Mode

This semantics is based on the idea that we answer $q(\mathbf{x})$ by collecting all the graphs which contains a reference to a given URI (or collection of URIs), we merge them into a single graph and then we use the result of merging these graphs as the dataset against which the query can be processed.

Definition 19 (Direct Access answer)**.** The *direct access answer* at i of the query $q(\mathbf{x})$ over the graph space $\mathcal{G}$ is defined as follows

$$d\_ans_i(q(\mathbf{x}), \mathcal{G}) = ans\left(q(\mathbf{x}), \mathsf{merge}_{i \in I}(g_i)\right)$$

Intuitively the definition of direct access answer states that answering a query submitted to a graph g_i is the same as submitting the query to the entire (merged) graph space at once. Indeed from the definition 19 we immediately have that $d\_ans_i(q(\mathbf{x}), \mathcal{G}) = d\_ans_j(q(\mathbf{x}), \mathcal{G})$ for every $i, j \in I$; that is, the graph at which the query is submitted is irrelevant in the computation of the answer.

Definition 20 (Direct access model)**.** M is a *direct access model* if it is a model for $\mathcal{G}$ and for all $j : x$ and for all $i \in I$, $r_{ji}((j : x)^{\mathcal{I}_i}) = (j : x)^{\mathcal{I}_j}$.

$g\_ans_i^D(q(\mathbf{x}), \mathcal{G})$ is defined as the global answer $g\_ans_i(q(\mathbf{x}), \mathcal{G})$ restricted to the direct access models.

Theorem 3. $g\_ans_i^D(q(\mathbf{x}), \mathcal{G}) = d\_ans_i(q(\mathbf{x}), \mathcal{G})$

5 Discussion

In the Semantic Web community there is an ongoing and lively discussion on the contextual nature of URIs for entities on the Web[15]. This is a very interesting discussion, because it is based on the technical architecture of the Web and thus touches the heart of web-based KR (as opposed to non web-oriented KR).

There are two main parties. On the on had, several authors support the view that a the meaning of a URI is context-dependent, as choosing to use a URI for an entity somehow implies that one “endorses” a specific view on that entity (i.e. an identity), which is expressed in the set of statements about that URI that is accessible when a URI is dereferenced (see e.g. [Booth, 2008; Jaffri *et al.*, 2008]). In the other hand, other authors support the idea that URIs are context-independent rigid designators, which refer to the same real world entity in any possible description, and as such it provides a form of direct reference which is not equivalent (nor can be reduced) to any set of statements about it (see e.g. [Bouquet *et al.*, 2008a]). In other words, in the first view *reference is mediated by description*, and the latter is more

[14] The proof of this Theorem, and of the other Theorems presented in the rest of this paper can be found at `http://dkm.fbk.eu/index.php/Image:IR-KR-2009-TechRep.zip`.

[15] For more in this, see for example the IRW2006 (`http://www.ibiblio.org/hhalpin/irw2006/`), I^3 (`http://okkam.dit.unitn.it/i3/`) and IRSW2008 (`http://www.okkam.org/IRSW2008/`) workshops which were organized on this topic in conjunction with the most important Web and Semantic Web conferences of the last years, or the large number of papers on the identity of resources [V. and A., 2008; Halpin and Presutti, 2009], cool URIs (`http://www.w3.org/Provider/Style/URI`, `http://www.w3.org/TR/cooluris/`), identity crisis `http://www.ontopia.net/topicmaps/materials/identitycrisis.html`.

fundamental than the former; in the second view, reference is not necessarily mediated by any description, as *reference is a primitive and direct relation* between a real world entity and its identifier[16].

The practical consequences of these two views are very relevant for the development of web-based KR. The first view entails that people should not reuse an existing URI for describing an entity *unless* the intention is to endorse what is said about that entity when it is dereferenced; otherwise, a new URI should be minted and, optionally, linked to other existing URIs. The second view entails that the reuse of URIs should be maximized, as the URI by itself is an opaque identifier, which works as a global access key to information about that entity, no matter where this knowledge is stored.

This decision has an impact on the three modes we discussed for accessing knowledge on the GGG. Indeed, the first view is very much in line with the navigational view. When in a graph g one finds a mention of a foreign URI (or an identity statement connecting a local URI $i : x$ with a foreign URI $j : y$), this is viewed as a link which allows applications to jump from the first graph to the other. However, this way of linking entities seems to be justified only the basis of the condition we described in Section 4.2, namely that there is a domain relation connecting the interpretation of the two occurrences of the same URI in the two RDF graphs; otherwise the link is not logically justified. And this in turn seems to imply that the problem is not that the two entities are the same, but only that one wants to keep the different information sources distinct as they may be associated to different levels of trust.

The second view is for sure more oriented to information integration, and as such seems more consistent with the direct access view. If the same URI is used for the same entity in any collection of RDF graphs, then this is interpreted as the fact that all these occurrences directly refers to the same real world entity, and therefore all statements about that resource can be legitimately unified by graph merging. This of course is logically legitimate, but somehow begs the question of how “trusted” an entity’s URI is. In the previous approach, trust is mainly delegated to the DNS (a URI containing a trusted domain can be more reliable than a statement made in a less trusted domain); in the second, trust cannot be based on the domain name contained in the URI, but only on the domain name of graph itself (e.g. where it is physically stored).

Like the Web, the two views can perfectly coexist, and need not be thought of as mutually exclusive. They can easily be integrated, as nothing prevents developers from adding location-based URIs with an opaque URI which may act as a global key. However, both views heavily depend on the fulfillment of tough preconditions:

1. the first view needs a large number of links, e.g. the identity statements used in the Linking Open Data initiative;
2. the second view relies on the fat that a large number of independent developers have access to an opaque URI for the resources named in their content.

As to the first, it must count on the collaboration of users, which need to invest time in linking their local URIs with other external ones; as to the second, it seems to presuppose the availability of a service which can guarantee easy access to a repository of persistent opaque identifiers. In this respect, an important initiative is the OKKAM project[17], whose main goal is to deploy a global *Entity Name System* (or ENS) for supporting the creation and reuse of global, opaque and rigid identifiers [Bouquet *et al.*, 2008a].

References

[Booth, 2008] Davide Booth. Why uri declarations? a comparison of architectural approaches. In *Proceedings of the 1st International Workshop on Identity and Reference on the Semantic Web (IRSW2008)*, volume Volume 5021/2008 of *CEUR Workshop Proceedings*, Tenerife, Spain, June 2nd, 2008 2008.

[Bouquet *et al.*, 2008a] Paolo Bouquet, Heiko Stoermer, and Barbara Bazzanella. An Entity Name System (ENS) for the Semantic Web. In *The Semantic Web: Research and Applications. Proceedings of ESWC2008.*, volume Volume 5021/2008 of *Lecture Notes in Computer Science*, pages 258–272. Springer Berlin / Heidelberg, June 2008.

[Bouquet *et al.*, 2008b] Paolo Bouquet, Heiko Stoermer, Claudia Niederee, and Antonio Mana. Entity Name System: The Backbone of an Open and Scalable Web of Data. In *Proceedings of the IEEE International Conference on Semantic Computing, ICSC 2008*, number CSS-ICSC 2008-4-28-25 in CSS-ICSC, pages 554–561. IEEE Computer Society, August 2008.

[Ghidini and Serafini, 1998] Chiara Ghidini and Luciano Serafini. Distributed First Order Logics. In D. Gabbay and M. de Rijke, editors, *Frontiers Of Combining Systems 2 (Papers presented at FroCoS’98)*, Studies in Logic and Computation, pages 121–140. Research Studies Press/Wiley, 1998.

[Halpin and Presutti, 2009] Harry Halpin and Valentina Presutti. An ontology of resources: Solving the identity crisis. In D. Gabbay and M. de Rijke, editors, *Proceedings of ESWC2009*, Studies in Logic and Computation, pages 121–140. Research Studies Press/Wiley, 2009.

[Jaffri *et al.*, 2008] Afraz Jaffri, Hugh Glaser, and Ian Millard. Managing uri synonymity to enable consistent reference on the semantic web. In *Proceedings of the 1st International Workshop on Identity and Reference on the Semantic Web (IRSW2008)*, volume Volume 5021/2008 of *CEUR Workshop Proceedings*, Tenerife, Spain, June 2nd, 2008 2008.

[Kripke, 1972] S. Kripke. *Naming and necessity*. Harvard University Press, 1972.

[V. and A., 2008] Presutti V. and Gangemi A. Identity of resources and entities on the web. *International Journal on Semantic Web and Information Systems*, 4(2), 2008.

[16] See [Kripke, 1972] for a philosophical discussion of this thesis.

[17] See `http://www.okkam.org/`.

RDF-AI: an Architecture for RDF Datasets Matching, Fusion and Interlink

François Scharffe
Semantic Technology Institute, University of Innsbruck, Austria
francois.scharffe@uibk.ac.at
Yanbin Liu, **Chunguang Zhou**
College of Computer Science and Technology, Jilin University, China

Abstract

With the recent publication of large quantities of RDF data, the Semantic Web now allows concrete applications to be developed. Multiple datasets are effectively published according to the linked-data principles. Integrating these datasets through interlink or fusion is needed in order to assure interoperability between the resources composing them. There is thus a growing need for tools providing datasets management. We present in this paper RDF-AI, a framework and a tool for managing the integration of RDF datasets. The framework includes five modules for pre-processing, matching, fusing, interlinking and post-processing datasets. The framework inplementation results in a tool providing RDF datasets integration functionalities in a linked-data context. Evaluation of RDF-AI on existing datasets shows promising results towards a Semantic Web aware datasets integration tool.

1 Introduction

The Semantic Web is an evolution of the Web allowing machines to process data. Its foundations lies in the availability of structured data described using ontologies. Web datasets are structured data sources following the Semantic Web standards, and maintained by a single entity. Different datasets managed by different entities may offer similar contents. For example two datasets containing musical data, Musicbrainz [1] and Jamendo [2] overlap deeply. Many of the resources they describe refer to the same real-world objects. Overlaps between datasets will become usual as more and more Web datasets are published.[3]

On of the fundaments of the Semantic Web is the use of Uniform Resource Identifiers (URIs) to identify objects. The use of URIs assures that real world objects can be identified and referred to unambiguously. If many Web datasets describe resources using different URI schemes there is a need to indicate that two resources refer to the same real world object, even though they have a different URI. The following three approaches can be considered, ordered by increasing distributivity. **Merging datasets** together in order to have a unique URI assignment scheme. While it is not feasible at Web scale, merging datasets can be in some cases useful. We also consider this approach in the system described in this paper. **URIs equivalence servers** provide centrally maintained lists of equivalent resources. This approach is followed in [Bouquet *et al.*, 2008]. **Equivalence lists attached to datasets** are published by the datasets maintainers. Each dataset refer in this approach to other datasets containing similar resources. This approach is followed in [Jaffri *et al.*,].

In each of the aforementioned approach, equivalences between resources need to be given in order to either fusion datasets or build the equivalence lists. Given the large size of some datasets, it is not realistic to consider constructing resources equivalences manually. A more resonable approach consists in using a matcher to automatically detect them. We propose in this paper RDF-AI, a framework and a tool for automatically matching RDF datasets.

RDF-AI takes in input two datasets and generates in output either a new dataset resulting from the fusion of the two input datasets, or a list of correspondences between equivalent resources of the two datasets. RDF-AI architecture is modular, allowing to use any matching algorithm able to take RDF graphs as an input and to output alignments specified in the ontology alignment format.[4]

The contributions of this paper are as follows:

- An architecture for matching Web datasets
- A tool, RDF-AI, implementing this architecture
- A new resources matching algorithm

This paper is organized as follows. In Section 2, we overview related existing works. In Section 3, we present the system architecture and detail in Section 4 RDF-AI

[1] http://www.musicbrainz.org

[2] http://www.jamendo.com

[3] The linked datasets cloud gives an idea of the number of datasets availablehttp://esw.w3.org/topic/SweoIG/TaskForces/\CommunityProjects/LinkingOpenData

[4] http://alignapi.gforge.inria.fr/format.html

implementation and algorithms. In section 5, we test and evaluate the system on two pairs of datasets. Finally, in Section 6 we conclude and present new perspectives opened by the work presented in this paper.

2 Related work

We have a look in this section over related approaches and techniques related to the integration of RDF datasets. Our study is organized along two axes. We first overview the various approaches developped for similar problems, and then see which techniques are used to solve these problems.

Detecting the similarity between records inside and between relational data-bases is a well studied area. Two similar problems can be distinguished: a set of database records is analysed to detect duplicates in order to perform data cleansing; two sets of database records are analysed to detect similar records and perform record linkage. Record linkage can then be used to perform databases fusion. These areas have been largely studied both theoretically [Fellegi and Sunter, 1969], and technically (see [Elmagarmid *et al.*, 2007; Winkler, 2006] for recent surveys).

Matching RDF datasets is also closely related to the well studied area of ontology matching [Euzenat and Shvaiko, 2007]. Matching ontologies includes matching instances, but in this case only instances from a specific ontology are generally considered. RDF datasets matching on the contrary deals most of the time with datasets described using many ontologies.

Another area can be distinguished under the terms of identity recognition, instance unification or URI equivalence mining. Two approaches have been recently considered for the management of URI equivalence: The OKKAM project [Bouquet *et al.*, 2008] tries to tackle the issue by proposing Entities Name Servers that act as resources directories around common identifiers. Each real world entity is provided with a specific identifier, which is then linked to its various URIs. The approach in [Jaffri *et al.*, 2008] uses Consistent Reference Services finding equivalent URIs using equivalence lists assigned to every datasets. In both approaches there is a need for a system such as the one we present in this paper identifying equivalent resources.

