



## LarKC

*The Large Knowledge Collider*

*a platform for large scale integrated reasoning and Web-search*

FP7 – 215535

---

### D2.1.1

## Selection and Retrieval Methods

---

H. Cunningham, A. Roberts, Y. Li, A. Kiryakov, L. Schooler, J. Quesada, H. Neth, E. Della Valle, D. Braga, N. Zhong

<b>Document Identifier:</b>	LarKC/2008/D2.1.1/V1.0
<b>Class Deliverable:</b>	LarKC EU-IST-2008-215535
<b>Version:</b>	1.2 final
<b>Date:</b>	September 4, 2009
<b>State:</b>	final
<b>Distribution:</b>	internal

## EXECUTIVE SUMMARY

This work package will construct methods for the retrieval and selection of propositions contained in large-scale semantic repositories. Such retrieval and selection are needed to support ceiling-free reasoning: we need to be able to dynamically reduce or expand the data set we're working with depending on factors such as cost of processing or confidence in result. To do this we will exploit methods from Information Retrieval, Machine Learning, Cognitive Science and Stream Management Systems. In most cases the relevant methods will need adaptation to the new context, and will apply in some cases of the selection task and not others.

## DOCUMENT INFORMATION

<b>IST Project Number</b>	FP7 – 215535	<b>Acronym</b>	LarkC
<b>Full Title</b>	Large Knowledge Collider		
<b>Project URL</b>	<a href="http://www.larkc.eu/">http://www.larkc.eu/</a>		
<b>Document URL</b>	<a href="http://wiki.larkc.eu/LarkcProject/WP2">http://wiki.larkc.eu/LarkcProject/WP2</a>		
<b>EU Project Officer</b>	Stefano Bertolo		

<b>Deliverable</b>	<b>Number</b>	2.1.1	<b>Title</b>	Selection and Retrieval Methods
<b>Work Package</b>	<b>Number</b>	2	<b>Title</b>	Selection and Retrieval



<b>Date of Delivery</b>	<b>Contractual</b>	M6	<b>Actual</b>	31-Sept-08
<b>Status</b>	1.2 final		final <input type="checkbox"/>	
<b>Nature</b>	prototype <input type="checkbox"/> report <input checked="" type="checkbox"/> dissemination <input type="checkbox"/>			
<b>Dissemination Level</b>	public <input checked="" type="checkbox"/> consortium <input type="checkbox"/>			

<b>Authors (Partner)</b>	University of Sheffield, OntoText, MPG, WICI, CEFRIEL			
<b>Resp. Author</b>	H. Cunningham		<b>E-mail</b>	hamish@dcs.shef.ac.uk
	<b>Partner</b>	University of Sheffield	<b>Phone</b>	+44 114 222 1891

<b>Abstract (for dissemination)</b>	This work package will construct methods for the retrieval and selection of propositions contained in large-scale semantic repositories. Such retrieval and selection are needed to support ceiling-free reasoning: we need to be able to dynamically reduce or expand the data set we're working with depending on factors such as cost of processing or confidence in result. To do this we will exploit methods from Information Retrieval, Machine Learning, Cognitive Science and Stream Management Systems. In most cases the relevant methods will need adaptation to the new context, and will apply in some cases of the selection task and not others.
<b>Keywords</b>	Selection, retrieval, WP2, parameters

<b>Version Log</b>			
<b>Issue Date</b>	<b>Rev No.</b>	<b>Author</b>	<b>Change</b>
3/June/2008	1	Hamish Cunningham	0.1
6/Sept/2008	2	Hamish Cunningham	1.0 draft
17/Sept/2008	3	Hamish Cunningham	1.0 incorporating internal reviewer comments and more input from all partners
17/Dec/2008	4	D. Braga and E. Della Valle	1.1 better integration of contribution about Streams.
18/Dec/2008	5	Hamish Cunningham	1.1 final (new introduction)
15/Aug/2009	6	Yaoyong Li	1.2 (adding section Semantic spaces for lexicon in Chapter 5; changing introduction and conclusion of Chapter 5)
04/Sep/2009	7	Danica Damljanovic	1.2 final (editorial)

## PROJECT CONSORTIUM INFORMATION

Acronym	Partner	Contact
Semantic Technology Institute Innsbruck <a href="http://www.sti-innsbruck.at">http://www.sti-innsbruck.at</a>	STI 	Prof. Dr. Dieter Fensel Semantic Technology Institute (STI) Innsbruck, Austria E-mail: dieter.fensel@sti-innsbruck.at
AstraZeneca AB <a href="http://www.astrazeneca.com/">http://www.astrazeneca.com/</a>	ASTRAZENECA 	Bosse Andersson AstraZeneca Lund, Sweden E-mail: bo.h.andersson@astrazeneca.com
CEFRIEL SCRL. <a href="http://www.cefriel.it/">http://www.cefriel.it/</a>	CEFRIEL 	Emanuele Della Valle CEFRIEL SCRL. Milano, Italy E-mail: emanuele.dellavalle@cefriel.it
CYCORP, RAZISKOVANJE IN EKSPERIMENTALNI RAZVOJ D.O.O. <a href="http://cyceurope.com/">http://cyceurope.com/</a>	CYCORP 	Michael Witbrock CYCORP, RAZISKOVANJE IN EKSPERIMENTALNI RAZVOJ D.O.O., Ljubljana, Slovenia E-mail: witbrock@cyc.com
HÄ́uchstleistungsrechenzentrum, Universitaet Stuttgart <a href="http://www.hlrs.de/">http://www.hlrs.de/</a>	HLRS 	Dr. Georgina Gallizo HÄ́uchstleistungsrechenzentrum, Universitaet Stuttgart Stuttgart, Germany E-mail: gallizo@hlrs.de
Max-Planck-Institut für Bildungsforschung <a href="http://www.mpib-berlin.mpg.de/index.js.en.htm">http://www.mpib-berlin.mpg.de/index.js.en.htm</a>	MAXPLANCKGESELLSCHAFT 	Dr. Lael Schooler, Max-Planck-Institut für Bildungsforschung Berlin, Germany E-mail: schooler@mpib-berlin.mpg.de
Ototext Lab, Sirma Group Corp. <a href="http://www.ontotext.com/">http://www.ontotext.com/</a>	ONTO 	Atanas Kiryakov, Ototext Lab, Sirma Group Corp. Sofia, Bulgaria E-mail: atanas.kiryakov@sirma.bg
SALTLUX INC. <a href="http://www.saltlux.com/EN/main.asp">http://www.saltlux.com/EN/main.asp</a>	Saltlux 	Kono Kim SALTLUX INC Seoul, Korea E-mail: kono@saltlux.com
SIEMENS AKTIENGESELLSCHAFT <a href="http://www.siemens.de/">http://www.siemens.de/</a>	Siemens 	Dr. Volker Tresp SIEMENS AKTIENGESELLSCHAFT Muenchen, Germany E-mail: volker.tresp@siemens.com
THE UNIVERSITY OF SHEFFIELD <a href="http://www.shef.ac.uk/">http://www.shef.ac.uk/</a>	Sheffield 	Prof. Dr. Hamish Cunningham THE UNIVERSITY OF SHEFFIELD Sheffield, UK E-mail: h.cunningham@dcs.shef.ac.uk
VRIJE UNIVERSITEIT AMSTERDAM <a href="http://www.vu.nl/">http://www.vu.nl/</a>	Amsterdam 	Prof. Dr. Frank van Harmelen VRIJE UNIVERSITEIT AMSTERDAM Amsterdam, Netherlands E-mail: Frank.van.Harmelen@cs.vu.nl

---

<p>THE INTERNATIONAL WIC INSTITUTE, BEIJING UNIVERSITY OF TECHNOLOGY <a href="http://www.iwici.org/">http://www.iwici.org/</a></p>	<p>WICI</p> 	<p>Prof. Dr. Ning Zhong THE INTERNATIONAL WIC INSTITUTE Maebashi, Japan E-mail: <a href="mailto:zhong@maebashi-it.ac.jp">zhong@maebashi-it.ac.jp</a></p>
<p>INTERNATIONAL AGENCY FOR RESEARCH ON CANCER <a href="http://www.iarc.fr/">http://www.iarc.fr/</a></p>	<p>IARC2</p> 	<p>Dr. Paul Brennan INTERNATIONAL AGENCY FOR RESEARCH ON CANCER Lyon, France E-mail: <a href="mailto:brennan@iarc.fr">brennan@iarc.fr</a></p>

---

# CONTENTS

1	INTRODUCTION	1
1.1	Data Model . . . . .	2
1.2	Parameters . . . . .	3
1.3	Methods . . . . .	3
2	SELECTION AND INFORMATION RETRIEVAL	7
2.1	IR Uses . . . . .	7
2.2	IR Processes . . . . .	8
2.3	IR Models . . . . .	8
2.4	IR and Semantics: old enemies, new friends? . . . . .	9
2.5	IR, Selection, and WP2 . . . . .	9
3	ACTIVATION, SELECTION AND ACT-R	12
3.1	Overview of the ACT-R Cognitive Architecture . . . . .	12
3.2	The Declarative Module . . . . .	14
3.2.1	Conducting Search . . . . .	14
3.2.2	Stopping Search . . . . .	14
3.2.3	Activation Reflects Relevance . . . . .	15
3.2.4	Activation and Retrieval Probability . . . . .	16
3.2.5	Summary . . . . .	16
3.3	Using ACT-R Activation to Predict and Retrieve Relevant Triples (MPG) . . . . .	17
3.3.1	The Statistical Analysis of How Triples are Needed to Achieve Processing Goals . . . . .	17
3.3.2	Selection Based on ACT-R Activation . . . . .	18

---

3.4	Declarative Memory and Multiple Information Granule Networks (WICI) . . .	18
3.4.1	Outline . . . . .	18
3.4.2	An Investigation of Information Retrieval and Selection in Human Memory System from the View of Granular Reasoning . . . . .	20
3.4.3	Organization of Multiple Information Granule Networks . . . . .	21
3.4.4	Case Study: Travel Planning Demo . . . . .	22
3.5	References . . . . .	24
4	SPREADING ACTIVATION IN RDF GRAPHS	26
4.1	Relation to Existing Projects and Tools . . . . .	26
4.2	The Activation Model . . . . .	28
4.3	Using Spreading Activation for Selection . . . . .	29
4.4	Relevance to Other Reasoning Tasks . . . . .	31
4.5	References . . . . .	32
5	GEOMETRICAL SEMANTIC SPACES	33
5.1	Information retrieval . . . . .	34
5.2	Semantic space for lexicon . . . . .	36
5.3	Quantum Mechanics . . . . .	38
5.4	Analogy between QM and IR . . . . .	40
5.5	QM Phenomena in IR . . . . .	41
5.6	Quantum Logic and its application in document retrieval . . . . .	42
5.7	Tensor product and its application in combination of meaning of individual words . . . . .	44
5.8	Conclusion and future work . . . . .	46
5.9	References . . . . .	47

---

6	SELECTION METHODS FROM DATA STREAM MANAGEMENT SYSTEMS	48
6.1	Selection in C-SPARQL . . . . .	48
6.2	A sketch of the new features introduced by C-SPARQL . . . . .	50
6.2.1	RDF Stream Data Type . . . . .	50
6.2.2	Windows . . . . .	50
6.2.3	Query Registration . . . . .	51
6.2.4	Aggregation in C-SPARQL . . . . .	52
6.2.5	Timestamp Function . . . . .	53
6.3	Execution of C-SPARQL Queries . . . . .	54
6.4	References . . . . .	55
7	APPENDIX 1: FIS PAPER ABOUT STREAMS IN LARKC	56
	BIBLIOGRAPHY	67

# 1 INTRODUCTION

LarKC work package 2 is about retrieval and selection of propositions with respect to large-scale datasets (or RDF [20] multi-graphs, as in SPARQL [30]). Retrieval is the task of deriving or extending the dataset from external datasources; if the dataset is hosted by a semantic repository, this is the process of population of the repository, e.g. by extracting RDF triples from text or from other type of data. Selection is the task of selecting a portion of the dataset which is relevant to the reasoning task and context; selection can be also considered as sampling in some reasoning scenarios.

Retrieval and selection are needed to support ceiling-free reasoning: we need to be able to dynamically reduce or expand the data set we are working with depending on factors such as cost of processing or confidence in result.

To do this we will exploit methods from information retrieval, machine learning, cognitive science, and stream management systems. In most cases the relevant methods will need adaptation to the new context and will apply in some cases of the selection task and not others.

The baseline results of this work package will be made publicly available as an open source project.

Below we introduce the parameter space of the work package as follows:

- The rest of this chapter summarises key elements of the data model, the parameters relevant to choice of alternative selection methods and the principal characteristics of those methods.
- Chapter 2 contextualises the work within the wider field of Information Retrieval.
- Chapter 3 covers approaches to selection derived from cognitive models of attention and memory, and presents in detail the ACT-R Cognitive Architecture.
- Chapter 4 turns to an alternative, less cognitively rich but possibly more scaleable approach that models spreading activation in RDF graphs.
- Chapter 5 turns to geometrical Information Retrieval models and their use for triple selection, including a discussion of new work which exploits analogies with quantum mechanics.
- Chapter 6 turns to the problem of streaming RDF data and details the ways in which we hope to exploit previous work in stream management systems for selection. The

approach is contextualised in Appendix 1, a paper from the Future Internet Symposium about streams and LarkC.

## 1.1 Data Model

Here we provide definitions of key elements of our data model, in the context of which the selection and the retrieval tasks are defined (a more detailed and formal definition is subject of definition in WP1 or WP5).

An RDF **graph** represents a set of RDF triples, as specified in [20]. RDF is the fundamental data format of the Semantic Web, designed to allow easy representation and integration of diverse structured data. The atomic data element is a triple of the format  $\langle \text{Subject}, \text{Predicate}, \text{Object} \rangle$ , e.g.  $\langle \text{John}, \text{loves}, \text{Mary} \rangle$  or  $\langle \text{Mary}, \text{hasBirthday}, "14.11.1972" \rangle$ . In the graph abstraction, each triple defines an edge directed from the Subject to the Object and labeled with the Predicate. The nodes of the graph could be URIs (unified resource identifiers, e.g. a URL), blank (auxiliary nodes), or an XML literal. Predicates are always URIs. For most purposes, literals are not allowed in subject position, i.e. they cannot be the start of an edge in the graph - intuitively, literals are used to describe resources identified by URIs, but there is no sense in describing literals because they represent a primitive data value.

Intuitively, a **dataset** is a graph that integrates multiple RDF graphs in such a way that the graphs can still be distinguished and managed separately. Technically, the interpretation of the term dataset follows the definition in the specification of the SPARQL query language [30].

- Dataset  $D = \{ UG, \langle N_1, G_1 \rangle, \dots, \langle N_k, G_k \rangle \}$
- $UG$  is an RDF graph, called the default graph of the dataset
- each pair  $\langle N_i, G_i \rangle$  introduces a **named graph**, where  $N_i$  is an URI, which represents the name of the RDF graph  $G_i$ , a part of the dataset

Formally, a dataset could be represented as an RDF multi-graph, which in turn can be represented as set of quadruples  $\langle S, P, O, G \rangle$ , where the first three elements  $\langle S, P, O \rangle$  represent an RDF statement and the fourth element is the name of the named graph it belongs to.

A **triple** is part of a dataset, i.e. a multi-graph represented as a subset of its quadruples. Triplets are also named, i.e. each triplet is associated with a unique name.

The rationale behind the above augmentations of the RDF data model is that named graphs allow for tracking provenance, for example when multiple graphs from different sources are merged or referenced. Triplesets allow for designation or tagging of parts of datasets, e.g. for the purposes of selecting parts of it.

## 1.2 Parameters

For any particular task the choice of selection method depends most crucially upon three factors:

**Cost/Benefit.** The affordability of different scales of reasoning process and the expected cost-per-statement of the process. In other words, both the size and nature of the propositional content under consideration and the intended nature of the reasoning process must be taken into account when choosing a selection method.

**Provenance.** The source of a proposition is evidence of its relevance to a particular reasoning task. For example, propositions that have been derived from or linked to a document collection can be selected by using clustering and search methods from information retrieval.

**Context.** The context of the reasoning process can be used as a filter on the relevance of propositional content. For example, a degree of similarity to content on the user's desktop or in their browser history is an indication of relevance in certain conceptual search scenarios. Determination of the optimal mapping between these three factors and the selection methods that the project will adapt and develop requires quantification of the performance of each method when operating according to parameters derived from a representative sample of task environments. In other words, we must apply empirical measures to gather data for our theory of selection and of selection method choice.

These rather general comments are complemented by a more specific discussion of metrics in section 4 (Measurement) of D2.1.2 (Apparatus).

## 1.3 Methods

Information retrieval (IR) models have proven their Web-level scalability and ability to generalize over contradictory data and the most straightforward derivations of these types of method, applied to triple selection relative to the above parameters, will form the baseline

by which we measure progress on more advanced methods. Promising results in previous work include

- [31] extracted a virtual document for each URI reference in a RDF triple store (or equivalently each node in a RDF graph).
- [5] explored virtual documents of the URI references to build a system for searching and browsing RDF triples based on a vector space retrieval model.
- [14] presented an A-Box summarization technique for efficiently searching and browsing RDF triples.

(For more details see the section *Anatomy of a Selection Plugin* in deliverable D2.1.2, Apparatus.)

Beyond the baseline, we will investigate methods based on activation-based approaches, cognitive memory models, geometrical semantic spaces, and stream management systems. LarKC's plug-in design means that we will be able to experiment with a variety of selection methods briefly introduced in this section; more detailed descriptions are provided in separate sections below.

