Simulation Unification: Beyond Querying Semistructured Data

François Bry[1] **and Sebastian Schaffert**[2]

1 **Institute for Informatics, Ludwig-Maximilians University of Munich, Germany**
`http://pms.ifi.lmu.de`
2 **Salzburg Research Forschungsgesellschaft, Austria**
`http://www.salzburgresearch.at/`

Abstract

This article first reminds of simulation unification, a non-standard unification proposed at the 18th International Conference on Logic Programming (ICLP 2002) for making logic programming capable of querying semistructured data on the Web. This article further argues that, beyond querying semistructured data on the Web, simulation unification has a potential for Web querying of multimedia data and semantic metadata and for Web searching of data of all kinds.

1998 ACM Subject Classification D3.3 Language Constructs and Features

Keywords and phrases Simulation Unification, (Semantic) Web Querying

Digital Object Identifier 10.4230/LIPIcs.ICLP.2012.1

1 Introduction

This article is devoted to simulation unification, a non-standard unification which has been introduced in 2002 at the 18th International Conference on Logic Programming (ICLP 2002) with the article titled "Towards a Declarative Query and Transformation Language for XML and Semistructured Data: Simulation Unification" [15] and the long version [16] of that article. Simulation unification has been specified in more detail two years later, in 2004, in the doctoral thesis "Xcerpt: A Rule-Based Query and Transformation Language for the Web" [30] of Sebastian Schaffert.

This article recalls simulation unification and argues that it has a so far unexploited potential for Web querying of multimedia and semantic data as well as for Web searching of data of all kinds.

This article is structured as follows. After this introduction, Section 2 describes the context in which and why we developed simulation unification. Section 3 is a brief, and simplified, reminder of simulation unification. Section 4 is devoted to works related to simulation unification. Section 5 discusses how simulation unification could be applied to querying multimedia and semantic data and to searching. Section 6 is a conclusion.

2 What Led to Simulation Unification

At the beginning of the 90es of the 20th century, as the Web became a common medium, many computer scientists first did not fully realised what impact the Web would have on their areas of research. This was the case amongst others of the query answering community. At the end of the 90es, that community hastily investigated Web query languages, what resulted in XQuery [10], a "recommendation" of the W3C, so as to keep an hold on data access. This community celebrated XQuery amongst others for its roots in functional programming,

Technical Communications of the 28th International Conference on Logic Programming (ICLP'12).
Editors: A. Dovier and V. Santos Costa; pp. 1–13

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

and promoted it as a worthy descendant of SQL [20], the query language which had greatly contributed to the success of relational databases.

We took the enthusiasm for XQuery with a skepticism rooted at a logic programming practice. We found that XQuery was difficult to program with; we thought it would often yield inelegant and therefore costly to maintain programms and we guessed that it would require complicated runtime systems. Since, these intuitions have been amply confirmed and XQuery is no longer the subject of much enthusiasm.

The study reported about in [28] had hinted at the potential of logic programming for querying semistructured data. That article shows that restricting XPath [21], the data selection sub-language of XQuery, to its forwards axes, that is, to so-called Forward XPath, does not restrict the data selection language's expressivity. Since a Forward XPath expression basically amounts to a logic atom, a link between logic programming an Web querying was established. The afore mentioned article [28] has received some attention because it makes it possible to restrict formal investigations on XPath to XPath Forward what gives rise to significant simplifications. Surprisingly, that the restriction to XPath Forward also, and for the same reasons, gives rise to simpler queries and therefore eases both, programming and query evaluation, has been rarely noticed.

Pattern-based queries for Web data had been proposed with the Web query languages UnQL [19] and XML-QL [22] what suggested a full unification binding variables in the two terms considered instead of a pattern matching binding variables in only in the pattern.

These two observations led us to simulation unification, a technique that makes logic programming as convenient for querying semistructured data as for querying relational data.

Since, search engines, other tools the importance of which has not been immediately understood within the query answering community, have considerably reduced the need for Web query languages. Indeed, data are no longer only queried for but also, and mostly, search for. In this article, we arge that, beyond querying semistructured data, simulation unification also has a potential for both, Web querying of multimedia and semantic data and Web searching of data of all kinds.