These three areas make use of various techniques to perform the matching. String comparison techniques are necessary in all cases, sometimes using fuzzy methods [Chaudhuri *et al.*, 2003]. We perform string matching in RDF-AI using a sequence alignment algorithm based on dynamic programming [Rivasa and R.Eddy, 1999]. Other techniques relevant to the particular problem of RDF datasets matching are described below.

Specificities of matching datasets in a linked-data environmnent need to be considered. The distributed nature of resources makes data not necessarily known at the time of matching. Evidence must be acquired on-demand by dereferencing resources URIs, or by querying for their description using a SPARQL *Describe* query. A recent work on Web data interlinking and fusion [Raimond *et al.*,] exploits the graph and the Web based nature of Web datasets. The matching evidence is propagated through other resources via object properties, in a similar manner to the similarity flooding algorithm proposed in [Melnik, 2002]. This approach also tackles the problem of having large datasets available behind a SPARQL endpoint: a solution to reduce the size of the matching space by using an external query service is proposed.

Another set of techniques perform equivalence mining using ontology axioms [Hogan *et al.*, 2007; Saïs *et al.*, 2007; Nikolov *et al.*,]. In [Hogan *et al.*, 2007] inverse functional properties are used to find out about resources equivalence. Two resources are declared equivalent if they are both subject of an inverse functional property which range to the same object. The L2R method [Saïs *et al.*, 2007] uses a purely logic based approach using a set of predefined string equivalence logical facts, which are then combined with ontology axioms in order to deduce new facts about resources equivalence. A forward chaining algorithm is then used to propagate similarities. This method was recently combined with a numerical approach using string matching techniques [Saïs *et al.*, 2008]. In the Knofuss architecture [Nikolov *et al.*,], ontologies are used to perform a consistency checking on the dataset resulting from a fusion process.

Before describing in Section 4 the details of the matching algorithm, we present in the next section an architecture for Web datasets matching, fusion and interlink.

3 System architecture

RDF-AI architecture is composed of five independant modules allowing to pre-process, match, fusion, interlink and post-process RDF datasets. Inter-modules communication is realized using standard representation formalisms. The architecture overview picture is not given here for space reasons, it is available at `http://www.scharffe.fr/pub/ir-kr-2009/rdf-ai-architecture.pdf`

The pre-processing module performs operations on the datasets in order to prepare them for the process. The matching module takes two datasets as an input and returns an alignment between them. The interlinking module takes an alignment in input and returns a graph containing a set of *owl:sameAs* statements between resources of the two input graphs. The fusion module takes an alignment in input and returns a graph containing a new dataset resulting of merging the two input graphs. The post-processing module takes in input a graph resulting from the fusion module, check its consistency and process it for publication. Each module input includes a set of parameters given by the user.

Modules being independent from each other they can easily be interchanged. This feature is particularly important for allowing to use various matchers. We present in the following each module in detail.

Preprocessing This module is concerned with preprocessing operations preparing the source datasets for the matching process. Inputs are G_1 and G_2 the datasets to be matched and a set of parameters.Output s are $G_1^{'}$ and $G_2^{'}$ the processed input graphs and a report.

The list of possible operations for this module is given below. The user selects which operation needs to be performed using the module input parameters.

Checking The module checks if the input datasets are consistent with regard to their ontologies. It also checks that every resource is typed. This phase might trigger other operations if checks fail.

Materialization This operation consists in materializing RDF triples. For example materialization of inverse or transitive properties.

Translation This operation consists in translating given properties from one language to another.

Ontology evolution This operation consists in adapting a dataset to the other in the case one ontology is a more recent version of the ontology used for the other dataset.

Properties transformations These operations consist in modifying properties values of one or both datasets to prepare them for the matching process. For example, our system implements a foaf:name transformations changing “lastname, firstname” into “firstname lastname”.

The pre-processing module output two datasets corresponding to the two input datasets modified by the module operations. The new datasets are then passed to the matching module that will be used to detect similarities between their datasources.

Matching The matching modules takes in input two datasets and returns an alignment between their resources. Inputs are $G_1^{'}$ and $G_2^{'}$ the two datasets resulting from the preprocessing step. Parameters are given according to the matching system. They are used to tune the system by indicating properties relevance in the matching process. The output of the matching module is an alignment given in the alignment format [Euzenat, 2004] extended to represent more expressive correspondences [Euzenat *et al.*, 2007]. An alignment is represented as a set of *cells* containing correspondences between the resources of the two input datasets. The *measure* property of a cell indicates the degree of confidence the matcher gives to the correspondence between these two resources. The alignment format is the standard format for representating matching algorithms output in the ontology alignment evaluation initiative[5]. Using this format allows various matchers to be utilized in RDF-AI. The system then process the alignment in order to either interlink or fuse the datasets. These operations are described in the following two sections.

Interlink The interlink module takes in input an alignment between two datasets and outputs a named graph containing a set of interlinking primitives. Input is an alignment, which format was described in Section 3. The parameter of the module is a threshold above which a link will be created between two resources in the alignment. This threshold is compared to the *measure* property of the alignment. The output of the interlinking process is named graph containing a set of links between equivalent resources using the *owl:sameAs* property. The graph is described using the TriG syntax[6], as well as the Void vocabulary for describing Web datasets, and properties from the ontology alignment vocabulary[7].

A named graph containing a set of interlinks is typed as a *void:Linkset*. Each linkset is given two *void:target* properties referring to the datasets interlinked in this linkset. The *align:fromAlignment* and *align:threshold* properties refer to the alignment from which the linkset is generated and the threshold used during the generation. The linkset can then be used in linked-data applications.

Fusion The fusion module takes in input the alignment and the two original datasets and returns a new dataset corresponding to the result of merging them. The module takes an alignment as input, as well as parameters. The set of parameters allow the user to control how the fusion is performed. A *source dataset* and an *extension dataset* are given: the source dataset will be extended with properties that the extension dataset does not include. Properties appearing in both datasets are either fusioned or duplicated according to the user configuration. The output of the fusion module is a new graph resulting from this fusion process.

We will describe the detail of the fusion module implementation in RDF-AI in Section 4. Further processing of the the resulting graph is performed in the post-processing module.

Post processing This module is concerned with processing of the linkset or the fused graph. It checks inconsistencies that may appear as a result of the fusion, for example breaking an ontology axiom. Input is a dataset G_3, and a report is generated in output indicating the results of checking the inconsistencies in the dataset.

We describe next in Section 4 the implementation of the architecture presented in this section.

4 Implementation

In this section, we detail the implementation of RDF-AI on an illustrating example which involves the preprocessing, matching, interlinking, fusion and output phases. We run the system on two datasets describing J.S. Bach[8] musical compositions and works.

[5] OAEI: http://oaei.ontologymatching.org

[6] http://www4.wiwiss.fu-berlin.de/bizer/TriG

[7] http://semanticweb.org/wiki/VoiD, http://www.omwg.org/TR/d7/

[8] http://en.wikipedia.org/wiki/Bach

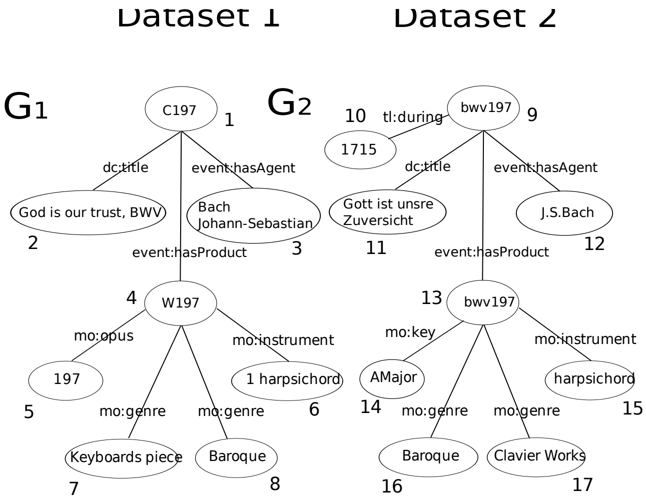


Figure 1: Example datasets

Preprocessing In the preprocessing module, RDF-AI integrates functionalities mentioned in Section 3 of this paper. RDF-AI uses the Jena framework[9] to load the ontologies and RDF files. The preprocessing step includes the following operations:

1. Translation function: the system translates literal content for the properties given in the parameters. The implementation of this translation function is performed using the Google Translate API.[10]

2. Name reordering: RDF-AI can automatically adjust family and given names order. In the implementation, we cache a list of surnames[11] and use this data to harmonize names. In our example, "Bach Johann Sebastian" will be reordered to "Johann Sebastian Bach".

Consider the two datasets represented in Figure 1. Each box corresponds to a resource, each ellipse to a literal, and edges to properties. The parameters for the preprocessing step are shown in the following code snippet, where the first parameter named "translation" makes RDF-AI translate the German literal content into English for the *dc:title* property in G_2, and the second parameter automatically adjusts the person name to the format "given name, family name" in G_1 dataset.

After this step, we obtain the modified graphs $G_1^{'}$ and $G_2^{'}$, nodes 3 and 11 were modified according by the pre-processing operations. This step homogeneize the datasets in order to optimize the efficiency of the matching algorithm described below.

Detailed matching RDF-AI allows to reduce the number of resources to compare during the matching phase. An initial query is used to return only those resources having a certain property value. In the Bach example, the matching space can be reduced to only those compositions having the same value for the "tl:during" property. This approach makes the fusion of large-scale datasets more efficient.

The matcher computes similarity values for resources in the datasets graphs according to the similarity of their comparable properties. RDF-AI actually includes two similarity computation algorithms to be selected by the user: a fuzzy string matching algorithm based on the sequence alignment algorithm [Rivasa and R.Eddy, 1999] and a word relations algorithm. There are two implementations to the latter: a synonyms comparison algorithm based on WordNet [Fellbaum, 1998] and a taxonomical similarity algorithm based on SKOS.[12] These two algorithms can be used in combination in the case more evidence is necessary to compute the similarity values.

Continuing the example, giving the graphs $G_1^{'}$ and $G_2^{'}$, we compute the similarity values for all possible pairs. Because the example graph is small, we do not need to use the initial SPARQL query reducing the matching space. The parameters are shown in the file. The "*string comparison*" value of the "*method*" field denotes that this property is compared using the string comparison algorithm, "*SKOS*" is the taxonomical similarity algorithm based on SKOS and "*WordNet*" the synonym comparison algorithm based on WordNet.

The algorithm enumerates every possible matching properties of (R_1,R_2) with the similarity value computed by the similarity matching algorithm. The normalized values of the string comparison algorithm are obtained by dividing each value by the maximal absolute value. The returned value of the word relation algorithm belongs to [0, 1], it differs from the string comparison algorithm because it is unsuited to quantitatively compute the semantic dissimilarity of two words. However, this difference will not affect the results.

Next, the algorithm selects the candidate property pairs to compute the resources similarity according to the normalized value. In the example, $(2, 11)$, $(3, 12)$, $(4, 13)$, $(6, 15)$, $(7, 17)$ and $(8, 16)$ are selected as their values are the most relevant for the graph similarity computation. The algorithm selects $(7, 17)$ and $(8, 16)$ for *mo:genre* because $S^0(8, 16)$ is the biggest in the *mo:genre* set $\{S^0(7, 16)$, $S^0(7, 17)$, $S^0(8, 16)$, $S^0(8, 17)\}$. It will be selected at first, and then all others matching pairs including 8 or 16 will be marked as invalid. The second valid candidate is thus $S^0(7, 17)$.

Then, the algorithm computes the similarity value of the resources (R_1,R_2). This method considers the co-affection of resources at different levels:
$S(4, 13) = (S^0(4, 13) + S^0(6, 15) + S^0(7, 17) + S^0(8, 16))/4 = 0.625$
$S(1, 9) = (S^0(2, 11) + S^0(3, 12) + S(4, 13))/3 = 0.732$
$S(R_1, R_2) = S(1, 9) = 0.732$

[9] http://jena.sourceforge.net

[10] http://code.google.com/apis/ajaxlanguage/

[11] Obtained from http://www.surnamefinder.com

[12] http://www.w3.org/2004/02/skos

Finally, the algorithm selects the the highest similarity measure and include it in the alignment, it is (R_1,R_2) in this example.

The overall space complexity of the matching algorithm is in $O(n)$, for n resources. It corresponds to the size occupied to store the dataset graph. The time complexity is in $O(n^2)$ in the worst case.

In the rest of the process, RDF-AI uses the alignment output in order to either generate a linkset or fuse the two datasets.

Interlinking and fusion The interlinkink module generates a linkset as a named graph including a set of *owl:sameAs* statement, according to the input alignment. The only parameter here is the threshold, set by the user, which is used to trigger the correspondences in the alignment to be included in the linkest. The correspondence is outputted as an *owl:sameAs* triple if its *measure* property value is bigger than the threshold, otherwise, this pair will not appear in the output graph. In this example, we set the threshold 0.5, and the *measure* statement value of (R_1, R_2) is 0.732, it will thus appear in the output linkset. Here, The URI of the matched resource of "c197" expressed by R_1 in Figure 1 is `http://bach1.example.org/composition/composition-197`, and R_2 is `http://bach2.example.org/composition/composition-bwv197`.

The fusion process fuses datasets according to the alignment, the original datasets graphs and a set of user parameters. It outputs the fused graph as a new dataset.

In this example, G_1 is the source dataset and G_2 is the extension dataset, the *namespaces* in the *add* node are the properties added to the source dataset, and the *namespace* in the *merge* node are the property merged into the source dataset from the extension dataset. In this case, the merged property value is kept from the source graph. The fused graph is the final result, the example output is shown Figure 2. Node 10, 11 and 14 are added, node 17 is merged to the *mo:genre* property, node 16 is ignored in Figure 1, because its property has same value with the node 8.

In order to evaluate the quality of the matches returned by RDF-AI, we have evaluated the tool on two pairs of overlapping datasets. We present the results of this evaluation next in Section 5.