In **activation based** approaches, datasets are treated by analogy to cognitive models of humans memory. Relevant memory elements (e.g. triples) are selected on the basis of their *level of activation*. Activation is determined through RDF priming - the well-known connectionist AI approach for spreading activation over artificial neural networks, adapted to operate over RDF graphs. The basic idea is that memory elements, representing the parameters (see the preceding section), are primed with a certain level of "energy", which is then spread to their neighbors in the graph. When the process is over, the *active* part of the graph is the desired selection; its size can be tuned through the activation threshold. This model, will be used also for other reasoning tasks, e.g. activations can be used as a currency for cost-based reasoning.

Empirical evidence from **cognitive science** suggests that standard logics are too brittle to represent phenomena such as contextual disambiguation or lexical priming. This insight has led, over several decades, to a range of research streams including connectionism, which adopts a physiological analogue of the brain and experiments with learning algorithms due to Hebb and others. Connectionist research spans a continuum from the neurophysiological (e.g. how can we best model the patterns of neural activation and activity flow as we see them in our MRI scanners?) to the symbolic (e.g. Smolensky's "harmony" has been a long-running attempt at layering symbolic processes on neural models).

---

ACT-R's symbolic declarative memory instantiates a view of human memory as performing near optimal retrieval of relevant information based on the statistical structure of environmental events. This statistical structure is captured in the pattern of connections between declarative memory structures.

More recently parallels between empirical knowledge of cognition and some of the characteristics of quantum mechanics has led to a **semantic space model** in which vector spaces stand in for the neurophysics and mechanisms such as Hilbert spaces and uncertainty are investigated as vehicles for computation (van Rijsbergen, *The Geometry of Information Retrieval*, 2004; Widdows, *The Geometry of Meaning*, 2004). Vector space models have a point of contact with cognitive scientists working on geometrical approaches to conceptual structures (see e.g. Gardenfors, *Conceptual Spaces*, 2000).

**Data streams** are unbounded sequences of time-varying data elements; they occur in a variety of modern applications, such as network monitoring, traffic engineering, sensor networks, RFID tags applications, telecom call records, financial applications, Web logs, click-streams, etc. Processing of data streams has been largely investigated in the last decade, specialized Data Stream Management Systems (DSMSs) have been developed, and features of DSMSs are becoming supported by major database products, such as Oracle and DB2. Data streams methods and techniques can be applied to all steps of the LarKC pluggable framework (see Section 7). In particular, selection and retrieval in the context of data streams is twofold. One aspect concerns the identification, within multiple candidate data streams, of the most relevant one to a given LarKC application. Another aspect concerns the selection, from within a data stream, of a limited amount of information so as to be able to deal with massive stream inputs while considering in the application only a limited amount of data, extracted according to a well-defined selection conditions.

In summary, we will work on five types of method:

- Minimalist techniques derived from standard Boolean retrieval models,
- Activation spreading approaches, which add context-sensitivity to information retrieval, based on biologically inspired connectionist networks associated with weighted RDF graphs,
- Activation spreading approaches inspired by cognitive models of human memory,
- Geometrical semantic spaces as a more far-reaching step on the road to cognitively plausible logics that support intelligent systems.
- RDF streams and C-SPARQL, a first rough adaptation of Stream Management Systems techniques for the Semantic Web.

To do this we will exploit methods from Information Retrieval, Machine Learning, Cognitive Science, and Stream Management Systems. In most cases the relevant methods will need adaptation to the new context, and will apply in some cases of the selection task and not others.

## 2 SELECTION AND INFORMATION RETRIEVAL

Selection and ranking problems bear a number of similarities to that of Information Retrieval (IR) and we will exploit a number of techniques from that field in this WP. We can briefly summarise the field of IR in terms of its **uses**, the **processes** that make up its systems and the **models** that these processes are based on.

### 2.1 IR Uses

Using a reasonably broad definition, IR is used for querying, classification, clustering and conceptual search:

- Querying (often called "ad hoc querying" to capture the unpredictable nature of the request) is the type of IR provided by Google etc.: the user types in keywords and phrases and the system returns a list of documents.
- Classification (or "static query") is where a set of categories for documents exists and the system's task is to assign new documents to one or more categories. Normally the categories have simple labels similar to query keywords.
- Clustering, or unsupervised navigation, is similar to classification in the case where the set of categories does not exist and has to be approximated by the machine. It is typically an aid to the exploration of unknown document spaces and doesn't require query keywords or category labels.
- Conceptual search moves beyond keywords (and their boolean manipulation) and provides a spectrum of logical semantic query methods that exploit reasoning over the propositional content of documents.

In each case IR users may be able to

- provide "relevance feedback" by commenting on the quality or relevance of the results
- refine queries based on previous results by adding or subtracting search terms

In relevance-based systems an important function is the ranking of results from most to least likely to satisfy the user's information need.

In the case of conceptual search, the information returned may not simply be a set of documents but information from a knowledge base, some of which may have been extracted from documents, some from structured data sources, etc. At this point the boundary between IR and other information systems begins to blur, and in fact this situation is common in the use cases (see for example the *Monograph production* and *Genome-wide Association Studies* sections of deliverable D7b1.1a from the cancer research use case).

## 2.2 IR Processes

Typical IR processes include:

- data capture, including crawling in the case of hypertext, or stream management in the case of news or textual database feeds
- indexing, including Information Extraction for Semantic Annotation where conceptual search is supported, and possibly personalised according to user context (the contents of the desktop or browser history, for example)
- searching, where user queries, category lists or logical statements are compared with indexed documents and a (often ranked) set of candidates returned

## 2.3 IR Models

We can identify five types of model typically used in IR systems:

- inverted file, the simplest and oldest, generally including boolean operators and very widely used for large datasets, especially where some additional mechanism for judging relevance is available (e.g. this is probably what Google uses, but Google also exploits the link structure of the web in its PageRank algorithm)
- vector space (used in e.g. Apache Lucene), where documents and queries are represented as multidimensional vectors which makes them amenable to techniques such as latent semantic indexing (now back on the agenda due to optimisations made possible by work on Random Indexing)
- probabilistic, which is an open research field but not very common in end-user systems at present

- various cognitively inspired systems including self-organising maps and other network structures
- logical semantic, based on reasoning over semantic annotation (from IE)

## 2.4 IR and Semantics: old enemies, new friends?

IR old-timers say: "we tried controlled vocabularies organised in hierarchies in the 60s and 70s and it didn't work then"<sup>1</sup>.

Semantics people say: "Google doesn't let us ask logical queries" and "language has meaning, not just word counts".

But: there's a new convergence in some respects, e.g.:

- the semantics community are interested in IR's proven web-level scaleability and ability to generalise over contradictory data
- the IR community are looking to new models that can capture a richer set of qualities in documents, and an important development here is logics that operate over geometrical models, or semantic spaces (below)

A hypothesis: these new models may be the common ground that brings logic-based knowledge representation and statistical text modelling together. An important source of support for these new models is the empirical record from cognitive science, and it is to selection methods that draw inspiration from this record that we turn in section 3. below.

## 2.5 IR, Selection, and WP2

We have been talking at a general level about IR and its current and future relationship to semantic technology. To finish our discussion, a note about the specifics of the work planned for WP2.

The first and clearest task that we wish to address is that of subsetting a knowledgebase, i.e. as a process which takes a repository, a context (which we'll define variously as a set

---

<sup>1</sup>This statement deserves some comment given that controlled vocabularies are in common use particularly in medical informatics but also in multimedia indexing e.g. at the BBC archives. The key point is that these uses are for high-value curated content and not for everyday information management needs. For more on this question see section 1 of deliverable D2.1.2.

of documents, or an activation state relative to a reasoning history, etc.) and returns a tripleset of those triples most relevant to the context. At an API level (but see also D2.1.2, Apparatus):

```
1 public TripleSet select(  
2   ReasoningTask task, SelectionContext context, TripleSet model  
3 );
```

where the context plus the reasoning task are the raw materials which we use to decide which portion of the dataset to return. For example, if the problem scenario is to identify likely factors to associate SNPs (known genetic polymorphisms) from a large body of literature:

- the task is a SPARQL query against the repository
- the context is the annotated corpus
- the model is the semantic repository (or some previously derived subset of it)
- the return value is a tripleset representing a new subset relevant to the task and context

So far so good: our task is to subset, our methods drawn from those from several areas of IR and cognitive science.

It is evident however, in order to be maximally relevant to end-users in general and our use-case WPs in particular, the LarKC platform has to support a number of off-line processes in addition to those that are obviously part of the pipeline loop. (In fact the pipeline is probably best seen as a broad brush generalisation that does not capture all the specificities of LarKC platform behaviour in particular cases.)

One example of this is methods that populate or augment the content of KBs, and it is here that we seem in need of methods that may well be categorised as "retrieval": creation of triples that map relational data into OWL or that interrelate partially compatible conceptual networks; extraction of triples from textual data. Another example: methods that subset a document space prior to extraction of triples.

Therefore we hold open the option of adding the concept of a retrieval plugin to the picture sketched above (and in more detail in D2.1.2). Retrieval plugins may take a set of DataSets plus a RetrievalContext as input and return a new (augmented, and/or merged) DataSet as output (whereas selection plugins take a DataSet and return a subset of that DataSet). Another variant takes a SelectionContext as input and returns a new (subsetting context).

So where as selection is about subsetting, retrieval is defined as creation, extension, or augmentation. Selection is potentially relevant to retrieval: first subset the document set, then do the "retrieval", i.e.

1. retrieve (subset) relevant docs
2. retrieve triples from the document space
3. selection from resultant KB (subset)

Actual implementation of these types of processes (and their plugin instantiations) are the preserve of our use cases, particularly the two halves of WP7. Therefore WP2 will not develop retrieval components as such, but may in future include API to describe those processes that we deliver in WP7.

### 3 ACTIVATION, SELECTION AND ACT-R

This section discusses work from MPG and WICI on using the ACT-R framework for selection and retrieval in LarKC.

The contributions of WICI and MPG to task 2.3 depend on the spreading activation based model of memory that is central to the ACT-R (Atomic Components of Thought Rational) theory of cognition (Anderson et al., 2004; Anderson, 2007; [act-r.psy.cmu.edu](http://act-r.psy.cmu.edu)). Before describing the work, we provide a brief review of ACT-R, paying special attention to its memory model.

#### 3.1 Overview of the ACT-R Cognitive Architecture

The ACT-R cognitive architecture is a unified theory of cognition that accounts for a diverse set of phenomena ranging from visual search to scientific discovery. A central distinction in ACT-R is between declarative knowledge (“knowing that”) and procedural knowledge (“knowing how”). ACT-R models procedural knowledge with sets of production rules, which are reminiscent of logical conditionals. The fundamental declarative representation in ACT-R is the chunk. Basically, a chunk is a set of slot-value pairs and so is more or less comparable to an rdf-triple.

As ACT-R’s production rules iteratively work to achieve goals by calling on semi-autonomous processing modules, it could be one place to look for ideas about how to thread reasoning and selection plugins. As illustrated in Figure 3.1, the ACT-R architecture interacts with the external environment through modules that contain various inputs (e.g., visual data chunks) and outputs (e.g., manual motor commands). Similarly, the cognitive system is internally de-composed into processing modules that handle control states (goals), declarative memory (memory chunks) and problem state representations. The production system coordinates communication between the modules by making requests to them (e.g., queries to declarative memory) and communicating results between them in the form of chunks. The overall behavior of the system is cyclical: On each cycle the central production system inspects the current buffer contents and selects the most appropriate production. The execution of a production may transform the contents of various buffers, but buffer contents are also modified by the internal operations of the modules themselves; new visual chunks appear due to new visual inputs. To give the flavor of how processing unfolds, consider how the system might solve a simple algebra problem, such as  $3x = 9$ . The system would, in effect, copy the equation from the visual buffer to the problem state buffer, productions

would transform the problem to  $x = 9/3$ , and send a query to the declarative memory module to retrieve the chunk that matches (`isa division-fact, numerator 9, denominator 3, answer ?number`) and so on. The production rules for solving algebra would coordinate these steps.

A crucial aspect of the system is the parallel activity of multiple modules coordinated by the central production system. The modules do not care how the other modules go about doing what they do. All that matters is that results are returned in the form of a chunk that the production system knows how to handle. Because communication between modules is in the form of chunks, reasoning and decision making within the ACT-R architecture necessarily must be implemented as a continuous process operating upon partial information, rather than a one-shot event that has to consider the entirety of its potentially relevant knowledge at any one time.

We next consider in detail the declarative memory module which tackles the problem of retrieving relevant chunks.

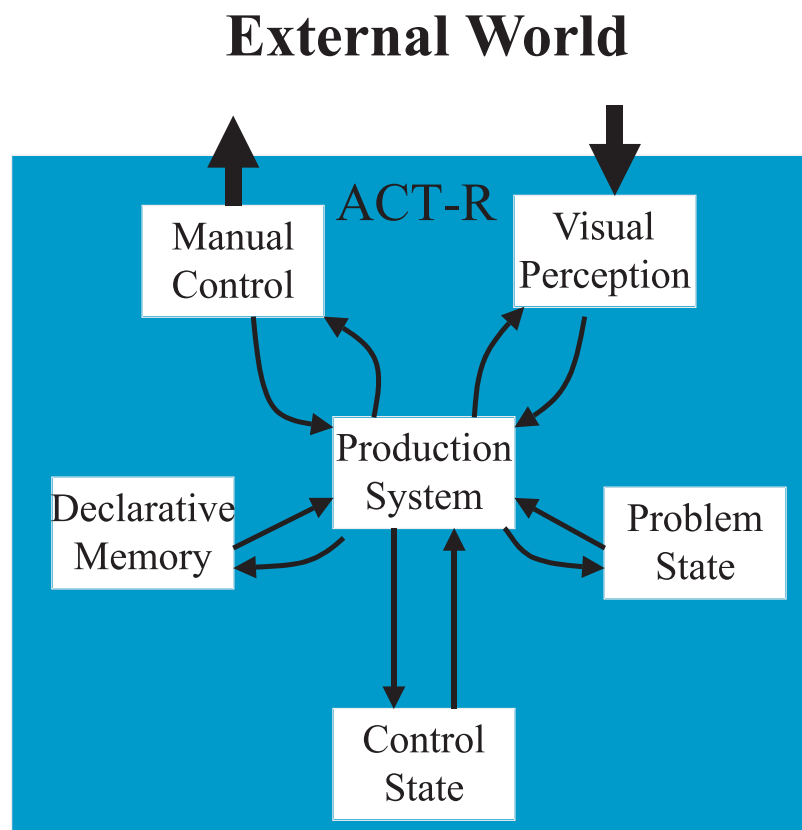


Figure 3.1: Basic structure of the ACT-R architecture (based on Anderson, 2005).

## 3.2 The Declarative Module

In ACT-R declarative memory and procedural memory interact through retrieval mechanisms that are sensitive to patterns with which information is needed to achieve processing goals. In essence, the chunks that the system retrieves at a given point can be seen as a bet about what will happen next. In this framework, human memory functions as an information retrieval system, and the elements of the current context constitute a query to long-term memory to which the memory system responds by retrieving the chunks that are most likely to be relevant.

### 3.2.1 Conducting Search

ACT-R makes the assumptions that information in long-term memory is stored in discrete chunks and that retrieval entails searching through these chunks to find a chunk that achieves some processing goal of the system. The explanatory power of the approach depends on the system's estimates of the probability that each chunk in long-term memory is the one sought. In keeping with common usage in library science, we call this the relevance probability. Any information retrieval system must strike a balance between recall, in this context the likelihood of finding the desired chunk, and precision, or the likelihood of retrieving only relevant chunks. In ACT-R, this balance is achieved through a guided search process in which the chunks are retrieved in order of their relevance probabilities, with the most promising chunks processed first.

### 3.2.2 Stopping Search

At some point, the information retrieval system must terminate search for further chunks. If  $p$  is the relevance probability,  $C$  is the cost of attempting to match a chunk against, say, a condition of a production rule, and  $G$  is the gain associated with successfully finding the target, then according to ACT-R the memory system should stop considering chunks when:

$$pG < C \tag{3.1}$$

In other words, the system stops searching for more memory chunks when the expected value ( $pG$ ) of the next chunk is less than the cost of considering it. If the next chunk to be considered has a relevance probability of less than  $C/G$ , search will be stopped.

The set up here involves searching for a unique target, but the retrieval criterion could just as well be used to obtain a selection of triples that could be fodder for the reasoning plugins.

### 3.2.3 Activation Reflects Relevance

In ACT-R, the activation of a declarative chunk reflects its relevance probability. Specifically, activation,  $A$ , equals the log odds (i.e.,  $\ln[p/(1-p)]$ ) that the chunk will be needed to achieve a processing goal (e.g., that it will match a condition of a production rule that fires). A chunk's activation,  $A_i$ , is calculated by a combination of the base-level strength of the chunk,  $B_i$ , and the  $S_{ji}$  units of activation the chunk receives from each of the  $j$  elements of the current context:

$$A_i = B_i + \sum_j S_{ij} \quad (3.2)$$

Contextual strength is measured in terms of associative ratios that approximate the likelihood ratios common in Bayesian statistics.

$$S_{ij} = \log \left( \frac{P(i|j)}{PI_i} \right) \quad (3.3)$$

The denominator of this ratio,  $P(i)$ , is the base rate probability of needing a particular chunk. The numerator,  $P(i|j)$ , is the conditional probability of needing the chunk given the presence of some cue  $j$ . The larger this ratio the better the cue, the greater the associative strength. The overall strength of the context is taken to be the sum of the associative ratios linking each individual cue in the context to that chunk. The  $S_{ji}$ 's are also equivalent to Pointwise Mutual Information (Church & Hanks, 1990).