3 What is Simulation Unification?

Given two terms t_1 and t_2 simulation unification [15, 16, 30] determines, if possible, a most general unifier σ for the variables in t_1 and t_2 such that every ground instances of $t_1\sigma$ simulates in a ground instance of $t_2\sigma$.

Simulation unification is based on an adaption of graph simulation to terms aimed at representing, selecting (or querying) and constructing XML data. Simulation unification's principles are relatively simple. The syntactical richness necessary for an easy expression of data selections and construction makes it, however, complicated.

In the remainder of the current section 3, rooted graph simulation is introduced in Section 3.1, database terms, query terms and construct terms in Section 3.2, term simulation and answers to query terms in Section 3.3 and simulation unification in Section 3.4.

3.1 Rooted Graph Simulation

Simulation, also called graph simulation, has been studied in [26, 27]. A term t_1 (seen as a graph G_1) simulates in a term t_2 (seen as a graph G_2) if there is a mapping of the nodes of G_1 (that is of the subterms of t_1) in the nodes of G_2 (that is, the subterms of t_2) which preserves the edges (that is, subterm nesting). Simulation is similar to, though more general

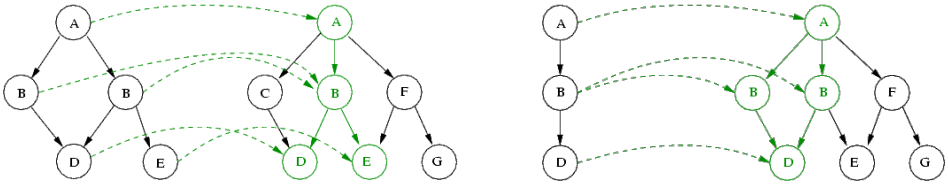

Figure 1 Two simulations (with respect to node label equality) [15].

than, graph homomorphism because it allows two nodes of the one graph being mapped to a single node of the other graph and vice versa.

In general, there might be more than one simulation between two graphs. Therefore, so-called minimal simulations are considered.

Figure 1 from [15] gives two examples of simulations. In each of these two examples a node of the left graph is mapped into a node of the right graph if their labels are identical. Such simulations are simulations with respect to label identity. More generally, a simulation can be defined with respect to any preorder relation (amongst other order and equivalence relations). In Section 5, we argue that considering other relations than label equality makes simulation unification convenient for querying multimedia and semantic data on the Web as well as for searching for data of all kinds on the Web.

The following definition from [30] which refines that of [15] is inspired from [26, 27]. A (directed) rooted graph $G = (V, E, r)$ consists in a set V of vertices (or nodes), a set E of edges (that is, ordered pairs of vertices), and a selected vertex r, called the root of G, from which each vertex of G is accessible.

▶ **Definition 1** (Rooted graph simulation with respect to a preorder relation $\sim$ [30]). Let $G_1 = (V_1, E_1, r_1)$ and $G_2 = (V_2, E_2, r_2)$ be rooted graphs and $\sim \subseteq V_1 \times V_2$ a preorder relation. A relation $\mathcal{S} \subseteq V_1 \times V_2$ is a *rooted simulation* of G_1 in G_2 with respect to $\sim$ if:

1. $r_1 \; \mathcal{S} \; r_2$.
2. If $v_1 \; \mathcal{S} \; v_2$, then $v_1 \sim v_2$.
3. If $v_1 \; \mathcal{S} \; v_2$ and $(v_1, v_1', i) \in E_1$, then there exists $v_2' \in V_2$ such that $v_1' \; \mathcal{S} \; v_2'$ and $(v_2, v_2', j) \in E_2$

A rooted simulation $\mathcal{S}$ of G_1 in G_2 with respect to $\sim$ is *minimal* if there are no rooted simulations $\mathcal{S}'$ of G_1 in G_2 with respect to $\sim$ such that $\mathcal{S}' \subset \mathcal{S}$ (and $S \neq S'$).