5 Experimental results

In order to evaluate the performance of RDF-AI in matching and fusing RDF datasets, we have tested the system on the following dataset pairs from two different domains:

1. AKT EPrints archive and Rexa datasets[13].
2. The works of Johann Sebastian Bach in two different datasets [14].

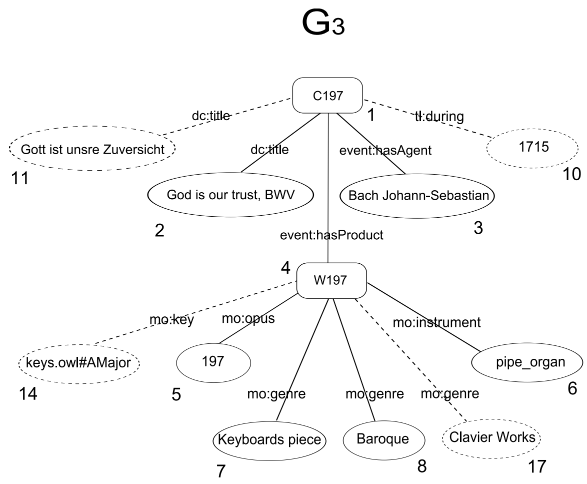


Figure 2: Fused graph

EPrints and Rexa are both described using the same ontologies. Authors are described using FOAF, publications are described using the Opus ontology[15]. The EPrints dataset contains 314 resources and Rexa 2103 resources. RDF-AI loads them and run according to the configuration file specifying the resources to match and the operation to be performed: matching, fusion, interlink. RDF-AI obtains on these datasets a precision of of 95.9% which is higher than the one obtained by the KnoFuss architecture (92%) on the same datasets [Nikolov *et al.*,]. The optimal configuration uses the name ordering function in the preprocessing module, compared with the 92% not using the function, the precision is much improved.

Two datasets on Bach musical works datasets are mainly described using the Music ontology as well as the timeline and events ontologies (See Section 1). In this experiment, the system matches resources of the composition class, which contains 771 resources in the first dataset, and 800 of the second. The optimal precision is 97.5%. The optimal configuration uses the translation function for the *dc:title* property, from German into English. If we do not use this function, the precision drops to 87.3%. The change is significant, and is easily understandable as the title is the most important property to distinguish the different objects. Without the translation, there are significantly more underived resources, the reason is that the similarity value becomes less than the fixed threshold. If we replace WordNet with a string comparison algorithm for the *mo:genre* computation, the precision only drops from 0.1 point. This small improvement is due to the fact that the evidence gained by computing other properties is enough to find the matches. However, the word relation algorithm significantly improves the similarity value in the alignment through the *measure* property of correspondences.

[13] `http://eprints.aktors.org,http://www.rexa.info`

[14] `http://www.scharffe.fr/pub/ir-kr-2009/`

[15] `http://knoesis.wright.edu/library/ontologies/\swetodblp/august2007/opus_august2007.rdf`

This evaluation of the system shows promising results for matching Web datasets. However, the system would need to be evaluated on larger datasets that cannot be entirely loaded at runtime. We discuss this issue together with others and conclude in the following section.

6 Conclusion and future work

We have presented in this paper RDF-AI, an architecture and a tool tackling part of the fundamental problem of Web datasets interoperability. The architecture provides the basic workflow and specifies data exchange formats at every steps of the matching, interlinking and fusion process. RDF-AI was succesfully evaluated on matching two pairs of Web datasets. User input is required at each step in order to configure the tool optimally. We are actually improving it so that the user input is reduced to the minimum. We particularly plan to work on an algorithm automatically acquiring the datasets structure. We are currently implementing some of the missing functionalities of RDF-AI: usage of ontology axioms in order to derive new matches, and concistency checking of the output of the fusion process.

Acknowledgment

This research works are supported by European Commission under grant No.TH /Asia Link/010 (111084) and under the EASAIER project (EU-FP6-IST-033902).

References

[Bouquet *et al.*, 2008] Paolo Bouquet, Heiko Stoermer, and Barbara Bazzanella. An Entity Naming System for the Semantic Web. In *Proceedings of the 5th European Semantic Web Conference (ESWC2008)*, LNCS, June 2008.

[Chaudhuri *et al.*, 2003] Surajit Chaudhuri, Kris Ganjam, Venkatesh Ganti, and Rajeev Motwani. Robust and efficient fuzzy match for online data cleaning. In *SIGMOD '03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 313–324, New York, NY, USA, 2003. ACM.

[Elmagarmid *et al.*, 2007] A.K. Elmagarmid, P.G. Ipeirotis, and V.S. Verykios. Duplicate record detection: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 19(1):1–16, January 2007.

[Euzenat and Shvaiko, 2007] Jérôme Euzenat and Pavel Shvaiko. *Ontology Matching*. Springer-Verlag, Heidelberg (DE), 2007.

[Euzenat *et al.*, 2007] Jérôme Euzenat, François Scharffe, and Antoine Zimmermann. D2.2.10: Expressive alignment language and implementation. Project deliverable 2.2.10, Knowledge Web NoE (FP6-507482), 2007.

[Euzenat, 2004] Jérôme Euzenat. An API for Ontology Alignment. In Frank van Harmelen, Sheila McIlraith, and Dimitri Plexousakis, editors, *The Semantic Web - ISWC 2004: Third International Semantic Web Conference,Hiroshima, Japan, November 7-11, 2004. Proceedings*, volume 3298, pages 698–712. Springer, 2004.

[Fellbaum, 1998] Christiane Fellbaum. *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*. The MIT Press, May 1998.

[Fellegi and Sunter, 1969] Ivan P. Fellegi and Alan B. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64(328):1183–1210, 1969.

[Hogan *et al.*, 2007] Aidan Hogan, Andreas Harth, and Stefan Decker. Performing object consolidation on the semantic web data graph. In *In Proceedings of 1st I3: Identity, Identifiers, Identification Workshop*, 2007.

[Jaffri *et al.*,] Afraz Jaffri, Hugh Glaser, and Ian Millard. Uri disambiguation in the context of linked data. In *Proceedings of the Linking Data On the Web workshop at WWW'2008*.

[Jaffri *et al.*, 2008] Afraz Jaffri, Hugh Glaser, and Ian Millard. Managing uri synonymity to enable consistent reference on the semantic web. In *IRSW2008 - Identity and Reference on the Semantic Web 2008 at ESWC*, 2008.

[Melnik, 2002] Sergey Melnik. Similarity flooding: a versatile graph matching algorithm. In *Proc. 18th International Conference on Data Engineering (ICDE), San Jose (CA US)*, 2002.

[Nikolov *et al.*,] Andriy Nikolov, Victoria Uren, Enrico Motta, and Anne de Roeck. Handling instance coreferencing in the knofuss architecture. In *Proceedings of the workshop: Identity and Reference on the Semantic Web at 5th European Semantic Web Conference (ESWC 2008)*.

[Raimond *et al.*,] Yves Raimond, Christopher Sutton, and Mark Sandler. Automatic interlinking of music datasets on the semantic web. In *Proceedings of the Linking Data On the Web workshop at WWW'2008*.

[Rivasa and R.Eddy, 1999] Elena Rivasa and Sean R.Eddy. A dynamic programming algorithm for rna structure prediction including pseudoknots. *Journal of Molecular Biology*, 285:2053–2068, 1999.

[Saïs *et al.*, 2007] Fatiha Saïs, Nathalie Pernelle, and Marie-Christine Rousset. L2r: A logical method for reference reconciliation. In *AAAI*, pages 329–334, 2007.

[Saïs *et al.*, 2008] Fatia Saïs, Nathalie Pernelle, and Marie-Christine Rousset. Combining a logical and a numerical method for data reconciliation. *Journal of Data Semantics*, 12, 2008.

[Winkler, 2006] W.E Winkler. Overview of record linkage and current research directions. Technical Report 2006-2, Statistical Research Division. U.S. Census Bureau, 2006.

From unstructured to linked data: entity extraction and disambiguation by collective similarity maximization

Tadej **Štajner**
Jozef Stefan Institute
tadej.stajner@ijs.si

Abstract

In this paper, we describe a pipeline of methods for identifying and resolving entities from unstructured data using a semi-structured background knowledge database. For this purpose, we employ named entity extraction, co-reference resolution and investigate performance of disambiguation using collective maximization of inter-entity similarity, compared to using only pair-wise disambiguation. We explore possibilities of using DBpedia and Yago as background knowledge databases with the goal of annotating unstructured text documents with global entity references.

1 Introduction and motivation

Annotating plain-text documents is an important building block in extracting knowledge from unstructured data. Identifying entities from text has always been a challenge due to ambiguities in names, adding uncertainty in the process of linking those names to real-world objects without having an explicit global identifier key. Representing in-text surface forms as globally identified entities is therefore crucial for integrating unstructured data with structured and semi-structured databases. Having such representations opens up new possibilities in semantic search, navigation, visualization – any information retrieval task that requires or makes use of data, more explicit than just pure plain text.

The problem is defined with the following: on one side, we have unstructured data, from which we want to extract named entities. On the other side, we have our background knowledge, representing real-world objects as entities, described with attributes. Our mission is to find out which objects are mentioned in the document.

The general approach usually requires solving the following sub-problems: identifying named entities from a given document, resolving co-references between them, retrieving potential candidate concepts from our background knowledge and selecting the most appropriate ones that represent the in-text entities. Word sense disambiguation can be viewed as a special entity resolution problem, where we represent word meanings as entities in our background knowledge.

In this paper, we will focus on collective disambiguation, which means that we don't handle each disambiguation case independently but instead focus on using the added information about previous disambiguation choices in a document to make further decisions more accurate. This paper will discuss a collective disambiguation method of employing the inter-entity similarity between candidate entities in a document where maximizing inter-entity similarity should improve disambiguation performance. The goal of this paper is to explore the possibilities of leveraging linked data in this scenario.

2 Related work

Such approaches, as proposed in [Bhattacharya and Getoor, 2005], are more often found in structured data, whereas our approach attempts to use these techniques on linking unstructured text with semi-structured data.

Cucerzan [2007] has already shown good performance in disambiguation with using Wikipedia as an example of a background knowledge source and linking anchors to Wikipedia concepts via document vector space similarity.

The identification and disambiguation problem can also be formulated as an entity resolution problem. The mathematical model is defined by Fellegi and Sunter [1969] and proposes using entity characteristics to decide whether two entities represent the same object with a certain probability. In our case, one of the entities in this pair-wise comparison is an unknown named entity, detected in a body of text via named entity extraction. The other one is a candidate from our background knowledge, which can be represented by the extracted surface form. Our analogy to comparing characteristics in the basic model is comparison of vector space model representations of the given document with a description of the candidate concept.

An assumption, often made in disambiguation tasks, is that given a single document, an anchor from that document will usually have neither multiple meanings per discourse, nor multiple meanings per collocation [Yarowsky, 1995]. However, it may be represented by multiple different aliases in a scope of a document. Therefore, resolving co-references within a document is an important step in disambiguation, as it likely reduces the possible solution space. [Mann and Yarovsky, 2003] also presented a specialized disambiguation approach for person resolution, based on bibliographic features appearing in documents.

Li et al. [2005] proposed an approach using a machine learning approach to feature extraction from entity pairs to predict co-occurrences of entities in a given context.

Our approach uses not only pair-wise disambiguation of an anchor and its' entity candidates, but also inter-entity similarities to do collective disambiguation, similar to an attribute similarity based graphical approach as defined in [Bhattacharya and Getoor, 2007]. Other graphical approaches for entity disambiguation with an emphasis on inter-entity relations were already proposed, such as [Mihalcea, 2005] using PageRank as a quality estimate and an adaptive approach by [Chen *et al.*, 2007]. [Lloyd *et al.*, 2005] proposed linguistically enhancing feature vectors for a high-precision disambiguation implementation. However, their implementation merges entities by clustering, whereas we have explicit candidate entities to merge the extracted anchor with. Another clustering-based approach was proposed by [Schütze, 1998], while some work has already been done on ontological approaches, such as [Dill *et al.,* 2005].

3. Problem formulation

To reformulate this problem as an entity resolution scenario, we must first consider the preprocessing steps, which are necessary for identifying local entities. After local entities are identified, we perform entity resolution with the background knowledge, assigning global identifiers to entities.

3.1 Entity extraction

Our first step in the disambiguation pipeline is identifying anchors in text that could represent an entity. The entity extractor also classifies entities by their type, e.g. person, organization or location, providing at least some information on in-text entities.

3.2 In-document entity co-reference resolution

Before using our background knowledge base, we can still perform a part of co-reference resolution with the identified entities. At this stage, we perform the following tasks:

- Canonicalization: Person names are converted to a common form with prefixes or suffixes, for example "Dr. Smith, John", whereas organization names are appended with the full suffix such as "ACME Incorporated". This is important for further name comparison.
- Partial name consolidation: For instance, if a documents refers to persons "John Smith" and "Mr. Smith", they likely represent the same entity, provided that "John Smith" is the only "Smith" in the text.
- Initials and acronym consolidation: "J. Smith" is joined with "John Smith", provided that there is no other candidate for that alias in the document. This is true if we follow the assumption that an ambiguous anchor takes on a single meaning per document.
- Simple attribute extraction: The "Mr." prefix also gives us indication that "John Smith" is a male.

However, this approach does not completely resolve co-references, especially in case where an entity's alias cannot be trivially matched to another alias without any background knowledge, such as matching anchors of "Barack Obama" and "President of United States" to a common entity.

The purpose of this simple co-reference resolution is to learn all what we can about anchors, so that we can perform better comparison further on. Simple de-duplication of extracted entities also helps to reduce the search space when performing collective disambiguation.

3.3 Candidate search

Once we have a basic understanding of which distinct entities we are trying to further identify, we can search our background knowledge for possible candidates that could match the named entities.