A record's base-level strength is rooted in its environmental pattern of occurrence. Specifically,  $B_i$  is determined by how frequently and recently the record has been encountered, or processed, in the past:

$$B = \ln \left( \sum_{j=1}^n t_j^d \right) \quad (3.4)$$

where the chunk has been encountered  $n$  times in the past and the  $j$ -th encounter occurred  $t_j$  time units in the past. Finally,  $d$  is a decay parameter that captures the rate of forgetting in declarative memory, thus determining how much information about an item's encounter

frequency is retained in memory. Typically,  $d$  is set to equal  $-0.5$ , a value that has been shown to fit a wide range of behavioral data.

### 3.2.4 Activation and Retrieval Probability

Whether a memory record's activation exceeds the retrieval criterion is determined by a noisy process. The sources of noise include momentary fluctuations in a chunk's estimated gain, estimated cost, and the activation it receives from the current constellation of context elements. These context elements could be part of our external environment, such as words on a sign, or internal, such as our mood or recently activated chunks. Because of its variability, the activation of a chunk is better represented by a distribution of activation values than by a single value, with  $A_i$  (see Equation 3.2) as the distribution's expected value. In ACT-R, activation is modeled as a logistic distribution, which approximates a normal distribution.

The probability that a record will be retrieved, that is, that its activation will exceed the retrieval criterion, can be expressed as a logistic function:

$$\text{Probability of record retrieval} = \frac{1}{1 + e^{-(A-\tau)/s}} \quad (3.5)$$

where  $s$  captures momentary and permanent fluctuations in the activation level of record  $i$ . Parameter  $\tau$  equals  $\ln(C/(G - C))$ , a stopping rule that is equivalent to the  $p < C/G$  criterion from Equation 3.1 but transformed into the activation scale. The proportion of a record's activation distribution that is above the retrieval criterion,  $\tau$ , gives the probability that the particular record will be retrieved.

Typically, when modeling human cognition noise is added to activations to help capture variability in behavior. Sometimes you can retrieve the name of someone you meet at conference, while on another occasion you can not. Activation noise can be functional. Variability in the order in which chunks are considered can lead the system to explore different regions of declarative memory, perhaps enabling the system to resolve an impasse. In other applications it may be best to remove all activation noise to simplify the processing.

### 3.2.5 Summary

In brief, the relevance probability of each memory chunk is reflected in its distribution of activation. Chunks are searched in order of their activation until either a chunk is found that

satisfies a processing goal of the system or the activation of the next chunk to be considered is so low that it is not worth considering.

### 3.3 Using ACT-R Activation to Predict and Retrieve Relevant Triples (MPG)

We will undertake statistical analyses of how triples are needed to achieve processing goals and tailor the ACT-R activation equations to model these need statistics (3.3.1). We will develop a proof-of-concept-plugin (poc-plugin) that illustrates how ACT-R's activation equations could be used in selection (3.3.2).

#### 3.3.1 The Statistical Analysis of How Triples are Needed to Achieve Processing Goals

The key question in this section is to look at the system's usage history to see which triples have been relevant in the past and how these histories can be used to predict which triples will be relevant in the future. An object's history is a list of dates (e.g., years, days, seconds) on which the object has been relevant. In previous analyses, the objects have most often been words (Anderson & Schooler, 1991; Schooler & Anderson, 1997), but similar patterns have been found for files (Pitkow & Recker, 1996). The statistics of the use-case data sets will be analyzed and compared to the statistics of previously studied data sets. The data can consist of any discrete objects, such triples, articles, experiments, proteins, etc. In ACT-R, a chunk of information is relevant when it is retrieved from long term memory and matches a condition of a production rule. For the use-cases, we would need a concrete definition of relevance. For the IARC Cancer monographs, an article could be relevant when it is cited in another article. A person could be relevant, when they are cited in an article. In the city use case, a location could be relevant when someone decides to visit or a means of transportation could be relevant when someone decides to use it.

We will start this investigation by analyzing 15 million search queries carried out by visitors to `msn.com` sampled over one month. This data will be provided as part of the Workshop on Web Search Click Data, associated with the Second ACM International Conference on Web Search and Data Mining to be held in February 2009. These statistics will give us a sense of whether the informational demands placed on the search engine are anything like what we have found in the past analyses. Analyzing search terms should give us a sense of what we might expect when we move to an analysis based on triples.

A key quantity in ACT-R is activation, which can be interpreted as the log-odds of needing a chunk of information. This activation is decomposed into two parts. The first is the base level activation, which depends on its history. The recency, frequency, and spacings between when the triple has been needed in the past. The second component, associative activation, captures the association between elements of the triple and elements of other triples or context more generally. The basic question is how well ACT-R's activation equations can predict what triples and search terms will be relevant?

There are variants of ACT-R's basic activation equations that vary in terms of their computational demands. One focus of this section is to explore the trade offs between these computational demands and how well the equations predict relevance. Past work at ABC suggests that the simpler versions may be better at predicting future relevance than the more complex versions.

A convenient feature of how activation is calculated in ACT-R is that the contribution of associative and base level activation can easily be separated. That way, it should be possible to take the other associative methods (Dual's spreading activation; Quantum Semantics) that are being developed and combine those with ACT-R's base level activation.

### **3.3.2 Selection Based on ACT-R Activation**

The statistics of file access on one www repository suggest that activation methods should help LarKC hone in on relevant information (Pitkow & Recker, 1996; Pirolli, Pitkow & Rao, 1998). MPG will develop and test an ACT-R activation based proof-of-concept-plugin (poc-plugin) that demonstrates how these activation equations could be used in selection. By poc-plugin, we mean a retrieval plugin that would run on a triple-store, but perhaps one of limited size based on queries of limited complexity.

## **3.4 Declarative Memory and Multiple Information Granule Networks (WICI)**

### **3.4.1 Outline**

The main contributions of the WICI to WP2 (as well as to WP4) are to investigate human granular reasoning related functions, such as human declarative memory organization

and retrieval with multiple information granule networks, and to develop cognition-inspired computing at the Web scale.

In a philosophical sense, granular reasoning can describe a way of thinking that relies on the human ability to perceive the real world under various levels of granularity (i.e., abstraction) in order to abstract and consider only those things that serve a specific interest and to switch among different granularities (Hobbs, 1985; Yao, 2007; Yao and Zhong 2002). By focusing on different levels of granularity, one can obtain different levels of knowledge, as well as an in-depth understanding of the inherent knowledge structure. Granular reasoning is thus essential in human problem solving and would have a very significant impact on problem solving and reasoning at a Web scale. We take granular reasoning as a human intelligence inspired methodology and will develop specific methods for a reasoning process in a variable precision fashion.

ACT-R's declarative chunk representation provides a good framework for modeling granular reasoning. In every chunk, there is a special slot, called **IS-A**, which connects the current chunk to its category, which is also a chunk. During retrieval, both the probability to be retrieved and the retrieval time for each chunk are based on its activation, which is computed by adding the base level activation and associative activation from the context (Equation 3.2). In other words, whether a piece of information is retrieved or not is not only determined by the frequency and recency, but also the context elements. The introduction of the **IS-A** slot and associative activation for a declarative memory emphasizes the relationship between chunks in the memory system. According to the spreading activation theory of memory, declarative knowledge is represented in terms of chunks and associative pathways between chunks, which form a semantic network of concepts. A hierarchical structure is also present in this network, classifying concepts into more generic and more specific ones. Hence, ACT-R is very useful to represent multiple information granule networks as part of functions of granular reasoning plug-in at the Web scale.

Computational cognitive architecture models, such as ACT-R, in which the interaction among modules are assumed to play important roles in information processing in the human brain provide a theoretical framework for WICI's experimental work (Anderson et al, 2004). First (as proposed in Section 3.4.2, connected to WP4), we will perform a series of experiments on heuristics and granular reasoning in human problem solving that involves memory retrieval (such as retrieving heuristic rules) in multiple information granular networks and visual attention that involves checking the representation of the problem state. We will develop ACT-R models to predict when and how long the brain areas corresponding to memory retrieval and visual attention are active during the processes of problem solving, how the brain areas change when retrieving different information sources in different infor-

mation granular levels. We use data from fMRI recordings to test these models. Second (as proposed in Section 3.4.3), we will use the chunking structure in the declarative memory of ACT-R to build the plug-in systems for the organization of multiple information granule networks.

As a deliverable for WP2 (in connection to WP4), we will develop an information organization structure called multiple information granule networks and implement a trade-off algorithm for selecting and retrieving such networks based on the understanding of the question under constraints. The results can serve as a part of functions of granular reasoning plug-in.

### **3.4.2 An Investigation of Information Retrieval and Selection in Human Memory System from the View of Granular Reasoning**

The human brain can be regarded as a huge distributed knowledge base with multiple information granule networks. Humans can give a reasonable answer within a reasonable time by exploring variable precision when receiving a question (i.e., a reasoning problem). Thus, understanding the principles and mechanisms of information organization, retrieval and selection in human memory aims to find more cognition-inspired methods of information memory system, problem-solving and reasoning at the Web scale.

From the viewpoint of granular reasoning, data, information, and knowledge are arranged in multiple levels according to their granularity as multiple information sources (Yao 2007; Zhong et al. (Eds.), 2007). A higher level contains more abstract or general knowledge, while a lower level contains more detailed or specific knowledge. Reasoning and retrieval can be performed on various levels in single or multiple information sources. Results from a higher level may be imprecise but can be obtained faster. In contrast, one can move to a lower level to obtain more precise conclusion if more time is allowed. Hence, granular reasoning offers a multi-resolution reasoning scheme. One may choose a proper level of granularity to draw a desirable conclusion under certain constraints. Such a reasoning and retrieval scheme is commonly used by human for practical and real-time problem solving and decision-making. In this study, we will carry out two series of cognitive experiments to investigate the mechanism of information retrieval and selection in a variable precision fashion in human declarative memory.

Experiment 1 aims at answering the question of how human selects a suitable level of information granules (IGs) and retrieves in single or multiple information sources (ISs), which is

based on a trade-off between user needs and certain constraints. The experiment is divided into 4 cases by making different types of questions (tasks), namely, participants need to (1) use single IS and search an answer on single IG level in the IS, (2) use single IS, but search an answer on different levels of IGs in the IS, (3) use multi-IS, but search an answer on single IG in each IS, and (4) use multi-IS and search an answer on different levels of IGs in the ISs. The behavioral patterns and activities of the brain under these cases will be evaluated to understand the neural mechanism of information retrieval and selection in a variable precision fashion. The results will inspire us to develop an information organization structure with multiple information granule networks and a trade-off algorithm for selecting and retrieving on such networks based on the understanding of the question under constraints. The results will be a part of functions of granular reasoning plug-in (Badre and D'Esposito, 2007; Hu and Zhong, 2006; Zhong et al. (Eds.), 2007).

Experiment 2 is related to retrieval strategy and visual attention during problem solving. Different information has its own role in the problem solving process. For example, information with respect to a goal guides the search direction and information with respect to a situation is integrated to solve a problem. This experiment will use Sudoku to illustrate the role of different types of information granules during problem solving. In this experiment, memory load is also considered. Different memory loads affect the sufficiency of information maintenance and integration. In this study, with different memory loads, participants either retrieve information respecting to a goal or a situation. Based on the results, different behavior patterns and brain activities will suggest the mechanism of information granular types in problem solving, and provide evidence to support the relationship between an information granule retrieved and a task/problem when facing an unsettled situation (Qin et al., 2004).

For these cognitive experiments, a broadly used empirical methodology, functional magnetic resonance imaging (fMRI) will be used to investigate the neural mechanism of memory organization and information retrieval and to measure the dynamic change of BOLD effect in the brain. In addition, ACT-R will also be used to test and explain human's behaviors, as well as to clarify the relationship between cognitive modules and the brain (Anderson et al., 2004; Anderson, 2007).

### **3.4.3 Organization of Multiple Information Granule Networks**

A deliverable for WP2 is to provide an information organization structure with multiple information granule networks and implement a trade-off algorithm for selecting and retrieving on such networks based on the understanding of the question under constraints. The results will be connected to WP4 as part of functions of granular reasoning plug-in (Hu and Zhong,

2006; Zhong, Liu, and Yao, 2007). Information relevant to solve a problem is collected from global Web based distributed information sources, and these sources are organized by using ACT-R declarative memory representation under the need of granular reasoning. Furthermore, declarative knowledge is selected and retrieved based on the spreading activation theory of memory according to ACT-R.

### 3.4.4 Case Study: Travel Planning Demo

In addition to the series of cognitive experiments mentioned above, we will develop a travel planning system to demonstrate how granular reasoning can work in real-world problems. Firstly, we will design a multi-level granularity structure for information organization over large-scale semantic repositories. Secondly, we will implement a trade-off algorithm to switch between different levels of granularity, which is inspired by human intelligence in the problem solving process.

The information sources are organized with three levels, namely data level, information level, and knowledge level as shown in Figure 3.2. Although the information sources can be organized with more than three levels for real-world needs, the three-level model may be enough to demonstrate our methodology.

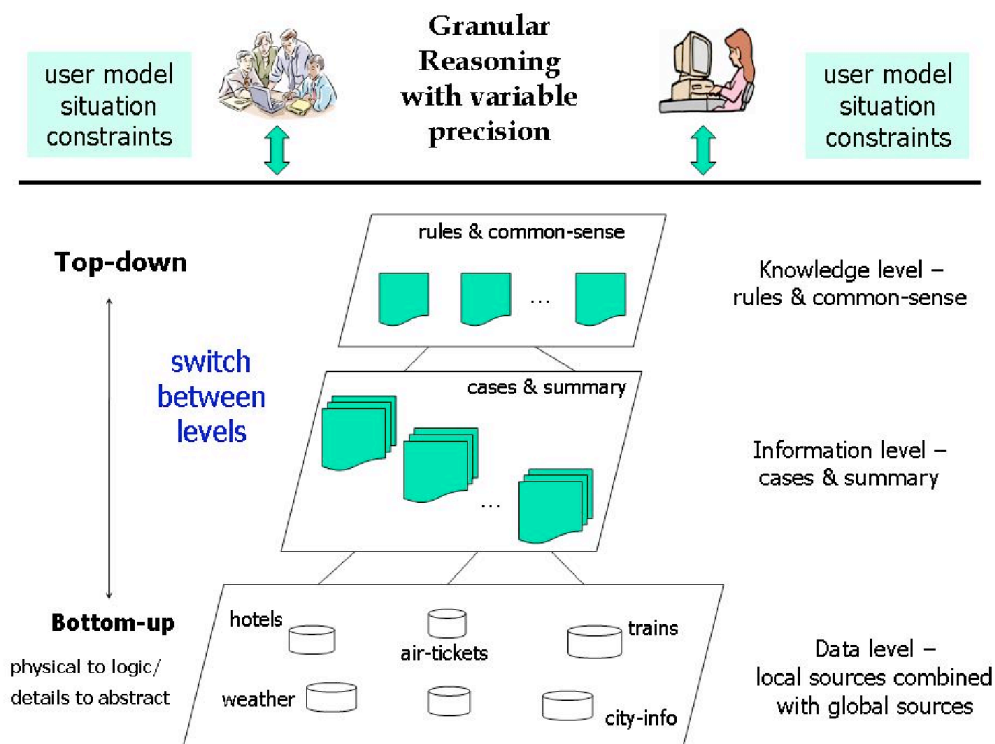


Figure 3.2: Travel planning system design with multiple information granule network.

The data level is built by coupling global semantic Web/social networks with local information sources, which is in the bottom of this information organization structure. The data sources can be represented as a set of RDF triples (or RDB) with activation weights, which are revised dynamically. In the specific case, the data level has the data stored in an organized way, such as air-ticket, hotel, weather, city-information, and so on, for the following processing.

The basic granule unit in the information level is case, which is formed by abstracting and summarizing multiple data sources in the data level. One or more cases can be classified/-grouped and represented as chunk(s). Furthermore, the cases/chunks in the information level are managed and changed according to associative and base level activations. For example, the plans for a business trip and for a tourist trip are different, because the situation and constraints are different. For a business trip, exactly following the schedule is always the first priority. For a tourist trip, such as for a student trip, the budget may be the first priority; in the case of a family trip, a comfortable trip needs to be considered, besides the budget. Therefore, in the difference cases, the way of abstracting and summarizing is different. The activation and weight for the multiple sources are also different. The algorithm of locating and selecting corresponding data sources is still a question.

The basic granule unit in the knowledge level is the rule (with common-sense), which is learned from cases in the information level (and/or from the data sources in the data level directly). One or more rules can be classified/grouped and represented as chunk(s) as well.

In order to carry out granular reasoning in a variable precision fashion, user related information also needs to be prepared by learning personalized user models with situation and constraints. Based on the preparation stated above, we can implement a trade-off algorithm for selecting and retrieving on such multiple information granule networks based on the understanding of the question under constraints.

As shown in Figure 3.2, both bottom-up and top-down processing are involved in this system, as information selection and retrieval will switch among levels based on the query from users. With support of the multiple granular structure and trade-off algorithm, the system can choose the proper granular level and then locate the necessary data sources effectively for the specific query and customer at the same time. More importantly, the result from the cognitive experiments may help us to understand the human problem solving mechanism deeply, and the principle can also be used to improve the system.