Graph simulation conveys well how the Web is queried. Web queries are mostly incomplete specifications of data striven for that are conveniently answered by data items containing more than the query specifies and allowing that distinct parts of the query are answered by the same data. The relevance of graph simulation for Web querying has been first pointed out in [19, 22].

3.2 Database Terms, Query Terms and Construct Terms

3.2.1 Database terms

Database terms are an abstraction of XML documents and a generalisation of the ground terms of logic. Database terms are similar to logic ground terms except that the arity of a function symbol, called "label", is not fixed but variable, and that the order of the arguments of a function symbol might not be compelling.

A database term whith a root labelled l and ordered children $t_1, \ldots, t_n$ is denoted $l[t_1, \ldots, t_n]$. A databasse term with a root labelled l and unordered children $t_1, \ldots, t_n$ is denoted $l\{t_1, \ldots, t_n\}$.

Cyles, possible in XML documents though hypertext links and ID-IDREF references, are allowed in database terms but not considered in the following for the sake of briefness. A database terms without cycles can be seen as a tree, a database term with cycles as a rooted graph.

3.2.2 Query terms

Query terms are patterns specifying selections of ground terms terms. They are similar to logic atoms except that they can express incompleteness in breath and depth and that a variable X in a query term can be restricted.

In a query term,

- the brackets [ ] and { } require answers with no more ordered respectively unordered subterms than the query term;
- double brackets [[ ]] and {{ }} accept answers having more ordered respectively unordered subterms than the query term;
- a variable X can be constrained to some query terms Q using $X \rightsquigarrow Q$, where $\rightsquigarrow$ is read "as";
- $X \rightsquigarrow \mathit{desc}\ t$, read "$X$ descendant t", is used to express that X is bound to a term containing a subterm t at an unspecified depth.

Multiple constraints for a a same variable are allowed. Figure 2 hints at the semantics of query terms formally specified in [17, 18, 30].

Constraining variables (with $\rightsquigarrow$) might results in cyclic constraints that cannot be answered by database terms because database terms are finite. A variable X is said to depend on a variable Y in a query term t if $X \rightsquigarrow t_1$ is a subterm of t and Y is a subterm of t_1. A query term t is said to be variable well-formed if it contains no variables $X_0, \ldots, X_n$ $(n \geq 1)$ such that $X_0 = X_n$ and for all $i = 1, \ldots n$ X_i depends on X_{i-1} in t. Only variable well-formed query terms are considered in the following.

A query term is ground if it contains no variables (and therefore no $\rightsquigarrow$ and no desc).

Further constructs such as "option" and "except" might occur in query terms so as to ease the expression of some queries [30, 11, 31]. They are not considered in the following for the sake of briefness.

3.2.3 Construct terms

Construct terms serve to re-assemble the values which are specified in query terms by variables, so as to form new database terms. Thus,[ ], { } and variables may occur in construct terms but neither [[ ]], nor {{ }}, nor $\rightsquigarrow$. In a construct term, a variable might be preceded by "all" meaning that all values, or bindings, for this variables are to be gathered.

Rules combine construct terms and query terms in the manner of logic programming: A rule head is a construct term; a rule body is build up from query terms, conjunctions, disjunctions, and negations.

Like in [15], simulation unification is defined below under the simplifying assumption that {} and {{}} are the only kinds of braces. The complete definition is given in [30].