This means that we will have to be able to query our background knowledge database with an alias, expecting to return all entities that can be referred to by that alias. Our data model specifies entities as objects having:

- uniform resource identifiers
- aliases - *phrases that can refer to that entity – a single alias may refer to multiple entities;*
- descriptions – *documents that describe the entity;*
- other attributes, such as entity type, references to other entities and more;

In practice, this means that we retrieve candidate entities by querying the background knowledge with each extracted named entity. This gives us the initial search space over which we then perform candidate selection.

3.4 Entity resolution

In the pair-wise scenario, we evaluate all links between an extracted entity and possible candidates. In our case, given a vector space model representation of the article text, we compute for each candidate the cosine similarity of TF-IDF entity candidate concept vectors with the article vector.

Another dimension, on which we are able to perform comparison on, is entity type. On the anchor side we use the entity type as classified by the entity extractor. We then compare it to the candidate entity type, as specified by the *wordnet_type* attribute, as defined from Yago.

The extracted entity is then matched with the candidate with the greatest similarity with the article.

3.5 Collective resolution

The premise of collective resolution is that we can use attribute and relational knowledge of candidate entities to determine, which *fit* into the document the most, not treating each resolution decision as independent. Since our case does not consider relational data, the fitness criterion is defined with maximizing inter-candidate similarity, which can also be formulated as maximizing intra-cluster similarity within a cluster of entities. In other words, given multiple possible candidate entities for a given anchor text, the most likely correct one is the one that is most similar to other selected candidates so that ideally, the correct candidates would have maximum intra-cluster similarity.

[Nguyen and Rayward-Smith, 2008] found that different intra-cluster similarity measures behave differently in regard to various cluster properties, such as density. In our case, maximizing cluster density relates well to candidate selection.

For all possible combinations of selections of candidates, the one where the candidates are all from the same topical context will have the highest intra-cluster similarity. However, more than one peak can exist when multiple ambiguous anchors have meanings on both topical contexts. This is expected to happen rarely and is further alleviated by combining this method with the basic pair-wise comparison.

This is the *Context Attraction Principle*, as described by [Kalashnikov and Mehrotra, 2005], which says that if anchor r made in the context of entity x refers to an entity y_j whereas the description provided by r matches multiple entities $y_1, y_2, . . . , y_j, . . . , y_N$, then x and y_j are likely to be more strongly connected to each other than x and y_i, where $i = 1, 2, ... , N$ and $i \neq j$. In our case, the connection strength can be measured as the aforementioned linear combination of attribute and text similarity.

Since the candidate entities in our background knowledge have significant additional data besides their vector space representations of descriptions, we include attribute and comparison in the computation of similarity score. In this case, we use the Jaccard coefficient comparison to measure the closeness between two entities.

3.6 Disambiguation

We perform disambiguation and compare results for two methods.

The baseline is selection of candidates via maximum pair-wise similarity of anchor and the candidate entity. The performance of this method has already been established in multiple publications [Cucerzan, 2007].

We propose to enhance this method by including a measure of maximum collective similarity of candidate entities. The main difference from the baseline method is that whereas pair-wise comparison treats each entity selection as independent, this method evaluates multiple entity resolutions at once. For example, distinguishing between candidate entities “London, UK” and “London, Ontario” for the anchor “London” is clearer once we include the fact that the anchor “UK” could resolve to “United Kingdom”, which in turn is much more similar to “London, UK” than “London, Ontario”. If we make this decision for each ambiguous anchor, we are presented with a problem of finding the best decisions for each ambiguous anchor.

The criterion for selecting the right set of decisions is maximizing inter-entity similarity. In other words, since the Context Attraction Principle suggests that the correct entities are somewhat more similar to each other than incorrect ones, we favor those decision sets which suggest entities with high similarity.

Given a anchor s_i, we are presented with a candidate entity vector (e_{i0}, ... e_{in}), n being the index of candidates. For pair-wise disambiguation of anchor s, we select entity $e_{selected}$ as:

$$e_{selected} = argmax_i(entsim(s, e_i))$$

Where similarity is defined as a linear combination of description similarity and attributes similarity.

$$sim_{ent}(e_i, e_j) = \alpha \cdot sim_{desc}(e_i, e_j) + (1 - \alpha) \cdot sim_{att}(e_i, e_j)$$

The similarity measure sim_{desc} is defined as cosine similarity of TF-IDF vectors of the document and description of entity e_i. In case of comparing an anchor to an candidate entity, the only attributes we can perform comparison on are the entity type (location, person, organization) and possibly gender, if the entity type is a person.

For collective disambiguation we want to solve the problem of selecting the best set of entities, each belonging to their respective anchor s_m. so that we maximize the intra-cluster similarity of the cluster C of m selected entities, where C always contains for each anchor exactly one candidate entity per corresponding anchor. Since we favor dense clusters, we choose a selection quality measure, which exhibits such behavior. As shown [Nguyen and Rayward-Smith, 2008], one such measure is average intra-cluster similarity:

$$sim_{intracl}(\mathrm{C}) = \frac{\sum_{\mathrm{e_i} \in \mathrm{C}} \sum_{\mathrm{e_j} \in \mathrm{C} \wedge \mathrm{e_i} \neq \mathrm{e_j}} sim_{ent}(\mathrm{e_i}, \mathrm{e_j})}{|\mathrm{C}|^2}$$

Each e_i represents a candidate for the i-th anchor. This measure computes the intra-cluster similarity of selected entities.

Description similarity is measured in the same way as comparing the document with the entity description in the pair-wise disambiguation. Attribute similarity is a Jaccard coefficient of attributes of both entities, which have more information than just entity type or gender, for instance references to other entities and Yago categories.

For our final entity selection, we compute each candidate’s score by combining both resolution results – the pair-wise similarity of anchor a and entity e_i with the intra-cluster similarity of the best cluster C having that candidate.

$$score(e_i, a) = sim_{ent}(e_i, a) \cdot max_{\mathrm{C}, e_i \in \mathrm{C}}(sim_{intracl}(\mathrm{C}))$$

In other words, we favor those entities which have high similarity with the document, as well as high similarity with each other.

4. Materials

4.1 Background knowledge

Our knowledge database should contain enough data to be able to perform the following tasks: first, it should be able to refer to each entity with multiple aliases to facilitate candidate retrieval, and second, it should be able to provide enough additional entity features, which we can use to compare those entities to each other and to article anchors that we attempt to link to.

Following these requirements, we chose to use a part of DBpedia, as described in [Auer *et al.*, 2007], for the fact that it provides both description and attribute data from Wikipedia, as well as references to foreign anthologies that describe other aspects of the same real-world objects. One of the additional attributes that we wanted to consider is entity type, comparable to the type, classified by entity extraction. This in turn gives us another attribute on which we can compare anchors with candidate entities. This data was obtained from the Yago ontology, defined in [Suchanek et al., 2008], which maps Wikipedia concepts to corresponding WordNet classes. Since a direct mapping from Yago to DBpedia exists, merging the two together is trivial.

However, both ontologies are much broader than what our approach requires – we currently only use information on aliases, textual descriptions, *rdf:type* attributes and Yago categories of entities.

Another crucial piece of information are the aforementioned entity aliases. These are obtained from DBpedia's redirect and disambiguation assertion collections: if an entity redirects to another one, the first entity's name is just an alias for the second one. On the other hand, DBpedia identifies multiple meanings of entities from Wikipedia disambiguation pages, which also provide us with potential aliases.

From the perspective of entity resolution, having those aliases enables us to define a reasonable solution space, from which we select the best candidate for linking.

4.2 Implementation

The pipeline consists of an entity extraction and coreference resolution component, implemented in Java. Named entity extraction is performed with a Conditional Random Field classifier using the Stanford Named Entity Recognizer [Finkel et al. 2005].

The component which transforms the corpus into a TF-IDF vector space model itself is implemented in C++, as is also the case with the entity resolution component, enabling comfortable response time for enriching an article with entity references.

For each article, the following sequence of steps is performed:

- Named entity recognition
- Co-reference resolution
- Pair-wise candidate entity evaluation
- Collective candidate entity evaluation (optional)
- Final candidate entity selection by maximum score

5. Evaluation

We perform evaluation using the New York Times article corpus [Sandhaus, 2008], using 57444 articles from October 2006 to April 2007 as training data for construction of TF/IDF weighted vectors. The articles were then processed with an implementation of the described algorithm. For evaluating the performance of different approaches we manually evaluated 693 entity resolution decisions from 50 articles as correct or incorrect. We compared two approaches: the baseline approach is performing disambiguation with using cosine similarity between candidate description and the article itself, whereas our approach performs an additional step of collective disambiguation.

As a performance metric, we use the F_α with $\alpha = 0.2$ and 1.0.

$$F_\alpha = \frac{(1+\alpha) \cdot precision \cdot recall}{\alpha \cdot precision + recall}$$

In some applications we want to rate precision higher than recall, as false positives are much less desired than false negatives. Therefore, we provide results for both α values.

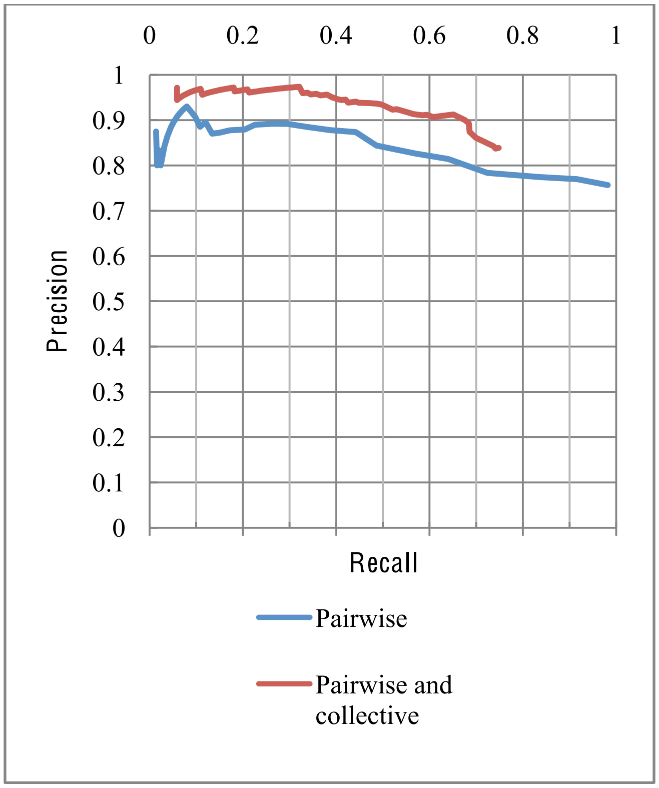


Figure 1 - Precision/recall performance

	$F_{1.0}$	$F_{0.2}$
Pair-wise	0.864	0.790
Pair-wise + intra-cluster similarity max.	0.790	0.855

Table 1: F-measure performance

Results from Table 1 show that additionally using collective disambiguation by intra-cluster similarity maximization show consistently better results with $F_{0.2}$. The collective method obtains a maximum of 0.855, compared 0.790 by only using the pair-wise method. On the other hand, the pair-wise-only approach performs better on some high-recall situations, although that is more of an exception, as Figure 1 shows that the collective approach achieves overall higher precision. This suggests that intra-cluster similarity maximization appears to bring additional disambiguation accuracy over the baseline, but there is still room for improvement, as some articles only mention certain entities, but are as a whole quite dissimilar to the entities' descriptions.

However, collective similarity maximization is a computationally very intensive problem, as it involves calculating average similarity over all combinations of candidates for anchors. Its complexity is growing exponentially with the number of ambiguous anchors, which requires the combination of using pair-wise similarity as a heuristic. For practical purposes, it makes sense to use the computationally cheaper pair-wise similarity score to estimate whether a candidate is even worth considering for further evaluation with intra-cluster similarity maximization, reducing the complexity to quadratic at the worst case.

6. Future work

One of the obvious future developments is improving the intra-cluster similarity maximization method by using a graph clustering algorithm and employing heuristics to reduce the candidate combination search space. We will also experiment with using different collective similarity measures, especially since the proposed one exhibits quadratic behavior in relation to anchor count. An interesting question is whether using a computationally cheaper metric with linear complexity, such as the average distance from the candidate entity centroid, is worth sacrificing over sensitivity to cluster density.

[Bhattacharya *et al.*, 2006] also proposed an adaptive approach, taking into account only potentially relevant relations. This shows that if explicit inter-entity relations are available, they can substantially improve resolution. Relational entity resolution is an interesting field of research which also has application on information extraction and is definitely worth pursuing.

Another source of information can be obtained by having entity co-occurrence statistics from the corpus, which can give a prior joint probability that tells us the likeliness that two entities occur together in an article, which can be a useful piece of knowledge in collective disambiguation. Similar approaches have been proposed in [Yarowsky, 1995] and [Li and Abe, 1998].

We believe that methods employing prior joint probabilities benefit most from bootstrapping the document corpus by performing entity extraction and disambiguation at first without considering co-occurrences. Those identified entities are then used as the first prior for joint probabilities
for the next pass over the corpus. This process can then be repeated iteratively with each pass having prior joint probability data available from the previous pass.

Another aspect, worth looking into, is exploiting relations in the semantic data as opposed to just expressing those relations as simple attributes. Given relations between entities, we can then employ more sophisticated methods of relational entity resolution, as proposed in [Bhattacharya and Getoor, 2007].

We can also further improve our background knowledge is to include relations between known entities, as proposed in [Lehmann et al. 2007].

7. Conclusion

Our work shows that adding entity disambiguation by coherence maximization can in fact improve disambiguation performance in comparison to plain pair-wise disambiguation, although with some additional computational complexity. Fortunately, there still exist ways in which we can further alleviate that problem. We also demonstrate the usefulness of using Linked Data to the task of collective disambiguation with including attribute similarity in coherence calculation.