### 3.5 References

- Anderson, J.R. (1983). A Spreading Activation Theory of Memory. *Journal of Verbal Learning and Verbal Behaviour*, 22, 261–295.
- Anderson, J. R., & Schooler, L. J. (1991). Reflections of the environment in memory. *Psychological Science*, 2, 396–408.
- Anderson, J. R. (2005). Human symbol manipulation within an integrated cognitive architecture. *Cognitive Science*, 29(3), 313–341.
- Anderson, J. R., Bothell, D., Byrne, M.D., Douglass, S., Lebiere, C., Qin, Y. (2004). An Integrated Theory of Mind. *Psychological Review*, 111, 1036–1060.
- Anderson, J.R. (2007). *How can the human mind occur in the physical universe?* Oxford University Press, New York.
- Badre, D., & D’Esposito, M. (2007). Functional Magnetic Resonance Imaging Evidence for a Hierarchical Organization of the Prefrontal Cortex. *Journal of Cognitive Neuroscience*, 19(12), 2082–2099.
- Church, K. W., & Hanks, P. (1990). Word Association Norms, Mutual Information and Lexicography. *Computational Linguistics*, 16(1), 22–29.
- Hobbs, J.R. (1985) Granularity. *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, 432–435.
- Hu, J., & Zhong, N. (2006). Organizing Multiple Data Sources for Developing Intelligent e-Business Portals. *Data Mining and Knowledge Discovery, An International Journal*, 12 (2-3), 127–150.
- Pirolli, L., Pitkow, James, E., Rao, R, B., (1998). System for predicting documents relevant to focus documents by spreading activation through network representations of a linked collection of documents, U.S. Patent No.5835905, Washington, DC: U.S. Patent and Trademark Office.
- Pitkow, J.E., & M.M, Recker, (1996). Predicting Document Access in Large Multimedia Repositories. *Transactions on Computer-Human Interaction*, 3, 352–375.
- Schooler, L.J., & Anderson, J. R. (1997). The role of process in the rational analysis of memory. *Cognitive Psychology*, 32, 219–250.
- Qin, Y., Carter, C., Silk, E., Stenger, V. A., Fissell, K., Goode, A., & Anderson, J.R. et al. (2004). The Change of the Brain Activation Patterns as Children Earn Algebra Equation Solving. *PNAS*, 101(15), 5686–5691.

- Yao, Y.Y. (2007). *The Art of Granular Computing*. LNAI 4585. Springer, Berlin, 101–112.
- Yao, Y.Y., & Zhong, N. (2002). Granular Computing using Information Tables, in: Lin, T.Y., Yao, Y.Y., and Zadeh, L.A. (Eds.). *Data Mining, Rough Sets and Granular Computing*. Physica, Heidelberg, 102–124.
- Zhong, N., Liu, J., & Yao, Y.Y. (2007). Envisioning Intelligent Information Technologies from the Stand-Point of Web Intelligence. *Communications of the ACM*, 50(3), 89–94.
- Zhong, N., Liu, J., Yao, Y.Y., Wu, J.L., Lu, S., & Li, K. (Eds.) (2007). *Web Intelligence Meets Brain Informatics*. State-of-the-Art Survey. LNAI 4845, Springer.

## 4 SPREADING ACTIVATION IN RDF GRAPHS

In the activation based approach discussed here, the dataset subject to reasoning, is treated by analogy to humans's memory. The context and the reasoning task are used to prime the RDF graph via mechanism known as "spreading activation", which effectively propagates through the graph relevance; the most active part of the graph is considered selected.

The high-level rationale behind such model is to synergize the beauties of the connectionist models with those of the logical reasoning, performed on top of RDF graphs. The latter is the most standard contemporary KR and reasoning setting that LARKC is aiming to transform in a way which makes it appropriate for the challenges of the web, e.g. enormous data scale, inconsistent and incomplete data. Crucial requirement for reasoning in web context (such as the one of the web search engines), is that that reasoning tasks should be handled in predictable and satisfiable fashion under strict constraints on various parameters and resources (e.g. response time, memory, CPU time).

Spreading activation is a method which is naturally suitable to work under limited resources because at each phase there are usable results - in a nutshell, more energy and time will result in finer grained activations of larger portions of the RDF graph. The underlying models and techniques are relevant not only to selection, but also to other stages in the reasoning pipeline of LARKC, most notably this model provides good basis for cost-based reasoning.

### 4.1 Relation to Existing Projects and Tools

The approach for combination of symbolic and connectionist AI methods, adapted here, have been initially implemented in the course of an EC research project RASCALLI <http://www.ofai.at/rascalli/>, for the sake modelling the memory of virtual personal assistants. The conceptual grounds were derived from a hybrid system called DUAL. Within RASCALLI, Ontotext developed an engine called DualRDF, which allows for priming of RDF graphs. This engine was used as basis for several types of reasoners: closed-domain reasoner, open-domain reasoner (based on open-world assumption), analogy reasoner, etc. DualRDF is taken as starting point in the development of the selection plug-in based on spreading activation.

Technically, DualRDF is implemented on top of ORDI <http://ordi.sourceforge.net/> - a middleware framework for ontology management and semantic data integration. The data model of ORDI is the one of RDF, extended with named graphs and triple sets - precisely this model is agreed for usage in the data layer of LARKC's platform and shortly presented in

section 1.1 of this document. As a middleware platform ORDI allows for easy integration of various backends, which can be classified into two categories - RDF database (semantic repositories) and non-RDF data sources. Part of the ORDI platform is a backend adapter for relational databases, based on JDBC. Two of the most scalable semantic repository engines are already integrated with ORDI: YARS2 <http://sw.deri.org/2006/07/yars2/> and TRREE <http://www.ontotext.com/trree/>. TRREE is the engine under the hood of OWLIM - semantic repository best known for its capability to efficiently perform light-weight reasoning on top of billions of RDF triples. TRREE is also the only engine which supports ORDI's data model in its full richness, including triplesets - a feature, which allows efficient handling the "weighted RDF" model discussed in the subsequent section.

The relationships between DualRDF, ORDI, TRREE, OWLIM and Sesame <http://www.openrdf.org>, a RDF database framework are presented on the following figure.

Within RASCALLI, DualRDF was meant to serve under specific requirement. There were three types of memory (or categories of data):

- working memory (WM) - the most active part of the memory of the agent, analogous to human's working memory; for the purposes of the project, this volume of this memory was limited to the scale of few thousands of statements;
- long-term memory (LTM) - the full capacity of agent's own memory, including any specific knowhow and episodic knowledge; it's size was limited in the range up to hundred thousands of statements;
- library - common knowledge that the agent can retrieve when necessary, in the way in which people tend to consult encyclopedias and other literature; the volume of this data have been limited to hundred million statements.

The reasoning architecture were determined as follows:

- Semantic database - a DualRDF configuration, working on top of TRREE, which hosts all sorts of relevant data;
- Spreading Activation and Learning Module (SALM) - taking care of the spreading activation. SALM is implemented on top of generic platform for sparse matrix operations where spreading activation is modelled as multiplication of matrices. SALM is holding the WM plus its immediate graph neighborhood in the RDF graph - the resulting activations are synchronized with the semantic database. SALM is also taking care of more complicated hypothesis generation and learning tasks, related to the analogy reasoner.

The beauty of this design is that the spreading activation task is performed independently from the RDF repository. Further, its implementation on top of standard mathematical operations allows for adoption of a wide range of HPC software libraries and specialized massively parallel hardware (e.g. FPGA).

## 4.2 The Activation Model

Connectionism is an approach in AI and cognitive science, that models mental phenomena as the emergent processes of interconnected networks of simple units [http://en.wikipedia.org/wiki/Spreading\\_activation](http://en.wikipedia.org/wiki/Spreading_activation). The most popular connectionist systems are the so-called artificial neural networks (ANN), which mimic biological NN from the central neural system. The general concept is that ANN is a network neurons (atomic processing elements), which can exhibit complex global behavior, determined by the connections between them (synapses). While their biological realism is heavily criticized, ANN were proven to be useful non-linear statistical data modeling tools - one of their most typical application is pattern recognition, where the networks are trained (through techniques like backward propagation) to map specific input signals to desired output signals. For the sake of clarity, we should state that the spreading activation based techniques to be investigated in LARKC are not directly related to the ANN and their applications as statistically trained recognizers.

Spreading activation (SA) is fundamental connectionist technique, used not only in ANN, but also in semantic networks of various sorts. The general notion of SA, which can be described as follows:

- suppose there is a directed graph (the network), where each node bears certain level of activation and each edge has a weight; the activations and weights are typically real number between 0 and 1;
- in its initial state, the activation levels all nodes are set to 0; the weights of the edges indicate the strength of the relationship between the nodes, e.g. in a network were each node represents a concept, the weights could represent the degree of some sort of association between the concepts;
- in the process of spreading activation the levels of activation change, while the weights of the edges usually remain static - they are considered part of the topology of the network;

- the process starts when non-zero activation levels are set to few of the nodes in the graph - the nodes to be activated and the corresponding activation levels can be considered as input of the spreading activation process;
- the activation of the input nodes is propagated through the graph, incrementing the activation to their neighbors; the amount of the increment of the activation depends on the activation of the starting (active) node and the weight the edges - the higher weights propagate better or faster. The specific function used to determine the propagation values can vary significantly, depending on the desired behavior of the network;
- if at the previous cycle of propagation a node has received activation above certain fairing threshold, it becomes a source of activation propagation at the next cycles;
- the propagation cycles continue until specific termination conditions are met; such condition could be that the termination process naturally finished (there are no more nodes to be fired), but also various other strategies are applicable (e.g. based on elapsed time, size of the active portion of the network, etc.).

Our approach here builds on top of DualRDF, which employs spreading activation over RDF graphs. In the optimal scenario, one should be able to associate weights with the arcs of the graph, i.e. with the RDF statements - in DualRDF, we call such extended RDF representation wRDF (from "weighted" RDF). In wRDF, weights are represented as predetermined set of activation levels, i.e. a set of discrete activation values (as opposed to real number between 0 and 1). This choice was made for two reasons - on one hand it resembles the intuition of the so-called fuzzy logics, on the other this choice allowed us to optimize the wRDF representation and lower the computational costs for the spreading activation process. The same approach was taken towards triple weights, so, weights are defined in terms of a predetermined discrete set (directly following the fuzzy logic notion of confidence of a logical statement).

In terms of data model, wRDF have the task to associate weights with statements in a dataset, i.e. one should be able to associate weight with each quadruple  $\langle s, p, o, c \rangle$  (see section 1.1). We implement this by creating an auxiliary triple set for each weight level. The statements are associated with one of those levels in dependence of their weight.

### 4.3 Using Spreading Activation for Selection

The essential idea for implementation of selection mechanism by means of spreading activation on top of RDF graphs is that the dataset subject to reasoning is primed with concepts

related to the reasoning task and the selection context (see section 2.5).

Let us get a simple example where the high-level reasoning task is answering a query about "family holiday at location with spa and ancient historical sites", and the contextual information is that the user is located in Vienna and the query is issued during the summer season.

The most simple and direct approach to provide the input for the spreading activation task would be to syntactically collect all the concepts appearing in the query and the context , e.g. "family holiday", "spa", "ancient history", "Vienna" and "summer" (here we presuppose that there is an easy mechanism to map these keywords and phrases to concepts, modelled as nodes in an RDF graph). Those concepts can be passed for priming, all of them paired with equal initial activation levels. A more comprehensive selection approach could involve deeper analysis of the query and its structure, possibly also some reasoning, to derive logically related concepts to be used as input for priming. Alternatively, one can think of a mechanism to derive different levels of activation for the different concepts in the query.

Another important question is related to the mechanisms for deriving weights for the arcs in the RDF graph. Although priming could be performed in a graph whose arcs have no specific weights, properly determined weights could be expected to allow for much more adequate modelling of relevance and thus, more useful results of the selection phase.

An important consideration to note is that determining the weights of the statements is can be expected to be relatively expensive computational task, so, it cannot be handled at query time - it can be seen as a sort of indexing, a preparatory preprocessing of the data which should take place off-line, outside the pipeline of a specific reasoning task. (Although there are algorithms like HITS, (Kleinberg 1999), which allow query-time graph analysis, those can be used for relevance ranking of the results of a task, but not for selection.) For scenarios, for which such pre-processing of the dataset is not possible, spreading activation will have to be performed over non-weighted RDF graphs.

We plan to investigate two approaches for weighting statements:

- co-occurrence in text - in scenarios where there is a free-text corpus linked (via semantic annotation) with the dataset subject of reasoning, we can investigate the co-occurrence of the subject and the object of the statement in the texts and derive a measure for there level of association on this basis. One should note that this type of analysis cannot be expected to derive the strength of the logical relationship, but this does mean that the derived weight is not useful for the purposes of "spreading relevance". Such co-occurrence analysis has been already successfully used for popularity ranking

of entities in the CORE module of KIM's semantic annotation and search platform <http://www.ontotext.com/kim>.

- statistical graph analysis - algorithms which allows for determining the importance of the links in a graph. Such example is Google's PageRank algorithm, (Brin and Page 1998) which allows determining the importance of nodes (i.e. pages) in the "web graph" - algorithm of this type can be adapted for analysis of RDF graph, than the weights of the statements can be determined as function of the weights of their subjects and predicates. There are also several relevant efforts on relevance ranking against large RDF graph, which should be considered: (Hogan et al. 2006).

The activation is determined othrough RDF priming - the well-known connectionist AI approach for spreading of activation over artificial neural networks, adapted to operate on over RDF graphs. The basic idea is that memory elements, representing the parameters (see the preceeding section), are primed with certain level of "energy", which is than spread to their neighbours in the graph. When the process is over, the "active" part of the graph is the desired selection; its size can be tuned through the activation threshold. This model, will be used also for other reasoning tasks, e.g. activations can be used as a currency for cost-based reasoning.

Spreading activation is a method which is naturally suitable to work under limited/resources because at each phase there are usable results (more energy and time will result in activations of larger portions of the RDF graph/memory). We will experiment with Weighted RDF (wRDF), an extension of the RDF model, which allows weighting of facts according to their contextual relevance. wRDF is inspired by DUAL and combines connectionist models with symbolic reasoning.

## 4.4 Relevance to Other Reasoning Tasks

Also, the idea of using ranking algorithms (PageRank, HITS) over RDF graphs (cf. Onto-Text's Namerini project, also some papers from DERI Galway may be relevant). An issue is that the interpretation of web links as popularity indicators probably doesn't directly translate into RDF (more links doesn't necessarily mean more relevance?). So some other dimension, e.g. activation, might need to be added to the picture?

## 4.5 References

Brin, S; Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine. In Proc. of the seventh international conference on World Wide Web 7, pp. 107-117.

Hogan, A; Harth, A; Decker, S. (2006). ReConRank: A Scalable Ranking Method for Semantic Web Data with Context. Second International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2006), Athens, GA, USA.

Kleinberg, J. (1999). Authoritative sources in a hyperlinked environment Journal of the ACM 46:5:604-632.

Kokinov, B. (1994). The context-sensitive cognitive architecture DUAL. In Proc. of the 6th Annual Conference of the Cognitive Science Society. Hillsdale, NJ: Erlbaum.

## 5 GEOMETRICAL SEMANTIC SPACES

This task will apply new geometrical models of document retrieval to selection of triples. Search engine algorithms like PageRank and HITS exploit the link structure of the web (essentially they mine human decisions about what pages are good to link to for a particular subject). For many information spaces, however, this type of data is not available. Vector space based retrieval has proven an effective substitute where the requisite behavioral data to support ranking algorithms is not present and is in widespread use in systems like Apache Lucene and (at least in early versions) Inktomi-derived systems.

Vector space model (VSM) generates a distributed representation (a vector) for a document (or a query) by using the content words in the document. The similarity between a document and a query is computed by the inner-product of the two vectors corresponding to the document and the query, respectively. The relevancies of documents to a query can be ranked according to their similarities. Furthermore, geometry of vector space has been explored to facilitate information retrieval. For example, Principle Component Analysis (PCA) and Latent Semantic Analysis (LSA) project the vector into a subspace for dimension and noise reduction [11]. Hyperspace Analogy to Language (HAL) uses vector representations of words to measure the similarity between words being close to each other [25, 3] and has been used for query refinement [26].

In recent work (van Rijsbergen, Widdows) vector space models have been extended to incorporate an apparently productive analogy with parts of the mathematical apparatus of quantum theory and the representation and manipulation of word meaning. The work has had promising results: phenomena like attention or lexical priming can be modeled by familiar geometrical operations involving movement in space, and the work has led to a new treatment of negation in vector space retrieval.

Very closely related to VSM in information retrieval, semantic space model uses a vector for a word representation. The representation is based on context of the word, which could be the documents containing the word or the surrounding words. Like in VSM, several very useful transformations, including LSA, random projection and permutation, have been applied in semantic space to extract and represent semantic information and facilitate computation. It is worth noting that, while VSM represents one document solely based on the content of the document, semantic space model often represents a word based on the surrounding words and their relations (such as order) to the current word [11].

The LarKC project deals with RDF triples. The key task of WP2 is to obtain a subset of relevant triples from a huge number of triples for a given query. From a geometrical point-

of-view, we need to represent those triples in a semantic space first, then segment the space into areas in a hierarchical fashion, and finally put a query into the semantic space and select those triples which surround the query in the space. The key question is how to construct a suitable semantic space for a set of triples. The first choice is the vector space – representing a triple as a vector and representing a query as a vector or sub-space in the vector space.

These are the problems we will address in Task 2.5, geometrical semantic spaces, as a candidate approach for the selection and retrieval problems of this work package.

The rest of this chapter reviews current thinking on geometrical models for information retrieval. We review the current works on VSM for information and the semantic spaces in Sections 5.1 and 5.2, respectively. Then we focus on the recent developments on geometric methods for information retrieval, in particular the analogy of quantum mechanics and some application examples. See also the *Anatomy of a selection plugin* section of deliverable D2.1.2 (Apparatus) for how this work may be used in practice to perform selection tasks for LarKC.

## 5.1 Information retrieval

An **Information retrieval (IR)** system finds relevant information from a collection of information objects. IR involves three elements:

- A collection of information objects, which may be documents, web pages, images, videos, audios, or RDF triples. The conventional IR system uses documents. Internet search engines search web pages, mainly based on the text content and/or hyper-links that web pages contain. Searching over multi-media objects such as images, video and audio is a new domain. Selecting the relevant RDF triples is a new task for semantic web applications. Note that the collection may be closed, in the sense that the objects in it do not change, or dynamic, in the sense that new objects may be added.
- An information need from a user. It may be a simple query containing one or more terms, or a more complicated information need described by a paragraph of text, or some typical examples (in this case the user wants to obtain more objects from the collection which are similar to the given examples).
- The search procedure, which finds objects from the collection which are relevant to the information need.