On the other hand, our paper barely touches the possibilities that could be employed by using globally identified data approaches, opening way for better data integration, visualization and using annotated documents to enable semantic search. We expect that the proposed semantic article enrichment method to yield even more improvement on tasks that depend on the added semantic information, such as document summarization, triple extraction and recommendation systems.

8. References

[Cucerzan, 2007] S. Cucerzan. Large-scale named entity disambiguation based on Wikipedia data. In Proceedings of EMNLP-CoNLL, 2007.

[Yarovsky, 1995] David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting on Association For Computational Linguistics* (Cambridge, Massachusetts, June 26 - 30, 1995). Annual Meeting of the ACL. Association for Computational Linguistics, Morristown, NJ, 189-196.

[Mann and Yarovsky, 2003] Gideon S. Mann and David Yarowsky, 2003. Unsupervised personal name disambiguation. In *Proceedings of the Seventh Conference on*

Natural Language Learning At HLT-NAACL 2003 - Volume 4 (Edmonton, Canada). Human Language Technology Conference. Association for Computational Linguistics, Morristown, NJ, 33-40.

[Kalashnikov and Mehrotra 2005] Dmitri V. Kalashnikov and Sharad Mehrotra, 2006. Domain-independent data cleaning via analysis of entity-relationship graph. *ACM Trans. Database Syst.* 31, 2 (Jun. 2006), 716-767.

[Chen *et al.*, 2007] Zhaoqui Chen, Dmitri V. Kalashnikov, and Sharad Mehrotra, 2007. Adaptive graphical approach to entity resolution. In Proceedings of the 7th ACM/IEEE-CS Joint Conference on Digital Libraries (Vancouver, BC, Canada, June 18 - 23, 2007). JCDL '07. ACM, New York, NY, 204-213.

[Bhattacharya and Getoor, 2007] Indrajit Bhattacharya and Lise Getoor, 2007. Collective entity resolution in relational data. ACM Trans. Knowl. Discov. Data 1, 1 (Mar. 2007), 5.

[Bhattacharya *et al.*, 2006] Indrajit Bhattacharya, Lise Getoor, and Louis Licamele, 2006. Query-time entity resolution. In *Proceedings of the 12th ACM SIGKDD international Conference on Knowledge Discovery and Data Mining* (Philadelphia, PA, USA, August 20 - 23, 2006). KDD '06. ACM, New York, NY, 529-534.

[Nguyen and Rayward-Smith, 2008] Quynh H. Nguyen and V. J. Rayward-Smith, 2008. Internal quality measures for clustering in metric spaces. *Int. J. Bus. Intell. Data Min.* 3, 1 (Apr. 2008), 4-29.

[Li *et al.*, 2005] Xin Li, Paul Morie, Dan Roth, Semantic integration in text: from ambiguous names to identifiable entities, AI Magazine, v.26 n.1, p.45-58, March 2005

[Auer *et al.*, 2007] S. Auer, C. Bizer, J. Lehmann, G. Kobilarov, R. Cyganiak, and Z. Ives. DBpedia: A nucleus for a web of open data. In Proceedings of the 6th International Semantic Web Conference and 2nd Asian Semantic Web Conference (ISWC/ASWC2007), volume 4825 of LNCS, pages 715-728, Springer, 2007.

[Suchanek *et al.,* 2008] Fabian M. Suchanek, Gjergji Kasneci, Gerhard Weikum, YAGO: A Large Ontology from Wikipedia and WordNet, Web Semantics: Science, Services and Agents on the World Wide Web, Volume 6, Issue 3, World Wide Web Conference 2007 Semantic Web Track, September 2008, Pages 203-217, ISSN 1570-8268

[Bhattacharya and Getoor, 2005] Indrajit Bhattacharya and Lise Getoor. Entity resolution in graphs. Technical Report 4758, Computer Science Department, University of Maryland, 2005.

[Mihalcea, 2005] Rada Mihalcea, 2005. Unsupervised large-vocabulary word sense disambiguation with graph-based algorithms for sequence data labeling. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing* (Vancouver, British Columbia, Canada, October 06 - 08, 2005). Human Language Technology Conference. Association for Computational Linguistics, Morristown, NJ, 411-418.

[Lehmann *et al.*, 2007] J. Lehmann, J. Schuppel, and S. Auer. Discovering Unknown Connections - the DBpedia Relationship Finder. In Proc. of CSSW, 2007.

[Finkel *et al.,* 2005] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL 2005), pp. 363-370.*

[Lloyd *et al.*, 2005] Levon Lloyd, Varun Bhagwan, Daniel Gruhl and Andrew Tomkins, 2005. Disambiguation of references to individuals. Technical Report, RJ10364(A0410-011), IBM Research.

[Dill *et al.*, 2003] Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John A. Tomlin, and Jason Y. Zien 2003. SemTag and seeker: bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the 12th international Conference on World Wide Web (Budapest, Hungary, May 20 - 24, 2003). WWW '03. ACM, New York, NY, 178-186.*

[Schütze, 1998] Hinrich Schütze, 1998. Automatic word sense discrimination. *Comput. Linguist. 24, 1 (Mar. 1998), 97-123.*

[Fellegi and Sumter, 1969] Fellegi, I. P., and Sunter, A. B. A theory for record linkage. *Journal of American Statistical Association 66,* 1 (1969), 1183--1210.

[Sandhaus, 2008] Evan Sandhaus*: The New York Times Annotated Corpus,* Linguistic Data Consortium, Philadelphia, 2008.

Uniting *a priori* and *a posteriori* knowledge: A research framework

Michael Witbrock, Elizabeth Coppock, and Robert Kahlert*
Cycorp, Inc.
{witbrock,ecoppock,rck}@cyc.com

Abstract

The ability to perform machine classification is a critical component of an intelligent system. We propose to unite the logical, *a priori* approach to this problem with the empirical, *a posteriori* approach. We describe in particular how the *a priori* knowledge encoded in Cyc can be merged with technology for probabilistic inference using Markov logic networks. We describe two problem domains – the Whodunit Problem and noun phrase understanding – and show that Cyc's commonsense knowledge can be fruitfully combined with probabilistic reasoning.

1 Introduction

Machine classification is a general problem of fundamental importance to the field of artificial intelligence. The ability to harness the vast amount of information freely available on the World Wide Web, for example, depends on technology for solving the *entity resolution* problem: Determining whether two expressions refer to the same entity. Classification is important in military and law enforcement domains as well; consider the *Whodunit Problem:* given features of a criminal act, who is the most likely perpetrator? Classification problems in natural language processing include *word sense disambiguation* and *noun phrase understanding*: in the phrase, *Swiss bank,* what sense of *bank* is involved, and what relation to Switzerland does the referent have? With good machine classification technology, it will be possible to solve many important problems across a wide range of domains.

Our research agenda is to develop systems that are capable of taking into account both empirical statistics and *a priori* knowledge in order to solve classification problems. *Description logic* [Baader *et al.*, 2003] is an example of a purely logical, *a priori* approach to the problem of classification. Description logic provides a medium for encoding facts about the real world, and can be used to classify objects based on their attributes. This approach, however, is fundamentally too brittle; failure to meet any one of the conditions for classifying an object into a particular class is equivalent to meeting none of the conditions.

On the other end of the spectrum are machine-learning techniques. This type of approach succeeds in being more flexible than approaches like description logic, by having weighted constraints that may combine in a gradient fashion. The pure machine-learning approach suffers, however, from being overly reliant on having large quantities of training data. Training data is often lacking: It is costly to produce labelled data, and even labelled data may be sparse for other reasons. For instance, because most words are infrequent (by Zipf's law), training data for many natural language processing tasks is likely to be missing. Moreover, information that is already known should not have to be re-learned; it should be possible to combine what is known already with knowledge gained from empirical statistics.

In recent years, the gap between statistical and logical approaches to classification has begun to narrow. In the field of information extraction, statistical and non-statistical methods have been combined, for example in the TextRunner system [Banko and Etzioni, 2008] and SOFIE [Suchanek *et al.*, 2009]. The fields of *relational data mining* [Dzeroski and Lavrac, 2001] and *statistical relational learning* combine ideas from probability and statistics with tools from logic and databases [Getoor and Taskar, 2007]. A wide variety of techniques within these field have been developed, such as probabilistic relational models, knowledge-based model construction, and stochastic logic programs, and many of these techniques are special cases of *Markov logic* [Domingos and Richardson, 2007], [Domingos *et al.*, 2006]. In Markov logic, weights are attached to arbitrary formulas in first order logic, defining a probability distribution over possible worlds [Richardson and Domingos, 2006].

We propose to integrate Markov logic with the Cyc project, a large-scale effort to represent commonsense knowledge. The Cyc system cannot trivially be converted into a Markov logic network, because the Cyc knowledge base is quite large, and Cyc uses higher order logic. However, it is possible to create Markov logic networks over subsets of the Cyc knowledge base, and to bridge these two resources in a way that usefully combines logical and statistical approaches to artificial intelligence.

2 Background

2.1 Cyc

For over 20 years, the Cyc project has been devoted to the development of a system that is capable of reasoning with commonsense knowledge. At the core, Cyc consists of a powerful inference engine combined with a knowledge base (KB) that contains over 6 million assertions. These assertions are expressed in a language (CycL) based on first-order logic, enhanced by a quoting mechanism and higher-order extensions [Matuszek *et al.*, 2006]. In normal inference, the assertions in the Cyc KB function as "hard constraints" in the sense that if a formula contradicts an existing fact (within a given context), it is considered simply to be false. Thus, for the most part, Cyc represents the logical, symbolic, *a priori* approach to artificial intelligence.[1]

A portion of the information in the Cyc KB is *taxonomic,* expressing (i) the class membership of terms, using the binary predicate `isa`, which relates an *instance* to a *collection* e.g. (`isa Snoopy Dog`), where `Snoopy` is an individual dog and `Dog` stands for the collection of all dogs; (ii) the subsumption relationships among those classes, expressed with `genls`, relating a *subcollection* to a *supercollection* e.g. (`genls Dog Animal`); (iii) disjointness information, expressed with `disjointWith`, which holds of collections that do not share any members. Cyc predicates (including `isa` and `genls`) are associated with *definitional* information, which constrain the types of entities that may appear as arguments to the predicate. Consider some of the argument constraint information for the predicate `biologicalMother` (read "has as the biological mother").

```
(arg1Isa biologicalMother Animal)
(arg2Isa biologicalMother FemaleAnimal)
```

The argument constraint information states that the notion "has as a biological mother" is defined for pairs of instances whose first member is an `Animal`, and whose second member is a `FemaleAnimal`. Definitional information, combined with the taxonomic hierarchy, makes Cyc into a higher-order system.

Another higher-order feature of Cyc is that predicates are also arranged in a generalization hierarchy; `biologicalMother` is a more specific predicate than `relatives`. This relation between the two predicates is expressed with the second-order predicate `genlPreds` as follows:

```
(genlPreds biologicalMother relatives)
```

This means that `biologicalMother` inherits all of the constraints on the predicate `relatives`, including the following rule (CycL variables, noted with question marks, are implicitly universally bound by default):

```
(implies
  (isa ?COL BiologicalSpecies)
  (interArgIsa1-2 relatives ?COL ?COL))
```

The predicate `interArgIsa1-2` specifies a type constraint on one argument, given the type of another. This rule requires, for example, that if X is Y's relative, and X is a bird, then Y is also a bird.

This higher-order information is used extensively by the Cyc inference engine to prune search when answering queries. The reduction in search space makes it feasible to perform inference over a KB of the size of Cyc. This means that the Cyc KB cannot be converted as a whole into Markov logic, but it is possible to use this higher-order information to identify subsets of the KB with which to build Markov logic networks.

2.2 Markov Logic Networks

Markov logic is a language that unifies first order logic with probabilistic graphical models [Richardson and Domingos, 2006]. In Markov logic, logical formulas are associated with weights. Intuitively, the higher the weight is for a given formula, the less likely it is to be contradicted. Formally, weights are interpreted using a Markov logic network, which defines a probability distribution X over assignments of truth values to propositional variables, or *worlds*. Given a set of formulas and their associated weights, the probability of a world x is defined as:

$$P(X=x)=\frac{1}{Z}\exp(\Sigma_{i=1}^{F}w_i n_i(x))$$

where F is the number of formulas, Z is a normalization constant ensuring legal probabilities, w_i is the weight of the ith formula, and $n_i(x)$ is the number of true groundings of the ith formula in x. This means that the more times a world violates a formula, the less likely the world is (when the weight is positive), and the higher the weight, the stronger the effect. When the weight is infinite, violations of the formula are impossible; this is how "hard constraints" are modelled.

Software for weight learning and inference with Markov logic networks is provided through the Alchemy system [Kok *et al.*, 2007]. Alchemy is a flexible software package providing generative and discriminative methods for weight learning and several methods of performing probabilistic inference, including MC-SAT, Gibbs Sampling, and Belief Propagation (ibid). In what follows, we describe a framework for integrating Cyc with Alchemy.

3 Merging Cyc with Markov Logic

3.1 The Whodunit Problem

The Cyc Analyst's Knowledge Base (AKB) is a portion of the Cyc KB that contains over 4500 events of terrorism, with information about each event including the type of attack, the location, and the agent. Given facts about an event, the goal is to predict who was the perpetrator – the *Whodunit Problem.*[2]

[1] Some assertions in Cyc are defeasible; CycL contains five possible truth values: *monotonically false, default false, unknown, default true,* and *monotonically true.* Default assertions can be overridden when two rules conclude P, but one concludes that P is monotonically true and the other concludes that P is default false. Then, all else being equal, Cyc sets the truth value of P to the one suggested by the monotonic rule [Panton *et al.*, 2006]. However, formulas are not associated with probabilities in Cyc.

[2] Several approaches to this problem were presented by [Halstead and Forbus, 2007].

In addition to specific facts about specific events, the AKB contains a rich hierarchy of event types; for example, a CarBombing is a type of Bombing, which is a type of IncurringPhysicalDamage, etc. It also contains a rich hierarchy of agent types. For example, any UrbanGuerillaGroup is also a RevoltOrganization, and by virtue of that, a PoliticalOrganization. These relationships are stated via genls assertions in the KB, and this information can be leveraged to construct Markov logic networks that accurately model the probability distribution over possible event perpetrators.

The event type and agent type hierarchies are both part of a single collection hierarchy in Cyc, but they could in principle be separated into two hierarchies, one for agents, and one for events. It turns out that the latter approach is more efficient for Markov logic networks, because it reduces the size of the model that must be constructed. With two separate hierarchies, it is possible to rule out from consideration the possibility that a something is both a RevoltOrganization and a Bombing within the Markov logic network. It is important to prevent Markov logic networks from considering such impossible states of affairs, because the network contains a node for every grounding of every formula, and this can easily grow quite large. (In contrast, Cyc's inference engine uses the fact that agents and events are disjoint to restrict search to only those groundings that are immediately relevant to its current inference problem.) Thus, in place of Cyc's isa predicate, we introduced two separate MLN predicates, IsaA and IsaE, which relate agents to agent types, and events to event types, respectively.

An MLN specification consists of a set of type declarations and a set of (possibly weighted) formulas. Here is an example type specification:

```
IsaE(event, event_type)
Perp(event, agent!)
```

The first declaration expresses that IsaE is a relation between something of type event and something of type event_type; the second declaration expresses that Perp is a relation between an event and an agent. (Note that the expressions event, event_type, and agent are types, whereas in the formulas below, the arguments of the predicates are implicitly universally quantified variables.) The exclamation point ('!') following agent indicates that there is exactly one agent who perpetrated each event; the relation is *exclusive and exhaustive* in its second argument. (This is a simplifying assumption; an event can have more than one perpetrator in principle, but assuming that this rare case is impossible has dramatic computational advantages.)

Here is an example of an MLN formula:

```
IsaE(e, +et) => Perp(e, +a)
```

The '+' symbol makes this a *per-constant* formula: When a variable in a formula is preceded by '+', a separate weight is learned for each formula obtained by grounding that variable to one of its values. This notational device makes it possible to gather statistics from the corpus during learning about what agents, specifically, tend to perform what types of events. The resulting weights associated with each formula can be used in weighted inference to predict the perpetrator of the event. For example, consider the following two formulas:

```
IsaE(e, MortarAttack) => Perp(e, AlFatah)
IsaE(e, MortarAttack) => Perp(e, LebaneseHizballah)
```

The weights associated with these formulas are -0.0078 and 1.16332, respectively. The contrast captures the fact that LebaneseHizballah is more prone to commit events of type MortarAttack than AlFatah is.

Using only event type and location information, it is possible to predict the perpetrator with approximately 70% accuracy. But this number can be improved by adding in "hard constraints". For example, the date of an event can be used as a soft indication of who perpetrated the event, but combined with knowledge about when a perpetrator existed, can be used to rule out a perpetrator absolutely: Events are not perpetrated by organizations/individuals that do not exist yet. For example, it is asserted in the Cyc KB that LebaneseHizballah was founded in 1982, and for a certain terrorist act, it is stated that it occured on May 25th, 1977. Reasoning with a rule stating that any agent who comes into being after an event takes place cannot be the agent of that event, it is possible to infer that LebaneseHizballah cannot have been the perpetrator of the 1977 event.

This section has demonstrated two ways in which Cyc's commonsense knowledge can be combined with probabilistic reasoning. First, Cyc provides a hierarchically structured ontology that underlies the content of the "soft constraints". Second, it provides rules that can be treated as "hard constraints".

3.2 Noun phrase interpretation

We now turn to another domain entirely, in order to demonstrate a more complex type of problem. One of the advantages presented by Markov logic networks over simpler machine learning techniques is that they jointly predict multiple variables. This is of crucial importance for the problem of *noun phrase interpretation.* For example, in a noun phrase such as *fire bomb,* (i) What does the head noun (e.g. *bomb*) mean in this context? (ii) What does the modifier (e.g. *fire*) mean in this context? (iii) What is the relation between the noun phrase and the modifier? In other words, what does the entire phrase mean? The noun phrase *fire bomb* describes a bomb that creates fire, but consider the many kinds of relations that can hold between modifiers and *bomb* (all taken from descriptions of terrorism events in the Analyst's Knowledge Base):

- *main ingredient:* petrol bomb, tear gas bomb, shrapnel bomb, nail-filled bomb
- *mechanical component:* pipe bomb
- *result:* fire bomb, smoke bomb, sound bomb
- *manner of camouflage:* car bomb, suitcase bomb, videocassette bomb, book bomb, parcel bomb
- *triggering mechanism:* time bomb, remote-control bomb

The problem of understanding noun phrases like this requires joint inference because the relation between the noun phrase and the modifier depends on the interpretation of the head noun and the modifier, but the interpretation of the head noun and the modifier can also be influenced by the relation; these problems are mutually interrelated.

Creating hand-labelled data is expensive, and there is already information about noun phrases within the Cyc knowledge base that can be used as training data. In particular, the Cyc lexicon contains thousands of lexical entries for strings such as *western film.* Under certain assumptions, this information indirectly reveals how the modifier and the noun are related. The assumptions are as follows:

(i) Western films are films. More precisely, if the meaning of the whole phrase bears the genls relation to some denotation of the head word, then the latter is what the head word denotes in this context. Possible denotations of *film* include 'photographic film', 'the act of filming', 'a sheet or coating', and 'movie'; since what *western film* denotes is a type of movie, it is clear that *film* means 'movie' in this context.

(ii) Western films have something to do with westernness. More precisely, if there is an assertion mentioning the meaning of the whole phrase and a concept that can be denoted by the modifier, then the latter concept is what the modifier denotes in this context. In our example, the modifier *western* can denote either the cardinal direction west, a western story, or a western conceptual work. The meaning of *western movie* is related to western stories and western conceptual works via the genls relation. Thus we can assume that in the phrase *western film,* the modifier string *western* has one of those two meanings.

In general, the semantic relation between the modifier and the phrase can be any binary relation. For another example, in the phrase *blackberry bush,* the relation is a fruitOfType relation, which holds between the meaning of *blackberry* – (FruitFn BlackberryBush) – and the meaning of the whole phrase: BlackBerryBush.

This corpus can be used as data to train a Markov logic network that simultaneously disambiguates the head noun and the modifier and identifies the relation that holds between the meaning of the modifier and the meaning of the phrase. Here is a sample set of type declarations for predicates capturing the relevant information about noun phrases:

$$
\begin{array}{l}
\mathtt{reln(np, reln)}\\
\mathtt{modDenotes(np, sense)}\\
\mathtt{headDenotes(np, sense)}\\
\mathtt{genls(sense, concept)}\\
\mathtt{denotes(word, sense)}\\
\mathtt{modWord(np, word)}\\
\mathtt{headWord(np, word)}\\
\mathtt{modGenls(np, concept)}\\
\mathtt{headGenls(np, concept)}\\
\mathtt{wellFormedArg1(sense, reln)}\\
\mathtt{wellFormedArg2(sense, reln)}
\end{array}
$$

These capture what the semantic relation involved is (genls, fruitOfType, etc.), the meaning of the head and the modifier, the taxonomic hierarchy and denotational information about each word, how the meanings of modifier and the head noun fit into the taxonomic hierarchy, and argument type constraints on semantic relations.

Here are sample hard constraints that could be expressed using these predicates:

$$
\begin{array}{l}
\mathtt{modDenotes(p, s)} \wedge \mathtt{modWord(p, w)} \Rightarrow \mathtt{denotes(w, s)}.\\
\mathtt{headDenotes(p, s)} \wedge \mathtt{headWord(p, w)} \Rightarrow \mathtt{denotes(w, s)}.\\
\mathtt{modGenls(p, c)} \wedge \mathtt{modDenotes(p, s)} \Rightarrow \mathtt{genls(s, c)}.\\
\mathtt{reln(p, r)} \wedge \mathtt{modDenotes(p, s)} \Rightarrow \mathtt{wellFormedArg1(s, r)}.
\end{array}
$$

These express the following constraints: words only denote senses that they have; if the modifier denotes a given type of concept and the modifier has a given sense, then that sense is that type of concept; if the phrase involves a certain semantic relation, and the modifier has a given sense, then that sense fits the argument constraints for the semantic relation.

Here are sample per-constant formulas that yield soft constraints:

$$
\begin{array}{l}
\mathtt{modGenls(p, +c)} \Rightarrow \mathtt{headGenls(p, +c')}\\
\mathtt{headGenls(p, +c)} \Rightarrow \mathtt{modGenls(p, +c')}\\
\mathtt{modGenls(p, +c)} \Rightarrow \mathtt{reln(p, +r)}\\
\mathtt{headGenls(p, +c)} \Rightarrow \mathtt{reln(p, +r)}
\end{array}
$$

These will capture regularities such as "the head tends to denote a kind of person when the modifier denotes an ethnicity"; "the relation tends to be a fruitOfType relation when the modifier denotes a fruit," etc.

The resulting model combines the Cyc ontology, logic, and probability to solve the problem of noun phrase interpretation. Taxonomic information (isa and genls) underlies the content of the soft constraints, and definitional information (argument type constraints) comes into play in defining hard constraints. These pieces of information are combined with statistics in a single unified model of this restricted domain.

4 Conclusion

We believe that a successful strategy for merging background knowledge and empirical statistics lies in unifying Cyc's strengths with those of Markov logic networks. Our initial case studies have shown that the knowledge in Cyc comes into play both as a way to categorize the data so that useful statistics can be computed, and as a way of enforcing regularities in the model that must hold. We have addressed efficiency issues by finding ways of minimizing the size of the Markov logic network that is constructed, for example, by using two separate predicates where Cyc uses only one. We see great potential in creating a bridge these two resources, for the sake of solving these problems, along with countless other problems of classification.

References

[Baader *et al.*, 2003] Franz Baader, Diego Cavanese, and Deborah L. McGuinness, editors. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press, 2003.

[Banko and Etzioni, 2008] Michele Banko and Oren Etzioni. The tradeoffs between open and traditional relation extraction. In *Proceedings of ACL-08: HLT*, pages 28–36, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[Domingos and Richardson, 2007] Pedro Domingos and Matt Richardson. Markov logic: A unifying framework for statistical relational learning. In Getoor and Taskar [2007], pages 339–371.

[Domingos *et al.*, 2006] Pedro Domingos, Stanley Kok, Hoifung Poon, Matt Richardson, and Parag Singla. Unifying logical and statistical AI. In *Proceedings of the Twenty-First National Conference On Artificial Intelligence*, pages 2–7. AAAI Press, Boston, MA, 2006.

[Dzeroski and Lavrac, 2001] Saso Dzeroski and Nada Lavrac, editors. *Relational Data Mining*. Springer, Berlin, 2001.

[Getoor and Taskar, 2007] Lise Getoor and Ben Taskar, editors. *Introduction to Statistical Relational Learning*. MIT Press, Cambridge, MA, 2007.

[Halstead and Forbus, 2007] Daniel T. Halstead and Kenneth D. Forbus. Some effects of a reduced relational vocabulary on the whodunit problem. In *Proceedings of IJCAI-2007*, Hyderabad, India, 2007.

[Kok *et al.*, 2007] S. Kok, M. Sumner, M. Richardson, P. Singla, H. Poon, D. Lowd, and P. Domingos. The Alchemy system for statistical relational AI. Technical report, Department of Computer Science and Engineering, University of Washington, 2007. http://alchemy.cs.washington.edu.

[Matuszek *et al.*, 2006] Cynthia Matuszek, John Cabral, Michael Witbrock, and John DeOliveira. An introduction to the syntax and content of Cyc. In *Proceedings of the 2006 AAAI Spring Symposium on Formalizing and Compiling Background Knowledge and Its Applications to Knowledge Representation and Question Answering*, 2006.

[Panton *et al.*, 2006] Kathy Panton, Cynthia Matuszek, Douglas Lenat, Dave Schneider, Michael Witbrock, Nick Siegel, and Blake Shepard. Common sense reasoning – from Cyc to intelligent assistant. In Y. Cai and J. Abascal, editors, *Ambient Intelligence in Everyday Life*, pages 1–31. Springer-Verlag, Berlin, 2006.

[Richardson and Domingos, 2006] Matt Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62:107–136, 2006.

[Suchanek *et al.*, 2009] Fabian M. Suchanek, Mauro Sozio, and Gerhard Weikum. SOFIE: a self-organizing framework for information extraction. In *Eighteenth International World Wide Web Conference (WWW2009)*, 2009.

Is the Web a Web of Documents or Things?*

Xiaoshu Wang
Renassaince Computing Institute
Chapel Hill, North Carolina, USA
xiao@renci.org

Arlindo L. Oliveira
INESC-ID/IST
Lisbon, Portugal
aml@inesc-id.pt

Abstract

How we design and structure our information in the Web is essentially influenced by our philosophical viewpoints on what the Web is. In this paper, we compared two fundamentally different positions: one takes the Web to be *a web of documents* and the other *a web of things*. By using Fred Dretske's semantic information theory, we discussed why we should favor the second model over the former through our formulation of two information triads. The first one is the Knowledge-Information-Data (KID) triad that allows us to clearly define these concepts within a communicative praxis. The second one is the Symbol-Information-Referent (SIR) triad that allows us to clearly define and connect all kinds of information systems, regardless they are the man-made or naturally occurring ones.