**Vector space models (VSMs)** have been widely used in IR [4]. A VSM represents the objects for retrieval and the information query in the same multi-dimensional Euclidean

space (or vector space). The coordinates of the vector space correspond to the content words (or terms) for text retrieval – one coordinate for each particular content word. The set of content words for one application can be pre-defined [36], or can consist of all the words contained in the documents except the function words and connectives (such as articles and prepositions, which are typically regarded as ‘stopwords’ in IR). Hence the vector space may be of very high dimensionality if there are many content words. For one document or query, the component of its vector representation for one particular coordinate is the number of occurrences of the term corresponding to that coordinate ( $TF$ ), which may be modified by the inverse document frequency ( $IDF$ ) of the term in some way, and the vector may be normalised.

The VSM measures the similarity between object and query by computing the cosine of the angle or inner product between the object vector and query vector. Then it ranks the objects according to the similarities to the query and presents the ranked list to the user.

Note that only part of user’s information need is expressed through the query. A user may give the IR system feedback about which objects are relevant and irrelevant, and the system may use the feedback to refine the query and make a new search. The VSM can handle the relevancy feedback and query refinement by using Rocchio’s method [4], for example. User feedback is one form of the interaction between users and IR systems. In another form, the user changes the information need after obtaining more information by reading some of the retrieved documents.

The changes in both the collection and the information need reflect the dynamic nature of IR. An IR system should address those issues.

The VSM uses the Euclidean space, a concept from geometry. Based on the VSM, Keith van Rijsbergen proposed the **geometry of IR** in [35], which uses the Hilbert space. In fact only the inner product of two vectors is important in the VSM. The mathematical space with inner products is Hilbert space. Therefore the VSM method can be generalised to a Hilbert space.

Hilbert space is more abstract (or general) than Euclidean space. For example, Hilbert space may have a finite or infinite dimension and can use real or complex numbers, while a Euclidean space has only finite dimension and uses real numbers. Hence the Hilbert space may provide more possibilities for improving IR methods. For example, [35] mentioned to use complex numbers instead of real numbers in the vector representation, and also proposed to regard a retrieval object as an abstract object (or element) in Hilbert space and a set of bases in the space as corresponding to one particular point of view to represent the retrieval objects.

Moreover, as Hilbert space is the mathematical foundation for quantum mechanics (QM), basing IR on Hilbert space creates an analogy between IR and QM and may usefully bring some concepts and methods from QM into IR.

Before we discuss the useful analogies between IR and quantum mechanics and the potential applications, we review the works on semantic spaces for representing lexicon in natural language processing, which could be useful for representing the RDF triples in vector space.

## 5.2 Semantic space for lexicon

It is our view that semantic space refers to vector representation for lexicon. It represents the semantics of a word using context (for example, the surrounding words in text or the documents containing the word). Vector representation provides some natural way (e.g. the inner product or the distance in vector space) to compute the similarities among words, which is the basis for clustering words for the application tasks such as automatic thesaurus generation, concept matching, and word monitoring. In cognition science, vector space generates a distributed representation for lexicon, which can be seen as a form of distributional memory. Different types of operations and transformation in vector space, such as superposition, inner production, and convolution, can be employed to study the cognitional phenomenon such as combinations of words, associative recognition, and language comprehension. In the following we discuss some of the well-known methods and models for semantic spaces, which originated either from computational linguistics or from cognitive science.

**Latent Semantic Analysis (LSA)** searches a vector space for a subspace that has coordinates with more semantic meaning in comparison to the original coordinates [11]. For example, in a vector space with coordinates corresponding to content words from a collection of documents, LSA finds out some subspace with the coordinates, each of which corresponds to a set of words that relate to one topic in the document collection. LSA applies singular value decomposition (SVD) on a Word $\times$ Document matrix, which is obtained from a text corpus, to find the subspace. LSA has been successfully used in many tasks such as linguistic feature selection [36], judgments of semantic similarity [21], and discourse comprehension [17]. However, LSA lacks of scalability because applying SVD to a large matrix requires too much of computer's memory and computational time. It is also often criticised for using a bag of words model which ignores the relations among the words in text.

**Random projection** maps vectors to a subspace using a random matrix [16]. It is much more computationally efficient than PCA and LSA, while it is as effective in dimension and noise reduction [1]. The experimental results in [1] showed that a sparse random ma-

trix and the general random matrix give similar results, whereas the former has additional computational savings in random projection. Therefore, random projection scales better in comparison to LSA and indeed can be used as an alternative to LSA for large data such as a large text corpus.

**Hyperspace Analogy to Language (HAL)** begins with a Word $\times$ Word matrix that is populated with frequency counts of word co-occurrences, in a sliding context window of word tokens[25]. The context window is direction sensitive, so that the rows and the columns in the resulting matrix represent co-occurrences to the right and left of each word. Each row-column pair is then concatenated and the least variant elements are discarded. Therefore HAL does encode the order information of words in text implicitly. But it does not have a mechanism to retrieve sequential dependencies. HAL has been used for query refinement [26].

In the conventional word space model, each of the surrounding words or documents corresponds to one coordinate in the space. So the dimension of the vector will increase if more words or documents are used as context. *Random indexing* [32] uses vectors with fixed dimensions to represent the context, by which a surrounding word or a document is assigned with a random vector with a pre-defined dimension. If a word has more than one surrounding words as a context, then the context of the word is the sum of the random vectors of all the surrounding words. In comparison to the vector with variable dimension in the conventional setting, the random indexing uses the vectors with the pre-defined dimension, which has the advantages in the computer implementations, such as scalability and incremental learning. The open source package *Semantic Vectors* at <http://code.google.com/p/semanticvectors/> implemented the random indexing algorithm to create a word space model which can be used for concept matching and other semantics related tasks.

**Bound Encoding of the Aggregate Language Environment (BEAGLE)** uses random indexing to create a computational model that builds a semantic space representation of meaning and word order directly from statistical redundancies in language [15]. It represents words by high-dimensional holographic vectors. It assigns a random environment vector with fixed dimensions to a word when the word is encountered at the first time. The context information for a word is a sum of the environmental vectors for the other words in the sentences containing the word. The order information for a word (one or more words are in front of the word and/or follow the word) is encoded by circular convolutions of the environment vectors of other words with a fixed vector  $\Phi$  representing the current word. Note that the circular convolution results in a vector that has the same dimension as the component vectors. Different order information vectors can be summed together and also with the context information vector. Hence a vector representation for a word in BEAGLE

contains both the context information and sequential dependencies. Moreover, the context and sequential information about a word can be retrieved by using the operations such as projection and inverse circular convolution (i.e. circular correlation).

BEAGLE has been evaluated on several benchmarking tasks including the synonym section of the TOEFL and word priming [15]. In this specific task, BEAGLE performed almost as good as LSA when relying on context information only, while in the case of using both context and order information, BEAGLE was better. On the word priming problem BEAGLE performed better than any other methods. Therefore, we conclude that it is a promising model for semantic space.

Inspired by BEAGLE, [33] used **permutations** for encoding the order information. They also use a sparse random vector as an environmental vector for a word. Despite the fact that the permutation operation is used for order information only, permutations can be used for encoding different types of relations among the words (because there are different permutations and each permutation has unique inverse operation). In contrast, BEAGLE can only encode one type of relation (namely order information) since it uses a pre-defined operation i.e. circular convolution for encoding order information.

Note that BEAGLE uses a dense random vector. In contrast, [34, 33] uses sparse random vector<sup>1</sup>, which requires less space for storing and makes computations more efficient.

### 5.3 Quantum Mechanics

**Quantum mechanics (QM)** is the study of mechanical systems whose scales are close to or below the atomic scale, such as molecules, atoms, electrons, protons and other subatomic particles. QM reveals some important properties of quantum system that are very different from the normal macroscopic mechanical system that the classical Newtonian mechanics studies and we experience with in our daily life. We will discuss some of the properties later on.

QM is based on complex Hilbert space. A quantum system is associated with a complex separable Hilbert space. The possible states of the system are represented by unit vectors in the Hilbert space. An observable (or measurable properties) of the quantum system, such as energy, position, and momentum, corresponds to a self-adjoint linear operator in the Hilbert

---

<sup>1</sup>For example, a random sparse vector has 3000 coordinates, and randomly selects 5 coordinates with the values of 1, 5 other randomly selected coordinates with the values of -1, and all other coordinates with the values of 0.

space<sup>2</sup>. The possible values of the observable are the eigenvalues of the corresponding self-adjoint linear operator<sup>3</sup>. Note that a linear operator may have discrete (and even limited) eigenvalues, meaning that the corresponding observable has only discrete values. During a measurement for one observable, the system collapses from a given initial state to a particular eigenstate corresponding to one eigenvector of the linear operator for the observable. The probability of collapsing to the eigenstate is the square of the absolute value of the inner product between the initial state and the eigenstate. The value of the observable for the measurement is the eigenvalue associated with the eigenvector (eigenstate).

The Hilbert space of a composite system is the Hilbert space *tensor product* of the state spaces associated with the component systems. The state of a composite system may or may not be the tensor product of the states of the component systems. If it is not, it is an entangled state, as explained below.

QM incorporates the following four types of phenomena that classical physics cannot account for.

- the quantization (discretization) of certain physical quantities, in the case that the associated self-adjoint linear operator has discrete eigenvalues.
- wave-particle duality, a concept that all forms of matter and energy exhibit wave-like properties (such as filling in the space and having a certain wavelength) and particle-like ones (such as having a position in space).
- the uncertainty principle, which states that locating a particle in a small region of space makes the velocity of the particle uncertain; and conversely, that measuring the velocity of a particle precisely makes the position uncertain. This principle can be explained by the wave-particle duality. In fact the uncertainty principle holds for any two observables if the associated self-adjoint operators  $A$  and  $B$  are non-commutative, namely  $AB \neq BA$ .
- quantum entanglement, a phenomenon in which the quantum states of two or more objects are linked together — even though the individual objects are spatially separated. This interconnection leads to correlations between observable physical properties of remote systems.

---

<sup>2</sup>It is worth noting that QM asserts only the existence of the correspondence between a quantum system and a Hilbert space, but does not provide a general solution for finding out the corresponding Hilbert space for any quantum system. Given a quantum system and some observable, you have to work out the particular Hilbert space corresponding to the quantum system and the particular self-adjoint linear operator for the observable.

<sup>3</sup>It has been proved that all the eigenvalues of a self-adjoint linear operator in complex Hilbert space are real numbers, which guarantees that the values of an observable are all real numbers.

Table 5.1: Analogy between QM and IR.

QM	IR
a quantum system	a collection of object for retrieval
complex Hilbert space	information space
state vector	objects in collection
observable	query
measurement	search
eigenvalues	relevant or not for one object
probability of getting one eigenvalue	relevance degree of object to query

Some of the QM phenomena have been found in IR and other information processing tasks. Before we describe those phenomena, we will discuss the analogy between QM and IR.

## 5.4 Analogy between QM and IR

Table 5.1 presents correspondences between the components of QM and IR. QM uses a complex Hilbert space to represent a particular quantum system, while IR needs some information space to represent the objects in one collection for retrieval. The state vector is the complete and maximal summary of the characteristics of the quantum system at a moment in time, which changes at different time. Objects in collection contain all the information available for retrieval. We may make an analogy of one object in collection as a state vector of the quantum system at one particular time (or the quantum system at one particular state). Observables are the physical quantities that one can measure on quantum system. Different quantum systems may have different types of observables. Query represents a question for which a user may obtain answer from one collection. Different collections have different types of sensible questions to ask. Measurement is a procedure to determine the value of an observable for one quantum system in one particular state, while search is a procedure to determine the relevance of an object in collection to a query.

As the VSM is a common method for IR, Table 5.2 compares the VSM with QM, based on the analogy between IR and QM discussed above. We can see that the QM use more general model than the VSM, which may inspire new methods beyond the VSM. For example, complex Hilbert space is more general and expressive than the Euclidean space. We may use complex numbers to represent quantities in IR. Hilbert space will enable us to represent the retrieval objects as abstract objects and to use any base and base transformation. In particular one may construct some base according to the query and use the base to represent each object from query's point of view. One may also represent the query using some subspace or operator in Hilbert space.

Table 5.2: Comparison of VSM with QM.

QM	VSM for IR
complex Hilbert space	real Euclidean space
state vector	vector
self-adjoint linear operator for observable	query vector
measurement – interaction between measurement device and quantum system	search – may involve interaction between user and system.
eigenvalues of the operator	relevant or not for one object
probability of obtain one eigenvalue	relevance degree of object to query

## 5.5 QM Phenomena in IR

Since the QM is based on Hilbert space, and the Euclidean space used in the VSM of IR and natural language processing (NLP) is a specific type of Hilbert space, it is not surprising that there are QM phenomena in IR and NLP.

[28] used a recommender system as example to demonstrate some QM phenomena in data analysis. As we know, such a system recommends the similar new objects (e.g. films, books) to the user based on his previous experience.

- In the VSM, the similarity of a new object to the previously seen objects is computed based on the correlation matrix of the vectors of the previously seen examples. The correlation matrix is a self-adjoint linear operator in the vector space, corresponding to an observable in QM.
- [28] obtained a Bell’s inequality of similarities and found some examples violating the inequality. Violation of Bell’s inequality means that one cannot predict the future using the user’s profile derived from the past in the framework adopted. It can be viewed as an indicator of a quantum statistical correlation.
- [28] also pointed out that the concept lattice based on the taxonomy of concepts is not distributive, namely,

$$x \wedge (y \vee z) \neq (x \wedge y) \vee (x \wedge z)$$

For example, in a taxonomy of animals, take  $x$  = “bird”,  $y$  = “human” and  $z$  = “lizard”. Then both  $x \wedge y$  and  $x \wedge z$  are empty, so that  $(x \wedge y) \vee (x \wedge z)$  remains empty. On the other hand,  $y \vee z$  = “vertebrates”, because vertebrates are the smallest class including both humans and lizards. Hence  $x \wedge (y \vee z)$  = “bird”, which is not empty. The non-distributive property does not hold in the classic set based logic but occurs in quantum logic (which will be discussed in the following section).

[27] discussed several psychological experiments for human’s word association and described some interesting findings from the experiments. In one experiment, giving a cue word to a human, they measured the likelihood that she recall one related word called target word. Note that one word may have several associative words. For example, given the word “planet”, a human may recall the related words such as “earth”, “moon” and “universe”. Figure 5.1 shows hypothetically a cue word, a target word and two associative words with the pre-existing links. There are three links from cue word to the target word and two associates, one associate-to-associate link, and one associate-to-target link from the Associate 2 to target. The values on the links indicates relative link strengths. Besides the links between the cue word and target word, the associate-to-associate links are also useful for the recall of target word from cue word. One explanation for why associate-to-associate links benefit recall uses the spreading activation equations

$$R(T) = S_{ct} + \sum_{i=1}^n S_{ci}S_{it} + \sum_{i=1}^n \sum_{j=1}^n S_{ci}S_{ij}S_{jt} = 0.8 + 0.1 * 0.7 + 0.2 * 0.6 * 0.7 = 0.954$$

which is based on the classic idea that activation spreads through a fixed associative network, weakening with conceptual distance by multiplying link strengths. Another explanation is based on the assumption that each link in the associative network contribute additively to the recall strength. It uses the activation at a distance equation to compute the recall likelihood

$$R(T) = S_{ct} + \sum_{i=1}^n S_{ci} + \sum_{i=1}^n S_{it} + \sum_{i=1}^n \sum_{j=1}^n S_{ij} = 0.8 + 0.2 + 0.1 + 0.7 + 0.6 = 2.4$$

It was shown in [27] (in which the target word was the same as the cue word and  $S_{ct} = 0.0$ ) that the psychological experimental results were much more consistent with the second explanation than the first one. The authors explained the additive effects of associate-to-associate and associate-to-target links on the recall level by the entanglement of the target and its associates. [27] speculated that there were entanglements between the target and its associates and because of the entanglements, the target word and the associated words acted as correlated instead of separated entities.

## 5.6 Quantum Logic and its application in document retrieval

Quantum logic is defined on the sub-spaces of Hilbert space (see [39], for example). It is also called vector logic. Given a Hilbert space  $\mathcal{H}$ , the logical connectives of the quantum logic

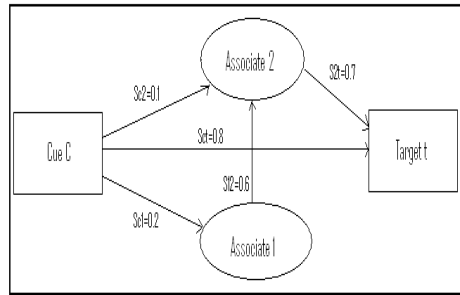


Figure 5.1: A hypothetical cue and target with two associates, and cue-to-target link, cue-to-associate links, associate-to-associate link, and associate-to-target links with link strengths (modified from [27]).

defined in the sub-spaces of  $\mathcal{H}$  are

- The negation of a sub-space  $A$  is the orthogonal sub-space of  $A$ .
- The conjunction of the two sub-spaces  $A$  and  $B$  is the sub-space consisting of the vectors belonging to both  $A$  and  $B$ .
- The disjunction of the two sub-spaces  $A$  and  $B$  is the sub-space consisting of the vectors each of which is a linear combination of two vectors respectively from  $A$  and  $B$ .