1 Introduction

The architecture of the World Wide Web (AWWW) is defined in terms of three essential concepts: URI, resource, and representation [AWWW, 2004]. A URI is a symbolic thing – a kind of thing that denotes or references another thing[1], which is called *resource* in the Web. Obviously, things have meaning, which is their significance [Quine, 1961]. To obtain meaning, however, requires a communication system because, in the absence of such a system, everything becomes unobservable and irrelevant to others. In the physical world, the system is space-time, where meaning is delivered in force and energy; in biological world, it is sense, where meaning is delivered in light, sound, and pressure etc; in the Web, the system is the network transportation, where meaning is delivered in *representations*.

The above description of *URI*, *resource* and *representation* should be consistent with how they are described in the current AWWW. But to make subsequent discussions clear, we will narrow the definition of resource. Resource is here used to refer those things that are neither symbolic (so that Resource $\neq$ URI) nor representative (hence, Resource $\neq$ Representation). In addition, a resource must have an established identity, i.e., an explicit and canonical URI, in the Web. The word "thing" will be used to replace the general notion of resource.

1.1 Information Resource and Document

In the current AWWW document, there is a fourth concept – *information resource*, also informally known as *document* [Berners-Lee, 2002], that is defined to be those things "that all of their essential characteristics can be conveyed in a message." However, not only the clarity of the above definition but also the essentiality of such a concept to the Web has been controversial. "What information resource is" is at the heart of many debates, most notably the httpRange-14 [W3C TAG Issue, 2002]. Previously, we have argued how the above ambiguous definition cannot possibly be objectively followed in practice [Wang, 2007]. In here, we will argue its philosophical shortcomings. To this end, Fred Dretske's writing on what information is could be instrumental.

> There is one way of thinking about information. It rests on a confusion, the confusion of information with meaning. Once this distinction is clearly understood, one is free to think about information (though not meaning) as an objective commodity, something whose generation, transmission, and reception do not require or in any way presuppose interpretive processes. One is therefore given a framework for understanding how meaning can evolve, how genuine cognitive systems – those with the resources for interpretive signals, holding beliefs, and acquiring knowledge – can develop out of the lower-order, purely physical, information-processing mechanisms.([Dretske, 1981], p. vii)

According to the above view, just the wording of "information resource" could already start out as a confusion. *Re-*

*This work is done while the first author is supported by a research contract financed by the Ciência 2007 program of Fundação para a Ciência e Tecnologia.

[1] A few notes on the choice of our words. First, URI is used, as oppoed to Internationalized Resource Identifiers (IRI), for the sake of being consistent with the current AWWW document. Second, in the current AWWW document, the word "identify" is used to describe the relation between URI and resource. But "identify" has at least two interpretations: (1) to cause to be or become identical, and (2) to establish the identity. We believe the intention of AWWW is the second one because a URI cannot possibly become identical to the resource that uses it to establish the resource's identity in the Web.

source is an inherently static concept and by nature meaningful because a completely meaningless thing lives in absolute solitude and, therefore, virtually non-exist. Information, on the other hand, is an inherently dynamic concept because it is often associated with an event, from which knowledge may transpire. To say "information", therefore, presupposes two things – a source and a recipient, but to say "resource" presupposes only one. Hence, the simple apposition of the two words already foretells an identity crisis.

Nevertheless, it is not too wrong to suggest that there are two kinds of things in the Web – one is informative and the other not. As a matter of fact, the Web used to have such distinction. The informative things were denoted by URLs and the others by URNs [Berners-Lee *et al.*, 1998]. However, as soon as the Web started its march to the Semantic Web, it was realized that the distinction between URL and URN is only arbitrary and inconsequential. Once again, it is due to the confusion of meaning with information. But this time, it is a variant of it – the confusion of reference with access [Hayes and Halpin, 2008]. Reference is the subject of semantics whereas access is the means of obtaining information. Unless URNs are used to denote things that cannot possibly be connected to the Web, URN-things can always be accessible in the web and, therefore, informative. In fact, these absolute URN-things do not even exist, at least within the confine of human knowledge.

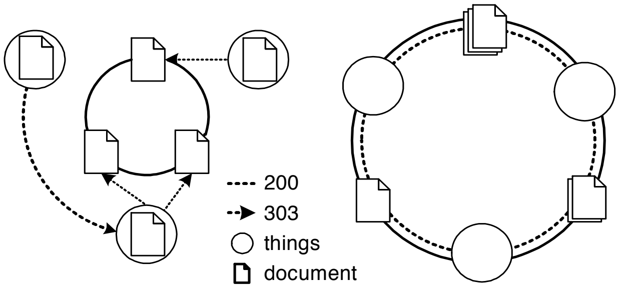
If the AWWW's definition of "information resource" simply stops at "can be conveyed in a message", there will be no debate and no controversy. Anything else would be nothing more than a play of words and a parade of self-conceit. However, in telling us what an *information resource* is with the ambiguous wordings, such as "all", "essential" and "can", and in telling us how a *non-information resource* should behave [W3C TAG Issue, 2002], and in telling us that non-cooperative behaviors may be potentially punished in the future[2], the AWWW's definition of *information resource* makes the Web impossible to work with. The reason is clear: unless there is a complete and indisputable set of knowledge on everything in the universe, what is "all", "essential", and "can" is always subject to debate and change.

2 Two Models of the Web

On a deeper level, the debate about the essentiality of information resource (we will use the word "document" from now on) reflects two contrasting views on what the Web is. In the first view, the Web is considered to be *a web of documents talking about things*. In the second, it is *a web of things talking with documents* (Fig. 1). Let us see how the two views will affect our thinking about the Web and our life in it.

2.1 A Web of Documents Talking about Things

The rearing of this view may come from history. As the Web was initially conceived to be a web of documents – those written in HTML and linked through HTTP, it seems right to suggest: as a natural progression, a web of things should be built on top of a web of documents. However, our general use of the word "document" has been ambiguous because, subconsciously, we often think that the thing (document) that we have retrieved from a URI is the thing denoted by that URI. But this is an obvious case of psychological identification but not a physical one because a thing cannot possibly be the same as another thing that represents it.



(a) Web of documents talking about things

(b) Web of things talking with documents

Figure 1: Two architectural views of the Web.

Nevertheless, putting aside the issue of network's reliability, practicing psychological identification in the Web may not matter much if there is a one-to-one relationship[3] between resource and representation. In fact, this one-to-one relationship is the prerequisite for the web-of-document model to stand. Of course, we do not know if this – the first architecture view of the Web – is what is behind the definition of *information resource*(document) and the resolution of httpRange-14. But the latter two certainly have helped reinforcing the former. Take the definition of *document* as example. If *all* essential characteristics of a *document* can be conveyed in *a* message, we can of course HTTP-GET a *document* – *all* in *one* representation. On the other hand, if *non-documents* cannot have it *all*, they must not have any (representation) because otherwise documents would have no privilege over non-documents in the Web. This not only makes any definition of document (as resource) a moot point but also makes the web-of-document virtually non-exist. However, by mandating non-document things to 303-redirect (per the bylaw of httpRange-14), a web of 200-documents (but not exactly a web of all documents) is created, upon which a web of other things could be built, though rather inconveniently.

The mechanism of content negotiation provided by HTTP [Fielding *et al.*, 1999], however, strikes a serious blow to the above model because what if a resource/document *can* but may not convey the *all* in one of the possibly many representations? This will once again blur the distinction between document and non-document things and subsequently jeopardize the web-of-document model. Of course, to make the model work, the *one*-representation can be thought as a semantic- rather than a syntactic-*one* so that all representations of a document become mathematically transformable. To make this work, however, requires all languages – both human and machine ones – to have the same expressivity on

[2] TAG mailing archive: http://lists.w3.org/Archives/Public/www-tag/2007Oct/0050.html

[3] A resource can have multiple representations; this is not at debate. What is at debate is whether all these representations should have some kind of equivalence.

every possible subject. This is neither true in reality nor likely to be true in the future. For instance, how can we, if ever, to capture the essence of a picture in a language? The only option to make the one-to-one model work will be through control. That is, to make a policy on what kinds of contents that a resource can and cannot be negotiated. If this were indeed to happen, we must seriously ask ourselves: is this what we want, or what we want the Web to be? First, any constraints on content negotiation will make the Web cumbersome because a resource must always respond in an all-out fashion as opposed to do it discriminatingly and selectively according to a client's special request. Second, it will hinder innovation because there are quite many real-world use cases that depend on a free-extending content negotiation[4].

As always, of course, we should sacrifice ourselves for the benefit of community. But the question – and a very important one – is: what could the Web have possibly gained had we made all the above sacrifices? From what we know and can possibly imagine, nothing – except a vaguely defined vocabulary and an arbitrary model.

2.2 A Web of Things Talking with Documents

What makes the first model fail, in fact, is neither its grip on document, which is an essence of the Web, nor its use of the word *information*, which is a correct instinct. The model fails because it misaligns *information* with *resource*. Perhaps, we have all forgotten that the Web is at the first place a communication system. As it is with any communication system, it is the nature of signal, but not that of its producers or receivers, that defines the system's characteristics. In this sense, the correct alignment of the word "information" should be with *representation* but not with *resource*. To follow Dretske's advice [Dretske, 1981] that, in order to provide a semantic theory of information, the word "information" should be used to refer to the semantic content of a signal, we think that the word "document" should be more meaningfully defined to be the semantic content of a *representation*. By this definition, *document* is neither *representation*, which is the signal that carries it, nor *resource*, which is the thing that the document describes.

The above notion of *document*, in fact, can help us further clarify the concept of *representation*, which is yet clearly defined in the current AWWW document. For instance, it is unclear which of the following three things should be called a "representation" in an HTTP-based web interaction? (a) The byte-stream (b) The HTTP message, i.e., the thing that is structured as request line, headers, and entity (c) The parsed HTTP entity, which can be an image, an XML/HTML tree, an RDF model, an PDF document, etc., depending on the content type of a particular message. With the above conceptualization, however, we can make a distinction between (b) and (c) by referring to them as representation and document, respectively. And, just as document is the semantic content of representation, representation is the semantic content of the byte-stream, which is, in turn, the semantic content of some other *signals*, such as electrons or photons, etc. This is, we believe, how the Web is fundamentally constructured – a point that we will discuss in the next section.

Conceptually, the *document* described here may not differ much from what the concept of "information resource" was initially conceived in the first model [Berners-Lee, 2002]. But the difference in their conceptual alignment, i.e., with resource vs. with representation, makes a significant difference in terms of how a *document* should be denoted. With the web-of-document model, document is just a kind of resource. Hence, as everything else, it will be denoted by just one URI. As a matter of fact, this is the working model that most, if not all, of us have been subconsciously using ever since the inception of the Web. Take the resource denoted by "http://dfdf.inesc-id.pt/voc/df" as an example. After browsing its content, many people will probably claim that the URI denotes an HTML document. But it is easy to find that the claim is false: dereferencing the URI will return an RDF/XML document if the HTTP request is set to accept "application/rdf+xml". In fact, the actual serving of a different content-type document is not even needed to challenge the above claim. Just the mere possibility that a different type of content might be served under the same URI will pose serious problem to the claim. In this light, the conflict between content negotiation and the web-of-document model is very fundamental[5] and the only way to solve it would be finding a general notion of document, namely, information resource. But the chase for a working definition of "information resource" has been shown to be very difficult, if not impossible. More importantly, even if were we able to find a workable definition, the conceptualization, we believe, would be too general to be even close to what we really want in practice – to unambiguously denote a structure with a symbol.

With the web-of-thing model, however, we will not be put in such a predicament. Document is pragmatically defined to be what is parsed from a representation. And what the parsed document is and how the parsing should be conducted will be defined by another thing – content-type[6]. In this model, *two* URIs – one for resource and the other for content-type – would be needed to denote a document. But in exchange for this minor inconvenience, we get a web model that is not only easier to work with (no more worry on information resource) but also lighter and faster (no more unnecessary 303). More importantly, we will get a model that makes the Web fit naturally with the rest of world.

Let us use a simple example to illustrate the last point. Imagine the apple that we have placed in front of us (`u:s`) and named it `an:apple` (Let `u:s` and `an:apple` be the QNames[7] of two hypothetical URIs). There are several information systems that connect `an:apple` with `u:s`. There is the light that gives `u:s` `an:apple`'s color and shape, and vice versa; there is the air that gives `u:s` `an:apple`'s scent, and vice versa; and there is the Web that gives `u:s` `an:apple`'s birth place, drug (pesticide) history, etc. and vice versa!

[4] The limited space prevents us from discussing it with a use case.

[5] Here "content negotiation" means more specifically the negotiation over the content-type but not language.

[6] We think content-type needs to be denoted in URI as well.

[7] Qualified name: http://www.w3.org/TR/REC-xml-names/#dt-qualname

Some may immediately cry afoul over our use of "vice versa": `an:apple` can *not* see, smell, let alone access the internet. But, let us refute them with "子非鱼, 焉知鱼之乐?" Our intention here, though, is neither to show off our Chinese (being the first author's native language) nor to direct this article to more philosophical issues than it is necessary. Rather, we are here to illustrate Dretske's point: information is objective but meaning is not. What `an:apple` is as a reality – that is, what makes `an:apple` `an:apple` but others not – is an ever eluding thing to know. But we can think – as a personal or public-accepted belief – that `an:apple` can not see, smell, and access the internet. But we do not know that for a fact because we do not know apple's language. This is exactly the same situation that we – you, the readers, and `u:s`, the authors – are in with regard to a third thing "子非鱼, 焉知鱼之乐?" As an information from `u:s`, the sentence carries some meaning in `u:s` but incurs (perhaps) nothing in you. Yet, what the sentence is as itself is unknown because we do not know what the sentence, the information, means to the sentence, the thing. Hence, information, as an objective entity, is always out there in spite of how it is thought. Conversely, how one thing thinks of another is independent of how the latter's information is acquired. For instance, even without our translation[8] of "子非鱼, 焉知鱼之乐?" you may still find it in some other way. Our information paths to the sentence are definitely different, but our conceptualization about it – that is, its meaning in us – may nevertheless be ended up the same.