In comparison with classic set logic (namely considering one sub-space as a sub-set of the vector set  $\mathcal{H}$ ), the conjunction is the same but the negation and disjunction are different in the two logics. In set logic the negation of one sub-set contains all the elements not being in the sub-set, while in quantum logic the negation of one sub-space contains only the vectors that are orthogonal to the sub-space. The conjunction of the two sub-sets in set logic is the union of the elements in each of the two sub-sets, while the conjunction of the two sub-spaces in quantum logic contains not only all the vectors of the two sub-spaces but also those vectors between the two sub-spaces. In other word, the quantum logic utilizes the structure of Hilbert space, whereas the set logic just uses the set relationships among the sub-sets of Hilbert space.

Quantum logic can be used in the VSM based methods. VSM has been widely used in IR and NLP applications. [36] applied the orthogonal negation in vector space for modelling word-meaning and document retrieval and compared the results with those using set logic. For document retrieval the experiments were done on the British National Corpus, one news article corpus, and one medical document corpus.

To obtain vector representation for document and query, 1000 content words were selected. Each document was represented as a 1000-dimensional vector — each coordinate corresponded to one content word and the tf-idf weighting scheme was used for the value of each

coordinate. A query had the form  $a \text{ NOT } b$ , where  $a$  and  $b$  are two terms. Each term was also represented as a 1000-dimensional vector by counting the occurrences of the selected content words in a 15 words context window centred around the term in document and using the tf-idf weighting scheme. Each vector was normalised. Suppose that the terms  $a$  and  $b$  have the vectors  $v_a$  and  $v_b$  respectively, according to the orthogonal negation, the query  $a \text{ NOT } b$  has the vector representation

$$v_q = v_a - \langle v_a, v_b \rangle v_b$$

Then the document retrieval was performed by computing the similarity between the query vector and document vectors and ranking documents according to the similarity.

The author also implemented the document retrieval using the negation of set logic. It first retrieved the documents by using the term vector  $v_a$  and then removing the document containing the unwanted term  $b$ .

Experimental results showed that, in comparison to the set logic negation, the vector negation method retrieved the documents containing a much lower number of negative synonyms and negative neighbours but, unfortunately, also a lower number of positive terms. The vector negation method produced promising results.

## 5.7 Tensor product and its application in combination of meaning of individual words

QM represents composite quantum system using the tensor product of the Hilbert spaces of individual quantum systems. The tensor product of two objects consists of the combinations of any two components of the two objects, respectively. For example, given two matrices  $\mathbf{A}$  and  $\mathbf{B}$

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$$

the tensor product of  $\mathbf{A}$  and  $\mathbf{B}$  is

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{12} \\ a_{11}b_{21} & a_{11}b_{22} & a_{12}b_{21} & a_{12}b_{22} \\ a_{21}b_{11} & a_{21}b_{12} & a_{22}b_{11} & a_{22}b_{12} \\ a_{21}b_{21} & a_{21}b_{22} & a_{22}b_{21} & a_{22}b_{22} \end{bmatrix}$$

Table 5.3: Verb noun similarities for several word pairs using three different vector composition methods.

First Pair	Second Pair	Vector Sum	Direct Sum	Tensor Product
earn money	pay wages	0.66	0.41	0.16
earn money	wages pay	0.66	0.51	0.26
eat apple	eat tomato	0.73	0.65	0.29
eat apple	throw apple	0.57	0.52	0.04

Similarly, the tensor product of two vectors is a matrix (or a long vector).

In the VSM methods for IR and NLP application, we often need the composite of two or more elements. In Section 5.2 we discussed several semantic space models such as BEAGLE and random indexing and permutations to encode the order information and other relations among words. [6] investigated the composite meaning of a sentence using the semantic representations of individual words. The method used was the tensor product of the semantic vectors of individual words. For example, given a sentence “John likes Mary”, and the vector representation of the semantics of the three words,  $v_{John}$ ,  $v_{likes}$ , and  $v_{Mary}$ , the sentence was represented as tensor product

$$v_{John} \otimes_{sv} v_{likes} \otimes_{vo} v_{Mary}$$

where  $\otimes_{sv}$  and  $\otimes_{vo}$  represent tensor products for subject and verb and for verb and object, respectively. Using the tensor product operations on semantic vectors, one sentence is represented as a matrix. Then we can compute the similarity of two sentences by using the inner product of the two matrices.

[37] used the vector composition to model semantic composition. In one experiment the author computed the similarity between verb-noun pairs using vector composition methods. It is expected that *eating a tomato* is relatively similar to *eating an apple*, whereas *throwing an apple* is quite different. This experiment first represents each word as a semantic vector using the 1000 content-bearing words, as described in Section 5.6. Then it computed a vector for a pair of words using three different methods. Given the two vector  $a$  and  $b$  with the same dimension  $n$ , the vector addition operation on them resulted in a vector of the dimension  $n$ ,  $a+b$ . The direct sum of the two vectors was the concatenation of the two vectors, a vector of the dimension  $2n$ . The tensor product of the two vectors was a vector of dimension  $n*n$ . As one pair of words was represented by a vector, the similarity of two pairs can be computed as the inner product of two corresponding vectors.

Table 5.3 shows some results of the comparison of the similarities of noun verb pairs. We can see that the tensor product numbers gave the best discrimination.

## 5.8 Conclusion and future work

Vector space model in IR uses distributional representations and geometric measures for comparing query and objects. Semantic space is also based on vector space and can encode context and order information for words in text. Geometric properties and transformations in vector space can be explored to compute similarity and to encode and retrieve context and order information. For example, LSA and random projections are effective for dimension reduction and feature selection. Random indexing and permutations provides effective ways for information encoding.

Vector space model has been used in text at two different levels, namely document and word, and in different fashions. Vector representation for document is based on the content of document. In contrast, the vector representation for word is based on the relations of the word to others. An RDF triple can be seen at an intermediate level between word and document – it both contains words and has rich and meaningful relations with some other RDF triples. Therefore, vector representation of an RDF triple should consider both the internal words of the triple and the relations with other triples. In another words, a combination of VSM in IR and semantic space for lexicon is needed to apply the geometric methods for RDF triples retrieval.

The geometry in Hilbert space provides a general framework for IR. Using the framework, IR can be studied from a broader view. There are many similarities between IR and QM. They both are based on Hilbert space and involve the interaction between system and user (or observer). From physics' point of view, QM studies the phenomena at sub-atomic level, which are very different from those physical phenomena at a human perception level. However, QM phenomena, such as entanglement, the uncertainty principle and state collapse, exist in information-rich areas such as information processing, biology, geosciences and economics [38]. The concepts and methods in QM may inspire new concepts and methods in those areas. For example, as discussed earlier, quantum logic and the tensor product operation have been used in IR and NLP and obtained promising results.

On the other hand, the quantum system is simpler than IR and natural language understanding because it involves a small number of types of elements and a few operations. In order to model the complicated phenomena in IR and natural language, one may have to employ more complicated mathematical tools. Differential geometry studies the objects with a more complicated structure than Hilbert space. In particular, it does not presuppose the existence of a global set of coordinates or attributes, is not constrained to being flat or linear, and pays much attention to the local context. Hence it may be more suitable for modeling the meaning of text. For example, instead of specifying some attributes as global

coordinates to represent all queries and retrieval objects, the information space for IR (or semantic space for NLP) may be segmented into different areas in a hierarchical fashion to represent different contexts or semantic types in natural language, and each context area is characterised by the attributes and structure relevant to the context.

## 5.9 References

These appear in the bibliography below.

## 6 SELECTION METHODS FROM DATA STREAM MANAGEMENT SYSTEMS

### 6.1 Selection in C-SPARQL

Streaming data in the LarKC framework will be queried in C-SPARQL, an extension of SPARQL for dealing with streaming RDF data (see Appendix 1). C-SPARQL is at a very early stage of design, yet is already provided with aggregation operators (lacking in SPARQL) as well as a continuous semantics. An extensive definition of the language is included in [1], which constitutes a contribution that can hardly be considered only relevant for one WP, as the definition of a language inevitably addresses many issues, all dependent one of another. The specification addresses the syntax and semantics of the language, and provides an exemplification of use for urban computing, as well as initial hints to how query execution can be optimized and parallelized.

Here, we give a sketch of the syntax of C-SPARQL and discuss how selection has been included in its scope, thus offering a specific reading key to [1] in the context of WP2.

Selection and retrieval in the context of data streams have several facets. One aspect concerns the identification, within multiple candidate data streams, of the most relevant one to a given LarKC application. This aspect is exemplified by a urban context where multiple sources of streaming information are available and we need to select the ones that are most useful to a given decision process, such as determining the amount of traffic in a urban area or the expected delay for going from a location to another.

At this stage, we assume that it will be always possible to split conditions into either

- a logical properties expressed within a C-SPARQL query (e.g., given a network of traffic sensors, each generating a stream, restricting to those nearest to a specific location, or those explicitly accounting traffic due to trucks), or
- b in terms of semantic properties (e.g. all streams dealing in their descriptors with concepts connected to "the airport").

The latter is a general retrieval problem, regardless of the fact that sources are streams, while the former is a problem that could be logically considered as part of the query language. We then focus only on the former.

Another aspect concerns the selection, from within a data stream, of a limited amount of information so as to be able to deal with massive stream inputs while considering in the

application only a limited amount of data, extracted according to a well-defined selection conditions. Here again we approach the problem by recognizing three distinct patterns.

1. The first and most trivial one is of course the specification of selection criteria by means of plain SPARQL-like patterns, to be matched against the streaming data (in C-SPARQL). As we are mostly dealing with streaming data, another basic means of selection is a rich language for the specification of windows over the stream, specified in terms of time duration, number of data items, sliding criteria, etc.
2. Another pattern deals with applying known aggregation functions (e.g. counting) to a stream for selecting items based on their aggregate properties or synthesizing the information they carry. The example is the monitoring of street traffic and the use of counting functions to identify as meaningful the data concerning those streets in which the number of vehicles (or vehicles of a certain kind) is neither too low nor too high; such query could either return all involved vehicle ids or return whether a street has light traffic, heavy traffic, is congested, or blocked. Moreover, this can be expressed as one or more continuous queries, each with an input stream consisting of a large amount of data (possibly but not necessarily expressed as triples) and constructing as output either RDF triples or RDF molecules. We expect this information to be possibly further queried at a higher level of abstraction or to be directly passed to specific reasoners. These issues are covered by the current status of our C-SPARQL proposed specification, as the language includes provisions for defining windows over streaming input data, intervals of publication of streaming output data, and all the required aggregation operations. The solution is both elegant and suggestive of powerful optimization and parallelization methods for the underlying query engine.
3. The third pattern deals with the use of statistical sampling to the input in order to select elements from it (e.g., randomly). We have not yet considered this aspect, however we do expect it to be easily integrated in the context of C-SPARQL by defining few well-understood sampling methods, applicable to each stream individually before considering stream elements into the query. We expect this part to be easily partitioned, and to be optimized in an orthogonal way w.r.t. the rest of the query.

All such patterns are covered by C-SPARQL (declaratively) and be the subject of optimization and parallelization methods that will make stream-based selection composable with the other phases of computations in an RDF-compatible context. We now sketch the features of the language that cover these patterns.

## 6.2 A sketch of the new features introduced by C-SPARQL

We briefly present the main extensions of C-SPARQL w.r.t. SPARQL. We first characterize RDF streams as a new data type, then introduce the syntax for identifying streams and for defining windows over streams, then introduce aggregation as an orthogonal extension, then discuss a function used for retrieving timestamp information. The syntax of each new feature is given in terms of additional productions to be added to the standard SPARQL grammar [4].

### 6.2.1 RDF Stream Data Type

C-SPARQL adds *RDF streams* to the data types supported by SPARQL. An RDF stream is defined as an ordered sequence of pairs, where each pair is constituted by an RDF triple and its timestamp  $\tau$ :

$$\begin{array}{c}
 \dots \\
 (\langle subj_i, pred_i, obj_i \rangle, \tau_i) \\
 (\langle subj_{i+1}, pred_{i+1}, obj_{i+1} \rangle, \tau_{i+1}) \\
 \dots
 \end{array}$$

Timestamps can be considered as *annotations* of RDF triples; they are monotonically non-decreasing in the stream ( $\tau_i \leq \tau_{i+1}$ ). They are not strictly increasing because timestamps are not required to be unique. Any (unbounded, though finite) number of consecutive triples can have the same timestamp, meaning that they “occur” at the same time, although sequenced in the stream according to some positional order.

### 6.2.2 Windows

The introduction of RDF streams as a new type of input data requires the ability to *identify* such data sources and to specify *selection* criteria over them.

As for *identification*, we assume that each data stream is associated with a distinct IRI, that is a locator of the actual data source of the stream. More specifically, the IRI represents an IP address and a port for accessing streaming data. In a typical C-SPARQL application we expect that several streaming data sources can be available on Internet, modeled as RDF data streams (possibly after data transcoding); RSS feeds or other publish-subscribe mechanisms allow to be seamlessly captured by the same abstraction.

As for *selection*, given that streams are intrinsically infinite, we introduce the notion of windows upon streams, whose types and characteristics are inspired by those of the windows in continuous query languages such as CQL [2].

Identification and windowing are expressed in C-SPARQL by means of the FROM STREAM clause:

<i>FromStrClause</i>	→ ‘FROM’ [‘NAMED’] ‘STREAM’ <i>StreamIRI</i> ‘[ RANGE’ <i>Window</i> ‘]’
<i>Window</i>	→ <i>LogicalWindow</i>   <i>PhysicalWindow</i>
<i>LogicalWindow</i>	→ <i>Number</i> <i>TimeUnit</i> <i>WindowOverlap</i>
<i>TimeUnit</i>	→ ‘MSEC’   ‘SEC’   ‘MIN’   ‘HOUR’   ‘DAY’
<i>WindowOverlap</i>	→ ‘STEP’ <i>Number</i> <i>TimeUnit</i>   ‘TUMBLING’
<i>PhysicalWindow</i>	→ ‘TRIPLES’ <i>Number</i>

A window extracts from the stream the *last* data stream elements, which are considered by the query. Such extraction can be *physical* (a given number of triples) or *logical* (a variable number of triples which occur during a given time interval).

Logical windows are *sliding* [3] when they are progressively advanced of a given `STEP` (i.e. a time interval that is shorter than the window’s time interval); they are *non-overlapping* (or `TUMBLING`) when they are advanced of exactly their time interval at each iteration. With tumbling windows every triple of the data stream is included exactly into one window, whereas with sliding windows some triples can be included into several windows.

The optional `NAMED` keyword works exactly like when applied to the standard SPARQL `FROM` clause for tracking the provenance of triples. It binds the IRI of a stream to a variable which is later accessible through the `GRAPH` clause.

### 6.2.3 Query Registration

Continuous queries in C-SPARQL are queries over RDF data streams (i.e. queries including at least one `FROM STREAM` clause).

A registered C-SPARQL query produces continuous output in the form of variable bindings tables or graphs. Each C-SPARQL query is registered through the following statement:

<i>Registration</i>	→ ‘REGISTER QUERY’ <i>QueryName</i> [‘COMPUTED EVERY’ <i>Number</i> <i>TimeUnit</i> ] ‘AS’ <i>Query</i>
---------------------	--



- The first is a new variable (i.e. a variable not occurring in the `WHERE` clause or in other aggregation clauses).
- The second is an aggregation function (one of: `COUNT`, `MAX`, `MIN`, `SUM`, `AVG`); `COUNT` may have no argument, while the other functions take one of the variables occurring in the `WHERE` clause as argument.
- The third is a set of one or more variables, occurring in the `WHERE` clause, which express the grouping criteria.

Every clause may also have an optional fourth part, a `FILTER` clause. The semantics of a query with aggregate functions consists in adding to the regular variable bindings computed by the `WHERE` clause some new bindings, one for each of the new variables introduced by the `AGGREGATE` clauses. The solution constructed in this way may be further filtered by a standard `FILTER` clause, which may refer to all the variables introduced in the `WHERE` and `AGGREGATE` clauses. The evaluations of aggregate functions are all independent from each other and take place after the computation of the bindings provided by the `WHERE` clause. We deliberately constrain C-SPARQL aggregates to use only the variables in the `WHERE` clause - and not other variables bound by other `AGGREGATE` clauses - in order to achieve their independence.

### 6.2.5 Timestamp Function

The timestamp of a stream element can be retrieved and bound to a variable using a timestamp function. The timestamp function has two arguments.

- The first is the name of a variable, introduced in the `WHERE` clause and bound by pattern matching with an RDF triple of that stream.
- The second (optional) is the URI of a stream, that can be obtained through SPARQL `GRAPH` clause.

The function returns the timestamp of the RDF stream element producing the binding. If the variable is not bound, the function is undefined, and any comparison involving its evaluation has a non-determined behavior. If the variable gets bound multiple times, the function returns the most recent timestamp value relative to the query evaluation time.

## 6.3 Execution of C-SPARQL Queries

A crucial issue is how existing technologies can be leveraged to execute C-SPARQL queries, so as to rapidly obtain a proof of concept of the feasibility of our approach without building a new engine from scratch.

A general approach to query evaluation can be based upon the mapping of C-SPARQL to queries upon RDF repositories, data streams, and relational table bindings separately. C-SPARQL queries assume the existence of RDF repositories and RDF data streams. The former are static, while the latter are included within time-varying windows. At a given query execution, however, every window is just a collection of RDF tuples associated with an IRI, hence the `WHERE` clause performs as in standard SPARQL, producing the `WHERE` binding table. If `timestamp` functions are present in the query, new columns need to be added to the binding table to include the timestamp of variables whose lineage is from RDF streams, extracted from the stream annotations. Then, the `WHERE` binding table can be filtered.

Putting streams into RDF format is essential for interoperability, but streaming data can occur in a variety of formats. Therefore, a generic architecture needs completed with a Stream Manager and a Stream Transcoder. The former extracts data streams from sources (e.g. sensors), the latter puts them into RDF format. Thus, executing C-SPARQL requires the computational power of a SPARQL engine, a relational engine, and pairs of stream managers and transcoders at the streaming sources.

Several issues need to be considered w.r.t. C-SPARQL query processing. Among them, two are relevant to retrieval and selection of data sources.

- Another issue concerns the ability to adapt query computations under heavy query loads dynamically, e.g. by sampling input streams or by relaxing the window refreshment. Queries could be registered together with confidence limits so as to help the system's tuning. Sampling functions could be made available as part of the query language, thus enabling an explicit declaration within queries.
- Another issue concerns the distribution of query execution nodes close to data stream sources. The idea is that each source could provide input stream of arbitrary nature and then specific nodes could perform the transformation of such streams into summarized RDF triples and then transmit them to higher-level nodes for more complex computations. Nodes local to streams could support only stream-specific operations. Optimization could reuse classic results of distributed database systems.

## 6.4 References

- [1] V. Tresp, Y. Huang, E. Della Valle, D. Braga, D. Barbieri and S. Ceri. D3.1 Survey of relevant literature on abstraction and learning. LarKC Technical Report. Available on line [http://www.larkc.eu/wp-content/uploads/2008/10/larkc\\_d31\\_survey-of-relevant-literature-on-abstraction-and-learning.pdf](http://www.larkc.eu/wp-content/uploads/2008/10/larkc_d31_survey-of-relevant-literature-on-abstraction-and-learning.pdf)
- [2] A. Arasu, S. Babu, and J. Widom. The CQL Continuous Query Language: Semantic Foundations and Query Execution. *The VLDB Journal*, 15(2):121-142, 2006.
- [3] L. Golab and M. T. Ozsü. Processing Sliding Window Multi-Joins in Continuous Queries over Data Streams. In *Proc. Intl. Conf. on Very Large Data Bases (VLDB 2006)*, pages 500-511, 2003.
- [4] E. Prud'hommeaux and A. Seaborne. SPARQL Query Language for RDF Grammar. Available on line <http://www.w3.org/TR/rdf-sparql-query/#sparqlGrammar>.

## 7 APPENDIX 1: FIS PAPER ABOUT STREAMS IN LARKC

# A First Step Towards Stream Reasoning

Emanuele Della Valle, Stefano Ceri, Davide F. Barbieri, Daniele Braga and  
Alessandro Campi

Dip. di Elettronica e Informazione, Politecnico di Milano, Milano, Italy  
email: {name.surname}@polimi.it

**Abstract.** While reasoners are year after year scaling up in the classical, time invariant domain of ontological knowledge, reasoning upon rapidly changing information has been neglected or forgotten. On the contrary, processing of data streams has been largely investigated and specialized Stream Database Management Systems exist. In this paper, by coupling reasoners with powerful, reactive, throughput-efficient stream management systems, we introduce the concept of Stream Reasoning. We expect future realization of such concept to have high impact on the future Internet because it enables reasoning in real time, at a throughput and with a reactivity not obtained in previous works.

**Keywords:** Data Streams, Reasoning, Real-time, Throughput-efficiency, Urban Computing, Pervasive Computing

## 1 Introduction and Motivation

Semantics is more and more evoked as a powerful tool to facilitate interoperability, flexibility and adaptability. The growing scalability of reasoning techniques [1] is key to the relevant role that semantics will play in the future Internet. While reasoners scale up in the classical, static domain of ontological knowledge, reasoning upon rapidly changing information has been neglected or forgotten.

Data streams are unbounded sequences of time-varying data elements; they occur in a variety of modern applications, such as network monitoring, traffic engineering, sensor networks, RFID tags applications, telecom call records, financial applications, Web logs, click-streams, etc. Processing of data streams has been largely investigated in the last decade [2], specialized Data Stream Management Systems (DSMSs) have been developed, and features of DSMSs are becoming supported by major database products, such as Oracle and DB2.

The combination of reasoning techniques with data streams gives rise to **Stream Reasoning**, an unexplored, yet high impact, research area. To understand the potential impact of Stream Reasoning, we can consider the emblematic case of Urban Computing [3–6] (i.e., the application of pervasive computing to urban environments). The very nature of Urban Computing can be explained by means of data streams, representing real objects that are monitored at given locations: cars, trains, crowds, ambulances, parking spaces, and so on. Reasoning

2 E. Della Valle, S. Ceri, D. Barbieri, D. Braga and A. Campi

about such streams can be very effective in reducing costs: for instance, looking for parking lots in large cities may cost up to 40% of the daily fuel consumption. Problems dramatically increase when big events, involving lots of people, take place; a typical Urban Computing problem is to help citizens willing to participate to such events in finding a parking lot and reaching the event locations in time, while globally limiting the occurrences of traffic congestions.

Some years ago, due to the lack of data, solving Urban Computing problems looked like a Sci-Fi idea. Nowadays, a large amount of the required information can be made available on the Internet at almost no cost: computerized systems contain maps with the commercial activities and meeting places (e.g., Google Earth), events scheduled in the city and their locations, positions and speed information of public transportation vehicles and of mobile phone users [5], parking availabilities in specific parking areas, and so on. However, current technologies are not up to the challenge of solving Urban Computing problems: this requires combining a huge amount of static knowledge about the city (i.e., urbanistic, social and cultural knowledge) with an even larger set of data streams (originating in real time from heterogeneous and noisy data sources) and reasoning above the resulting time-varying knowledge.

A new generation of reasoners is clearly needed in order to simultaneously instruct the car GPS of numerous citizens with the fastest route to the most convenient parking lot during exceptional events. Time constraints for such a reasoner are very demanding (i.e., few ms per query) because citizens are continuously making driving decisions and the traffic keeps evolving; therefore, continuous inference is required. In this work, we define Stream Reasoning as a new paradigm, based upon the state of the art in DSMS and reasoning, in order to enable such applications. By coupling reasoners with powerful, reactive, throughput-efficient stream management systems, we expect to enable reasoning in real time, at a throughput and with a reactivity not obtained in previous works.

In the rest of the paper, we identify the problem we want to untangle (Section 2). We introduce a Conceptual Architecture for Stream Reasoning (Section 3) that instantiates the pluggable algorithmic framework proposed in the LarKC project [7, 8]. We present two stream reasoning frameworks based on such architecture, one representing an evolutionary approach that combines existing solutions (Section 4), the other representing a revolutionary approach that proposes a new reasoning paradigm (Section 5). We conclude the paper discussing the challenges we are facing while planning our future work (Section 6).

## 2 Problem Definition

Our attempt to combine data stream and reasoning technologies starts from terminology. Database (DB) and Knowledge Engineering (KE) communities often use different terms to indicate the same concepts. DB community distinguishes among schema and data, whereas KE community distinguishes among factual, terminological, and nomological knowledge. The notion of data is close to the notion of factual knowledge, and similarly the notion of schema is close to the

notion of terminological knowledge. Nomological knowledge is information about rules defining actions and action-types and governing means-ends relationships in a given culture or society (e.g., when it rains, traffic gets slower); this notion is somehow captured by constraint languages for DBs, but it is mainly peculiar of KE. For the purpose of this paper, we name “*knowledge*” both terminological and nomological knowledge (thus we include in term knowledge the DB notion of schema) and we use “*data*” as a synonymous of factual knowledge.

Knowledge and data can change over time. For instance, in Urban Computing, names of streets, landmarks, kinds of events, etc. change very slowly, whereas the number of cars that go through a traffic detector in five minutes changes very fast. In order to classify knowledge and data according to the frequency of their changes we first need to introduce the notion of “*observation period*”, defined as the period when we the system is subject to querying. In the context of this paper, we consider knowledge as **invariable during the observation period**; only data can change. Of course, knowledge is subject to change, but then the mutating part of the system is not object of observation. This is not surprising: in the DB context, change of schemas occur by means of create or alter table command; while, for instance, the alter table command is executed all query processing relative to that table is suspended.

Examples of invariable knowledge, in the case of Urban Computing, include obvious terminological knowledge (such as an address is made up by a street name, a civic number, a city name, and a ZIP code), which defines the *conceptual schema* of the application, and less obvious nomological knowledge that describes how the world is expected to be (e.g., given traffic lights are switched off or certain streets are closed during the night) or to evolve (e.g., traffic jams appears more often when it rains or when important sport events take place).

Data can be further classified according with the frequency they are expected to change.

1. **Invariable** data: data that do not change in the observation period, e.g. the names and lengths of the roads.
2. **Periodically changing** data, for which a temporal law describing their evolution is present in the invariable knowledge. We can distinguish:
  - (a) *Probabilistic* data, e.g. the fact that a traffic jam is present in the west side of Milan due to bad weather or due to a soccer match is taking place in San Siro stadium;
  - (b) *Pure periodic* data, e.g. the fact that every night at 10pm Milan west-side overpass road closes.
3. **Event driven changing** data that got updated as a consequence of some external event not described in the knowledge, which are further characterized by the mean time between changes:
  - (a) *Fast*, as an example consider the intensity of traffic (as monitored by sensors) for each street in a city;
  - (b) *Medium*, as an example consider roads closed for accidents or congestion due to traffic;
  - (c) *Slow*, as an example consider roads closed for scheduled works.

4 E. Della Valle, S. Ceri, D. Barbieri, D. Braga and A. Campi

Traditional databases are suitable for capturing relatively small quantity of knowledge in their schema and huge dataset of both invariable data and event driven changing data whose mean time between changes is slow or medium. Periodically changing data can be modeled by means of triggers that perform updates; for example a trigger may update the state of a traffic light when it gets switch off for night-time.

Current reasoners are suitable for capturing large and complex knowledge, but at the cost of small datasets. Complex form of periodically changing data can be modeled by means of rules. However, reasoners cannot capture event-driven changing data whose mean time between changes is fast.

If we consider dynamic query generation, we observe that reasoners are best equipped to execute in reaction to the user's invocation, while many modern applications such as urban computing (but also network monitoring, financial analysis, sensor networks, etc.) require long-running, or continuous, queries or reasoning tasks.

Stream Database Management Systems (DSMS) represent a paradigm change in the database world because they move from persistent relations to transient streams, with the innovative assumption that streams can be *consumed* on the flight (rather than stored forever) and from user-invoked queries to *continuous queries*, i.e., queries which are persistently monitoring streams and are able to produce their answers even in the absence of invocation. DSMSs can support parallel query answering over data originating in real time and can cope with burst of data by adapting their behavior and gracefully degrading answer accuracy by introducing higher approximations.

Is combining data stream and reasoning possible? Can the innovation so far confined within the DB community be leveraged in realizing a new generation of reasoners able to cope with continuous reasoning tasks?

### 3 A Conceptual Architecture for Stream Reasoning

We are developing the Stream Reasoning vision with the LarKC European Research Project<sup>1</sup>. LarKC proposes [7, 8] a pluggable algorithmic framework which will be implemented on a distributed computational platform. The pluggable algorithmic framework ideally includes five steps to be iterated until a good enough answer [9] is found:

1. *retrieve* relevant resource/content/context;
2. *abstract* by extracting information, calculating statistics and transforming to logic,
3. *select* relevant problems/methods/data,
4. *reason* upon the aggregated knowledge, and
5. *decide* if a new iteration is needed.

In Figure 1 we present our vision in plugging data stream technologies in the LarKC framework. The top part of the figure represents the problem space

<sup>1</sup> <http://www.larkc.eu>

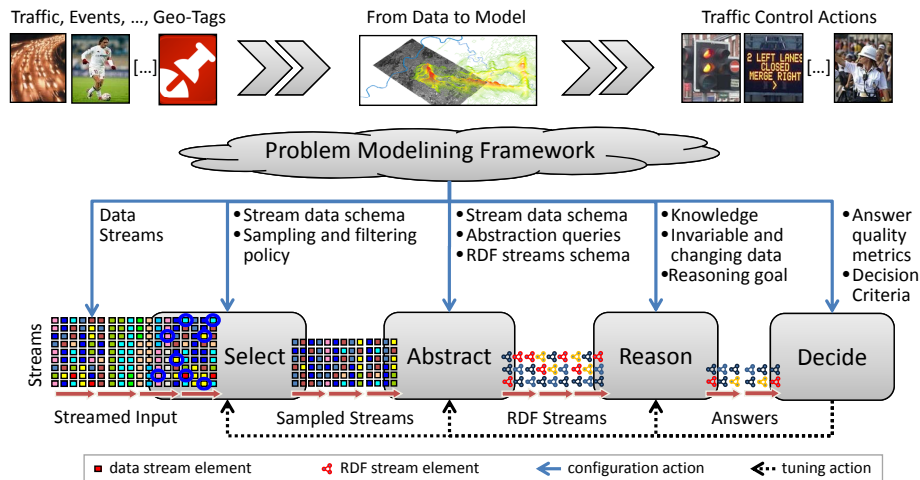


Fig. 1. Conceptual System Architecture

grounded in the Urban Computing field: data from the urban environment (e.g., traffic info, events, geo-tags, etc.) are translated in models and by reasoning on those models traffic control actions can be taken (e.g., controlling traffic lights, showing messages on traffic information panels, asking police intervention, etc.). The bottom part of the figure represents the four pluggable steps of the LarKC approach that we consider for Stream Reasoning<sup>2</sup> interconnected by the data that flow from left to right.

Data arrives to the Stream Reasoner as streamed input. A first step **selects** the relevant data in the input stream by exploiting *load-shedding* techniques [10]. Such techniques were developed to deal with bursty streams that may have unpredictable peaks during which the load may exceed available system resources. The key idea behind load-shedding is to introduce sampling policies that probabilistically drop stream elements as they are input to the selection step. Sampling and filtering policies can be either a) specified explicitly at stream-registration time, or b) inferred by gathering statistics over time, or c) by explicitly including punctuation in streams [11].

An example of sampling and filtering policy could be: if in a city a data stream originates from each traffic control camera, images should be sampled at given times rather than be continuously analyzed; in normal traffic condition, each stream could be sampled every 5 minutes, with options for increasing or decreasing the sampling rates (in congestion condition sample every 2 minutes, at night sample every 10 minutes).

The sampled streams resulting from the selection step are fed into a second step that **abstracts** from fine grain data streams into aggregated events. Such abstraction step can be done either by exploiting data compression techniques or

<sup>2</sup> We are explicitly omitting the retrieval step, because data stream retrieval should not be different from any other resource retrieval, therefore we will rely on pluggable components conceived by others.

6 E. Della Valle, S. Ceri, D. Barbieri, D. Braga and A. Campi

by aggregation queries. Data compression techniques includes the usage of histograms [12] or wave-lets [13], when the abstraction is meant to be an aggregation (e.g., counting the number of cars running through traffic detectors), and using Bloom filters [14] for duplicate elimination, set difference, or set intersection.

By **abstraction query** we mean, a continuous query that, given a large set of (possibly) unrelated low-level data in the input streams, produces an aggregated event. For instance, the abstraction step may rise a traffic congestion alert for a given street if the number of cars counted by the traffic control camera exceed 100 cars and it has been continuously increasing in the last 15 minutes.

The main proposition brought up in this paper is that, either for doing the abstraction step, or immediately after the abstraction, data streams are consolidated as **RDF streams**. RDF streams are new data formats set at the confluence of conventional data streams and of conventional atoms usually injected into reasoners. At this stage of our research, we envision two alternative formats for RDF streams:

- A **RDF molecules stream** is an unbounded bag of pairs  $\langle \rho, \tau \rangle$ , where  $\rho$  is a RDF molecule [15] and  $\tau$  is the timestamp that denotes the logical arrival time of RDF molecule  $\rho$  on the stream;
- A **RDF statements stream** is a special case of RDF molecules stream in which  $\rho$  is an RDF statement instead of an RDF molecule.

Descending from the two formats, we conceive two different stream reasoning frameworks. RDF molecule streams introduce stream reasoning as a progressive process, allowing for reuse of existing DSMS and reasoners. RDF statements stream introduce stream reasoning as a revolutionary process, requiring upon reasoners the same paradigm shift as the introduction of data streams upon databases. Section 4 and 5 describe respectively the two frameworks.

As last step, before producing the solution of the application problem of our concern (e.g., a congestion situation is monitored and traffic is rerouted according to planning activities), the answering process reaches the **decision** step. In such step quality metrics and decision criteria, defined by the application developer, are used to check if the quality of the answer is good enough and to adapt the behavior of each step (e.g., changing the sampling frequency).

## 4 RDF Molecules Stream Reasoning Framework

As we have just stated, RDF molecule streams introduce stream reasoning as a progressive process. They allow for reuse of existing DSMS and reasoners by coupling them using a transcoder and a pre-reasoner (see Figure 2).

In particular, the abstraction step can be realized using a DSMS and a transcoder. The DSMS receives the sampled data streams and generates an abstracted data stream by continuously answering the abstraction queries designed by the application developer, which typically perform an aggregation of events. The **transcoder** generates a stream element  $\langle \rho, \tau \rangle$ , where  $\rho$  is a RDF molecule

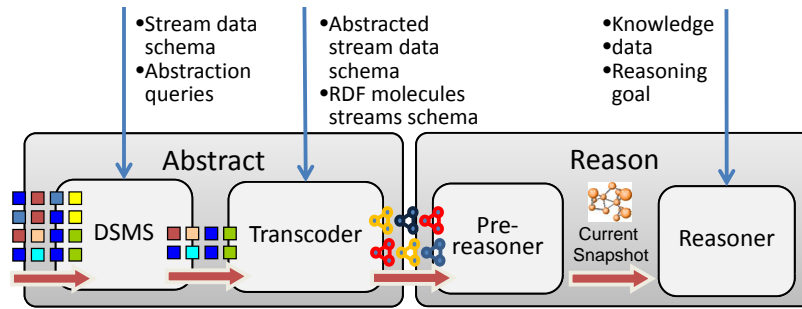


Fig. 2. RDF Molecules Stream Reasoning Framework.

and the timestamp  $\tau$  typically corresponds to the end of the aggregation interval, and puts it in the outgoing RDF stream.