With the above illumination on the subject, let us now ponder the question: is there any essential difference between the light, the air, and the Web as information systems? And, is there any essential difference between wavelength, scent molecule, and document as information? We think not. The Web, in the view of the second architectural model, is simply part of the natural world – as natural as it gets.

Any information system is in fact built from a web-of-thing model; what makes them different is the information that flows in the system. We can, for instance, build *a web of things talking with apples* quite easily. All that is required is our knowledge about one – but not *all* and not necessarily one *essential* – aspect of apple, such as our simple ability to tell sweet apples from sour ones. In fact, the Web is built upon another web of similar attribute. Only this time, it is not sweet-sour apples but on-and-off bits. The bit-web is, in turn, built upon an electron-web and a photon-web, and so on. It is through the bridging of all these information-webs that things become more accessible and meanings, and more meanings, evolve.

3 The Web as Information System

Before going further into our conceptual exploration, we think that it is worth elaborating on why we have favored Dretske's take on information among the many given definitions (see review in [Floridi, 2004] and [Mingers, 1996]). Evidently, we have not chosen Dretske's theory for convenient reason. Nor do we in fact choose it for being the only "correct" theory because doing so would have missed the point of a philosophical investigation. First, the word "information" is variously used; it refers to different conceptualizations in different theories. For instance, Shannon's account of information [Shannon, 1948] differs quite significantly from what is accounted by Dretske's, where the former would be called the "binary signal" that carries the latter. Such a diverse use of the same word, in fact, makes these theories incomparable in terms of "right" or "wrong" because they are essentially theories of quite different things.

Second, judging a philosophical theory by its truth value is at the first place impossible because it brings a question – what is truth? – that will bring the philosophy itself into question. What makes a philosophical inquiry useful, therefore, is for it to be therapeutic [Wittgenstein, 1953]. By shining a different light on the same ailing problem, a good philosophical theory would give us guidance in terms of how to formulate a problem in such that it becomes "rich in consequence, clearly defined, easy to understand, and difficult to solve, but still accessible" ([Hilbert, 1900], paraphrased by Luciano Floridi [Floridi, 2004]). This last point is what ultimately drives our favoritism toward Dretske's theory because, not only does his account give us a naturally fitting architectural model for the Web, but it also – as we will show in this section – helps us to reach a general definition for information system, which in turn helps us to define the Web.

3.1 Knowledge-Information-Data (KID)

Our account of information, though inspired from Dretske's theory, also departs from his in a few key aspects. We take Dretske's account of information as an objective entity, but we do not make a distinction between signal and its semantic content in terms of being information. In our account, anything that has a physical structure is information. Naturally, a signal should have a structure, so it is by our definition information. Similarly, a signal's semantic content can also be information – as long as it is manifested as a structure. What a signal and its semantic content differ is thus the systems where they serve as information. But they do not differ in terms of being information.

By contrast, meaning is a subjective commodity in the sense that it is the being (or existence[9]) of one thing in another. Obviously, how something *is* in one thing is very likely to be different from how it *is* in another. And to acquire what something means to one thing (source) and turns it into what it means to another (recipient) is the ultimate purpose of communication. Nevertheless, meaning is what is to be, but not what is actually, shared because otherwise it would not be called *subjective*. What is actually shared and objective is information. Now, let us define the word "data" to be the meaning in source and "knowledge" meaning in the recipient, we get the familiar knowledge-information-data (KID) triad. By this definition, to communicate is to turn data into knowledge – through the flow of information.

With this conceptualization, our common use of "data"

[8] "You are not fish, how can you know the happiness of fish?" It is a famous quibble recorded in the book of Zhuangzi (see more at *http : //en.wikipedia.org/wiki/Zhuangzi*).

[9] Existence is defined by Quine to be the value of a bounded variable [Quine, 1961].

should be more helpfully understood as its content but not its form. The former is its meaning while the latter its information. Once a data is fed into a processing unit, such as a computer program, and its form being analyzed, it becomes information and may output knowledge, which can in turn serve as the data for the next information processing unit. In other words, it is the data – as information – that is being analyzed but it is the data – as meaning – that drives an algorithm.

But, some may wonder: is meaning a structure? If it is, it is *by our definition* information; but if it is not, what else can it be because nothing can be built out of thin air? This conceptual dilemma seems pointing out a fault in our definitions because they lead to a question that cannot be answered. This is true; but it is a truth that we should expect. "Is meaning a structure?" is a fundamentally different question from, say "Is XML a structure?" The former asks the meaning of meaning while the latter the meaning of XML. We know that an XML means a tree (structure) to us. But, as Gödel's incompleteness theorems tells us, we cannot possibly know what XML means to XML and meaning to meaning. Nevertheless, this incompleteness does not deprive any pragmatic value from the definitions because what matters to our humans is the meanings in us (humans).

Within a communicative praxis, information is what will transpire meaning. Conversely, meaning is what will transform information. From this latter angle, we can find a different triad – signal-meaning-information (SMI)[10] – that can help us to understand things in behavioral terms. For instance, we can pragmatically define an HTTP endpoint as the entity that can turn an HTTP-message (a representation) into a document. In engineering, SMI is in fact the model that we use to build an entity. The KID triad, on the other hand, is the model that we apply to use the entity. It is through the successive interlocking of various SMI and KID triads that information transforms and meanings evolve.

3.2 Symbol-Information-Referent (SIR)

Fundamentally, meaning is the relation of symbols[11]. Among all kinds of relations, *equivalence* is the most essential because asserting synonyms from different information systems is how meaning is ultimately created [Quine, 1961]. For instance, when we named the apple in section 2.2 with the symbol `an:apple`, we have, in fact, created a meaning – `an:apple` is `the-apple`. There are three symbols in this assertion; they are from three different symbol spaces used by three different information-systems. "an:apple" is a symbol in the document-web; "is" is a symbol in the English-system; "the-apple" is a symbol (e.g., as a geodesics) in space-time. Obviously, all three symbols must be subsequently projected to the symbols in our brain in order for us to comprehend its meaning. But without the *is*, there will be no meaning. Both `an:apple` and `the-apple` would be just symbols. But with it, `an:apple` becomes a reference with `the-apple` being the referent. To assert the *is*, however, requires information. For us, the authors, the information could be a simple body gesture; for you, the readers, the information could be this document.

With the above understanding about symbol and references, we can formulate a more fundamental information triad – Symbol-Information-Referent (SIR) – to define any given information system. The referential realm of the symbol space necessarily defines the system's boundary and expressivity; and the kind of information defines the system's characteristics.

Anything is by definition a SIR system, with its form being both symbol and information and its content the referent. But a closed self-reference SIR system is useless to others because its meanings cannot be passed across. The system "3", for instance, will be meaningless to us unless its content is bound to a symbol, e.g., as a number, in our brain. Hence, in order for meanings to evolve, symbols of various SIR systems must be either bound (by asserting the equivalence) or shared. Only by symbol-binding and sharing can two SIR systems be combined into a larger SIR system, from which more information can be acquired and more meanings can evolve. In our earlier example, for instance, the light, the air and the Web forms a larger information system with our binding of `an:apple` to `the-apple`, which tells us a more complete story about the-apple. Of course, to verify the binding requires information, such as a wavelength, a scent molecule, a body gestures or a document. But whether one accepts or rejects the binding of symbols depends on whether the information incurred meaning is consistent with one's personal knowledge. To put it plainly, without information, nothing gives. But with it, nothing is a given.

3.3 The SIR Triads for the Web

Naturally, as a man-made SIR system, the Web must define its own information and implement transportation mechanism to deliver it. But being man-made does not and should not suggest its model be any different from the naturally occurring information system. In Table 1, we have defined a few web systems in terms of the SIR triad.

There are a few things in Table 1 that need further explanation. First, the choice of our word "physical" and "Semantic" is not important. What is important is the differences between the two systems. Obviously, being different kind of information system, they must differ in the form of their information. But in addition to that, we think that they should also differ in their symbol spaces. Symbols in the Semantic Web should be ideally URN but not URL in the sense that it is *technologically* independent of any transportation protocol[12]. The purpose here is not to create the so-called "persistent identifier", which is an ill-defined concept. Rather, it is to separate the semantic web cleanly from the physical web so that a semantic application can, in principle, be executed on any document-delivery system that will take the URN's symbol space, be it the physical Web or postal system.

Second, our use of the word "semantic web" differs from how the words are now commonly perceived, which mostly

[10] Here the word "information" is obviously used in a different sense from the KID triad.

[11] Symbol must have a structure, hence by definition information.

[12] This seems to be a retraction from our earlier criticism on URN in section 1.1. But the key difference is: we think humans, as oppose to machines, is the responsible party to bind a URN to a URL.

System	Symbol	Information	Referent
Semantic Web	URN	Document	Resource
Physical Web	URL	Representation	URL Endpoint
DNS	Domain	A Record	IP Address
Internet	IP Address	Packet	Machine

Table 1: The SIR Triads for the Web.

means the RDF/OWL web. In our conceptualization, any document-web, be it in HTML, RDF, or GIF etc., is a part of the Semantic Web. Of course, each of these sub-semantic webs should differ in their exact form of information; but they should all share the same symbol space.

Third, we stopped our writing of SIRs at the machine level. Nevertheless, based on the principle introduced above, it is not difficult to write it out all the way to "the lower-order, purely physical, information-processing mechanisms" as Dretske have envisioned. This last point, to echo our earlier quoting of Hilbert, is the consequence (a very rich one we believe) of adopting Dretske's framework. This is also the reason why we have accepted his take on information (as an objective commodity) because the Web is now "clearly defined, easy to understand, and difficult to solve, but still accessible."

4 Conclusion

As Guy Fitzgerald pointed out in [Fitzgerald, 1996], one of the most important contributions that philosophy can make to the world of information system is "to highlight the various assumptions that underlie our action." In this article, we identified one such faulty assumption. That is, what we get from a URI *is* what the URI denotes. We discussed how this implicit assumption, or the desire to make it one, may have led to the thinking that the Web is *a web of documents*. Furthermore, we showed how discriminating information from meaning can help us derive a more meaningful – the Web as a web of things – architectural model that fits seamlessly with the naturally occurring information systems. In fact, the web-of-thing model is not only more philosophically amenable but also more pragmatically profitable as it enables us to answer – with consistency and clarity – many hotly debated issues, such as the necessity of URN, the definition of metadata, and the treatment of URI fragment identifiers, etc. Unfortunately, the limited space prevents us from going deeper into that direction.

The remaining space is saved to draw reader's attention to a briefly mentioned subject – HTTP's content negotiation. Content negotiation, if not correctly understood, can be thought of only minor importance or even as an annoyance. But it is, in fact, a much larger topic than it is currently specified and appreciated. From our formulation of the SIR triad, we can see that content negotiation enables us to establish many sub-systems within the Web – with each of them tailored to a specific kind of agents but all of them anchored to the same URI. This is a mechanism not yet seen in any existing network transportation protocols and can provide us with many new and exciting possibilities to solve many seemingly difficult problems. Of course, to understand, and subsequently to take better advantage of, content negotiation requires us to open up our own mind – to rethink the meaning of those seemingly ever-familiar concepts, such as data, knowledge, document, and information etc. We hope, through the presented discussion, this article has helped, at least has helped provoke, our thinking along this direction.

References

[AWWW, 2004] AWWW. Architecture of the world wide web, volume one ⟨http://www.w3.org/tr/webarch⟩, 2004.

[Berners-Lee *et al.*, 1998] T. Berners-Lee, R. Fielding, and L. Masinter. IETF RFC 2396 (Obsolete): Uniform Resource Identifiers (URI): Generic Syntax, 1998.

[Berners-Lee, 2002] Tim Berners-Lee. What do HTTP URIs Identify? ⟨http://www.w3.org/DesignIssues/HTTP-URI⟩, 2002.

[Dretske, 1981] Fred I Dretske. *Knowledge and the Flow of Information*. MIT Press, 1981.

[Fielding *et al.*, 1999] R. Fielding, J. Gettys, J. Mogual, , H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. IETF RFC 2616: Hypertext Transfer Protocol – HTTP/1.1, 1999.

[Fitzgerald, 1996] Guy Fitzgerald. *Philosophical Aspects of Information Systems*, chapter Forward, pages ix–x. Taylar & Francis Group, 1996.

[Floridi, 2004] Luciando Floridi. Open problems in the philosophy of information. *Metaphilosophy*, 35(4):554–582, 2004.

[Hayes and Halpin, 2008] P.J Hayes and H Halpin. In defense of ambiguity. *International Journal on Semantic Web and Information Systems*, 4(2):1–18, 2008.

[Hilbert, 1900] David Hilbert. Mathematische Probleme. *Nachrichten von der Königl. Gesellschaft der Wiss. zu Göttingen*, pages 253–297, 1900.

[Mingers, 1996] John C. Mingers. An evaluation of theories of information with regard to the semantic and pragmatic aspects of information systems. *Systems Practice*, 9(3):187–209, 1996.

[Quine, 1961] Willard Van Orman Quine. *From a Logical Point of View*, chapter On What There is, pages 1–19. Happer & Row, New York, 1961.

[Shannon, 1948] C. E. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27:379–423, 623–656, 1948.

[W3C TAG Issue, 2002] W3C TAG Issue. What is the range of the HTTP dereference function? ⟨http://www.w3.org/2001/tag/issues.html#httpRange-14⟩, 2002.

[Wang, 2007] Xiaoshu Wang. URI Identity and Web Architecture Revisited ⟨http://dfdf.inesc-id.pt/tr/web-arch⟩, 2007.

[Wittgenstein, 1953] Ludwig Wittgenstein. *Philosophical Investigation*. Blackwell, second edition, 1953.