We choose RDF molecules [15] as the minimum amount of information, because RDF molecules are the finest component into which an RDF graph can be decomposed without loss of information. Given that a data stream is composed by tuples and each tuple carries a minimum amount of processable information, a direct transcoding of each tuple into an RDF molecule is always possible.

For instance, in our Urban Computing example we may have a system of traffic sensors that feed a data stream by recording every sensed car across a given road. An aggregator associated with each sensor counts the number of vehicles, distinguishing them according to their type; then, the transcoder encodes this information into an RDF molecule stream element. For instance, an RDF molecule for this example is composed of four triples connected by a blank node  $\_ : x$ .

$$\left\langle \begin{array}{l} \text{http://uc.ex/tcc\#123} \quad \text{uc:measure} \quad \_ : x. \\ \_ : x \quad \text{uc:numberOfCars} \quad 120. \\ \_ : x \quad \text{uc:numberOfTrucks} \quad 70. \\ \_ : x \quad \text{uc:numberOfOtherVehicles} \quad 37. \end{array} \right\rangle, \text{Jun.17, 09:06:16AM}$$

RDF molecule streams are fed into **pre-reasoners** that perform the incremental maintenance of materialized *RDF snapshots*, i.e. RDF views describing the state of the system at a given time, which are given as input to reasoners according to application-specific strategies. Reasoners are not aware of time and produce a set of answers that remain valid until pre-reasoners produce the next snapshot. The efficient incremental materialization of RDF snapshots performed by pre-reasoners is a research challenge under investigation; background studies concern the incremental maintenance of materialized ontologies [16] and indexing of temporal XML documents [17].

## 5 RDF Statements Stream Reasoning Framework

As we anticipated in Section 3, we are also considering a more revolutionary approach, where streams are directly represented in RDF, and therefore continuous

8 E. Della Valle, S. Ceri, D. Barbieri, D. Braga and A. Campi

and/or aggregation queries can be directly expressed in RDF languages. Compared to RDF molecule streams, RDF statement streams are fine grain streams of triples. We envision the possibility to define up to eight different types of RDF statements streams depending upon the kind of information that changes at each stream input, ranging from a completely unspecified to a completely specified RDF triple. In the former case, every new element in the stream is an arbitrary RDF triple; in the latter case, every new element in the stream corresponds to the occurrence of an instance of RDF stream which is totally fixed (e.g., another unidentified vehicle seen at a given sensor). The following table summarizes the eight cases:

Name	Subject	Predicate	Object	Denotation
free	-	-	-	free
bound subject	s	-	-	s
bound predicate	-	p	-	p
bound object	-	-	o	o
free subject	-	p	o	po
free predicate	s	-	o	so
free object	s	p	-	sp
bound	s	p	o	spo

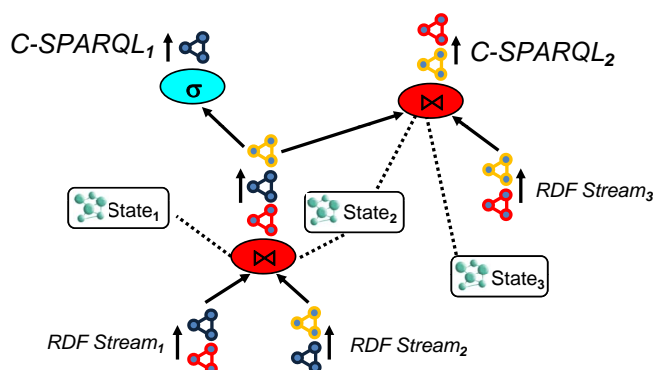
For RDF statement streams it is possible to define continuous queries both in terms of a formal abstract semantics and a concrete query language that implements the abstract semantics (namely *C-SPARQL*), following the path already explored in designing CQL [18] for DSMS.

As for CQL, the abstract semantics of such a C-SPARQL language is based on two data types, RDF statements streams (later on shortly named RDFstream) and **instantaneous RDF graphs** (later on shortly named **tgraph**). The two data types are a direct mapping of stream and relation data types in CQL. C-SPARQL queries are executed as trees of fine-grain operators performing selection and abstraction over streams; their optimization and parallelization can be approached by using techniques which are translated from DSMS systems. In Figure 3 we depict how we expect our C-SPARQL engine to share query plans among different registered queries in order to continuously answer in a throughput-efficient manner.

As for CQL, the abstract semantics of C-SPARQL includes operators of three classes:

- A *tgraph-to-tgraph* operator takes one or more tgraph as input and produces a tgraph as output.
- A *RDFstream-to-tgraph* operator takes a RDF statements stream as input and produces a tgraph as output.
- A *tgraph-to-RDFstream* operator takes a tgraph as input and produces a RDFstream as output.

RDFstream-to-tgraph operators use *sliding windows* [19] over RDF statements streams; their efficient evaluation can use the fact that stream elements



**Fig. 3.** RDF Statements Stream Reasoning Framework.

enter into windows and then exit from windows sequentially, according to the total order associated with time. RDF data is typically used in the context of ontological languages (e.g., RDF/S and OWL) enabling to describe resources and their properties. The efficient evaluation of multiple queries with several overlapping sliding windows upon both RDF data and language-specific ontological properties is a research challenge currently under investigation; methods presented in [16] can be adapted.

In this framework, scalability will be achieved by distribution and parallelism. Indeed, while each stream should be allocated to a given processor, all other operator-based computations can be distributed according to an explicit, well-defined data flow; hence, distributed database methods fully apply.

## 6 Conclusions and Future Works

In this paper we have presented some preliminary steps toward Stream Reasoning. Our main contribution is an integration architecture, taking advantage of the benefits of both data streams and reasoners, from which two stream reasoning frameworks can be derived.

The one based on RDF molecules is an evolution of the currently available solutions that relies on the possibility to couple DSMSs and state-of-the-art reasoners. This approach requires investigating an appropriate solution for incremental maintenance of time-varying RDF views and engineering throughput-efficient transcoder technology for bridging data streams to RDF Molecules Streams.

The one based on RDF statements is a revolutionary approach to reasoning that requires defining C-SPARQL semantics, studying its computational complexity, defining the concrete C-SPARQL language, and implementing a query processor that heavily exploits the intrinsic characteristic of streams.

### Acknowledgements

The work described in this paper has been partially supported by the European project LarkC (FP7-215535).

10 E. Della Valle, S. Ceri, D. Barbieri, D. Braga and A. Campi

## References

1. Kiryakov, A.: Measurable targets for scalable reasoning (2007)
2. Garofalakis, M., Gehrke, J., Rastogi, R.: Data Stream Management: Processing High-Speed Data Streams (Data-Centric Systems and Applications). Springer-Verlag New York, Inc., Secaucus, NJ, USA (2007)
3. Kindberg, T., Chalmers, M., Paulos, E.: Guest editors' introduction: Urban computing. *IEEE Pervasive Computing* **6**(3) (2007) 18–20
4. Arikawa, M., Konomi, S., Ohnishi, K.: Navitime: Supporting pedestrian navigation in the real world. *IEEE Pervasive Computing* **6**(3) (2007) 21–29
5. Reades, J., Calabrese, F., Sevtsuk, A., Ratti, C.: Cellular census: Explorations in urban data collection. *IEEE Pervasive Computing* **6**(3) (2007) 30–38
6. Bassoli, A., Brewer, J., Martin, K., Dourish, P., Mainwaring, S.: Underground aesthetics: Rethinking urban computing. *IEEE Pervasive Computing* **6**(3) (2007) 39–45
7. Fensel, D., van Harmelen, F., Andersson, B., Brennan, P., Cunningham, H., Della Valle, E., Fischer, F., Huang, Z., Kiryakov, A., il Lee, T.K., School, L., Tresp, V., Wesner, S., Witbrock, M., Zhong, N.: Towards larkc: a platform for web-scale reasoning, *IEEE International Conference on Semantic Computing (ICSC 2008)* (Aug. 2008)
8. Fensel, D., van Harmelen, F.: Unifying reasoning and search to web scale. *IEEE Internet Computing* **11**(2) (2007) 9695
9. Shvaiko, P., Giunchiglia, F., Bundy, A., Besana, P., Sierra, C., Van Harmelen, F., Zaihrayeu, I.: Benchmarking methodology for good enough answers. Technical report, DISI-08-003, Informatica e Telecomunicazioni, University of Trento (2008)
10. Tatbul, N., Çetintemel, U., Zdonik, S., Cherniack, M., Stonebraker, M.: Load shedding in a data stream manager. In: *vldb'2003: Proceedings of the 29th international conference on Very large data bases, VLDB Endowment* (2003) 309–320
11. Tatbul, N., Cetintemel, U., Zdonik, S., Cherniak, M., Stonebraker, M.: Exploiting punctuation semantics in continuous data streams. *IEEE Trans. on Knowledge and Data Eng.* **15**(3) (2003) 555–568
12. Thaper, N., Guha, S., Indyk, P., Koudas, N.: Dynamic multidimensional histograms. In: *SIGMOD '02: Proceedings of the 2002 ACM SIGMOD international conference on Management of data, New York, NY, USA, ACM* (2002) 428–439
13. Chakrabarti, K., Garofalakis, M., Rastogi, R., Shim, K.: Approximate query processing using wavelets. *The VLDB Journal* **10**(2-3) (2001) 199–223
14. Bloom, B.H.: Space/time trade-offs in hash coding with allowable errors. *Commun. ACM* **13**(7) (1970) 422–426
15. Ding, L., Finin, T., Peng, Y., da Silva, P.P., McGuinness, D.L.: Tracking RDF Graph Provenance using RDF Molecules. Technical report, UMBC (April 2005)
16. Volz, R., Staab, S., Motik, B.: Incrementally maintaining materializations of ontologies stored in logic databases. *J. Data Semantics* **2** (2005) 1–34
17. Mendelzon, A.O., Rizzolo, F., Vaisman, A.: Indexing temporal xml documents. In: *VLDB '04: Proceedings of the Thirtieth international conference on Very large data bases, VLDB Endowment* (2004) 216–227
18. Babu, S., Widom, J.: Continuous queries over data streams. *SIGMOD Rec.* **30**(3) (2001) 109–120
19. Babcock, B., Babu, S., Datar, M., Motwani, R., Widom, J.: Models and issues in data stream systems. In: *PODS '02: Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, New York, NY, USA, ACM* (2002) 1–16

## BIBLIOGRAPHY

- [1] E. Bingham and H. Mannila. Random projection in dimensionality reduction: applications to image and text data. In *Knowledge Discovery and Data Mining*, pages 245–250, 2001.
- [2] K. Bontcheva, V. Tablan, D. Maynard, and H. Cunningham. Evolving GATE to Meet New Challenges in Language Engineering. *Natural Language Engineering*, 10(3/4):349–373, 2004.
- [3] C. Burgess and K. Lund. Modelling parsing constraints with high dimensional context space. *Language and Cognitive Processes*, 12:1–34, 1997.
- [4] S. Chakrabarti. *Mining the Web – Discovering Knowledge from Hypertext Data*. Morgan Kaufmann Publishers, 2003.
- [5] G. Cheng, W. Ge, and Y. Qu. Falcons: Searching and Browsing Entities on the Semantic Web. In *Proceedings of WWW2008*, pages 1101–1102, 2008.
- [6] S. Clark, B. Coecke, and M. Sadrzadeh. A compositional distributional model of meaning. In *Proceedings of the Second Quantum Interaction Symposium (QI-2008)*. College Publications, 2008.
- [7] H. Cunningham. GATE, a General Architecture for Text Engineering. *Computers and the Humanities*, 36:223–254, 2002.
- [8] H. Cunningham. Information Extraction, Automatic. *Encyclopedia of Language and Linguistics, 2nd Edition*, pages 665–677, 2005.
- [9] H. Cunningham and K. Bontcheva. Computational Language Systems, Architectures. *Encyclopedia of Language and Linguistics, 2nd Edition*, pages 733–752, 2005.
- [10] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*, 2002.
- [11] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41:391–407, 1990.

- [12] M. Dowman, V. Tablan, H. Cunningham, and B. Popov. Web-assisted annotation, semantic indexing and search of television and radio news. In *Proceedings of the 14th International World Wide Web Conference*, Chiba, Japan, 2005. <http://gate.ac.uk/sale/www05/web-assisted-annotation.pdf>.
- [13] D. Fensel, F. van Harmelen, B. Andersson, P. Brennan, H. Cunningham, E. Della Valle, F. Fischer, Z. Huang, A. Kiryakov, T. K. Lee, L. School, V. Tresp, S. Wesner, M. Witbrock, and N. Zhong. Towards larkc: a platform for web-scale reasoning. In *Proceedings of the IEEE International Conference on Semantic Computing (ICSC 2008)*, Santa Clara, CA, USA, 2008. IEEE Computer Society Press.
- [14] A. Fokoue, A. Kershenbaum, L. Ma, E. Schonberg, and K. Srinivas. The summary abox: Cutting ontologies down to size. In *Proc. of the Int. Semantic Web Conf. (ISWC2006)*, pages 136–145, 2006.
- [15] M. N. Jones and D. J. K. Mewhort. Representing word meaning and order information in a composite holographic lexicon. *Psychological Review*, 114:1–37, 2007.
- [16] S. Kaski. Dimensionality reduction by random mapping. In *Proceedings of International Joint Conference on Neural Networks (volume 1)*, pages 413–418, 1998.
- [17] W. Kintsch. *Comprehension: A paradigm for cognition*. Cambridge University Press, New York, 1998.
- [18] A. Kiryakov, B. Popov, D. Ognyanoff, D. Manov, A. Kirilov, and M. Goranov. Semantic annotation, indexing and retrieval. *Journal of Web Semantics, ISWC 2003 Special Issue*, 1(2):671–680, 2004.
- [19] Atanas Kiryakov. OWLIM: balancing between scalable repository and light-weight reasoner. In *Proc. of WWW2006*, Edinburgh, Scotland, 2006.
- [20] G. Klyne and J. J. Carroll. Resource description framework (rdf): Concepts and abstract syntax. W3C recommendation - 10 feb 2004, W3C, 2004. Available at <http://www.w3.org/TR/rdf-concepts/>.
- [21] T. K. Landauer and S. T. Dumais. A solution to plato’s problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104:1997, 211–240.
- [22] Y. Li, K. Bontcheva, and H. Cunningham. SVM Based Learning System For Information Extraction. In M. Niranjan J. Winkler and N. Lawrence, editors, *Deterministic and Statistical Methods in Machine Learning*, LNAI 3635, pages 319–339. Springer Verlag, 2005.
-

- [23] Y. Li, K. Bontcheva, and H. Cunningham. Using Uneven Margins SVM and Perceptron for Information Extraction. In *Proceedings of Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, 2005.
- [24] Y. Li, K. Bontcheva, and H. Cunningham. Hierarchical, Perceptron-like Learning for Ontology Based Information Extraction. In *16th International World Wide Web Conference (WWW2007)*, pages 777–786, May 2007.
- [25] K. Lund and C. Burgess. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, and Computers*, 28:203–208, 1996.
- [26] V. Lux-Pogodalla and E. S. Juan. Query refinement by multiword term expansions and semantic synonymy. In *Proceedings of The 1st International Conference on Multidisciplinary Information Sciences and Technologies (InScit2006)*, 2006.
- [27] D. Nelson and C. McEvoy. Entangled Associative Structures and Context. In *Proceedings of the AAAI Spring Symposium on Quantum Interaction*. AAAI Press, 2007.
- [28] D. Pavlovic. On quantum statistics in data analysis. In *Proceedings of the Second Quantum Interaction Symposium (QI-2008)*. College Publications, 2008.
- [29] B. Popov, A. Kiryakov, A. Kirilov, D. Manov, D. Ognyanoff, and M. Goranov. KIM – A semantic platform for information extraction and retrieval. *Natural Language Engineering*, 10:375–392, 2004.
- [30] E. Prud’hommeaux and A. Seaborne. Sparql query language for rdf. W3C recommendation - 15 january 2008, W3C, 2008. Available at <http://www.w3.org/TR/rdf-sparql-query/>.
- [31] Y. Qu, W. Hu, and G. Cheng. Constructing virtual documents for ontology matching. In *Proceedings of WWW2006*, pages 23–31, 2006.
- [32] M. Sahlgren. An introduction to random indexing. Technical report, Swedish Institute of Computer Science, 2005.
- [33] M. Sahlgren, A. Holst, and P. Kanerva. Permutations as a means to encode order in word space. In *Proceedings of the 30th Annual Meeting of the Cognitive Science Society (CogSci’08)*, Washington D.C., USA, 2008.
- [34] M. Sahlgren and J. Karlgren. Buzz monitoring in word space. In *European Conference on Intelligence and Security Informatics (EuroISI 2008)*, Esbjerg, Denmark, 2008.
- [35] K. van Rijsbergen. *The Geometry of Information Retrieval*. Cambridge, 2004.

- [36] D. Widdows. Orthogonal negation in vector spaces for modelling word-meanings and document retrieval. In *The 41st Annual Meeting of the Association for Computational Linguistics*, pages 136–143, 2003.
- [37] D. Widdows. Semantic vector products: Some initial investigations. In *Proceedings of the Second Quantum Interaction Symposium (QI-2008)*. College Publications, 2008.
- [38] D. Widdows and P. Bruza. Quantum information dynamics and open world science. In *Proceedings of the First Quantum Interaction Symposium (QI-2007)*. AAAI Press, 2007.
- [39] D. Widdows and S. Peters. Word vectors and quantum logic: Experiments with negation and disjunction. In *Eighth Mathematics of Language Conference*, pages 141–154, 2003.