Query terms	Possible answers	No answers
$a[[b, c\{d, e\}, f]]$	$a[b, c\{d, e, g\}, f]$ $a[b, c\{d, e, g\}, f\{g, h\}]$ $a[b, c\{d, e\{g, h\}, g\}, f\{g, h\}]$ $a[b, c[d, e], f]$	$a\{b, c\{d, e\}, f, g\}$ $a[b, c\{d, e\}, f, g]$ $a\{b, c\{d, e\}, f\}$
$a[desc\ f[c, d], b]$	$a[f[c, d], b]$ $a[g[f[c, d]], b]$ $a[g[f[c, d], h], b]$ $a[g[g[f[c, d]]], b]$ $a[g[g[f[c, d], h], i], b]$	$a[b]$ $a[g, b[f[c, d]]]$
$a[X \rightsquigarrow b[c, d], Y, e]$	$a[b[c, d], f, e]$ X *bound to* $b[c, d]$ Y *bound to* f $a[b[c, d], f[g, h], e]$ X *bound to* $b[c, d]$ Y *bound to* $f[g, h]$	$a[c, f, e]$ $a[b[c], f, e]$ $a[h[b, c], f, e]$
$a\{X \rightsquigarrow b\{c\}, X \rightsquigarrow b\{d\}\}$	$a\{b\{c, d\}\}$ X *bound to* $b\{c, d\}$	$a\{b\{c\}\}$ $a[b[c], f, e]$
$a[X \rightsquigarrow b\{c\}, X \rightsquigarrow f\{d\}]$	*none*	$a[b\{c\}]$ $a[f\{d\}]$ $a[b\{c\}, f\{d\}]$
$a\{\{\}\}$	a	$a\{b\}$ $a\{b, c\}$ $a[b]$ $a[b, c]$

Figure 2 Query terms.

3.3 Term Simulation and Answers

Substitutions and grounding substitutions are defined as usual except that they assign construct terms, but no query terms, to variables. Instances and ground instances of query and construct terms are defined as usual except that an instance of $X \rightsquigarrow t$ is defined as an instance of X (that is, ignoring $\rightsquigarrow t$). $\rightsquigarrow$ and desc induce constraints on variables and subterms of a query term. Instances of a query that fulfill these constraints are called allowed instances. Only allowed instances are considered in the following.

Simulation of a graph G_1 into a graph G_2 is adapted into the simulation of a ground query term Q into a ground construct term t by paying the necessary attention to the brackets $\{\}$ and $\{\{\}\}$. Ground term simulation is then extended to query and construct terms with variables as follows: A query term Q simulates into a construct term t, denote $Q \preceq t$, if there exists a substitution σ such that every ground instance of $Q\sigma$ simulates into a ground instance of $t\sigma$.

An answer to a query term Q is a database term t such that an allowed instance of Q simulates in t. As usual, substitution (so-called answer substitutions) are associated with term answers. Because of the construct desc serving to express subterm constraints and in contrast to classical logic programming, answers cannot be fully defined by answer substitutions.

3.4 Simulation Unification

Simulation unification is a non-deterministic algorithm for solving equations of the form $Q \preceq t$, read Q simulates in t, on query terms Q and construct terms t. It is based on the following term decomposition rules – see [15, 16, 30] for a detailed description of the non-deterministic algorithm. The outcome of simulation unification, if it succeeds, is a finite set of substitutions called simulation unifier.

▶ **Definition 2** (Term Decomposition Rules)**.** Let l (with or without indices) denote a label. Let t^1 and t^2 (with or without indices) denote query terms.

- **Root Elimination:**
  (1) $l \preceq l\{t_1^2, \ldots, t_m^2\} \Leftrightarrow \mathit{true}$ if $m \geq 1$
  $l \preceq l\{\{\}\} \Leftrightarrow \mathit{true}$

  (2) $l\{t_1^1, \ldots, t_n^1\} \preceq l \Leftrightarrow \mathit{false}$ if $n \geq 1$
  $l\{t_1^1, \ldots, t_n^1\} \preceq l\{\{\}\} \Leftrightarrow \mathit{false}$ if $n \geq 1$

  (3) Let Π be the set of (total) functions $\{t_1^1, \ldots, t_n^1\} \to \{t_1^2, \ldots, t_m^2\}$:
  $l\{t_1^1, \ldots, t_n^1\} \preceq l\{t_1^2, \ldots, t_m^2\} \Leftrightarrow \bigvee_{\pi \in \Pi} \bigwedge_{1 \leq i \leq n} t_i^1 \preceq \pi(t_i^1)$
  if $n \geq 1$ and $m \geq 1$

  (4) $l_1\{t_1^1, \ldots, t_n^1\} \preceq l_2\{t_1^2, \ldots, t_m^2\} \Leftrightarrow \mathit{false}$ if $l_1 \neq l_2$ and $n \geq 0$ and $m \geq 0$

- **$\rightsquigarrow$ Elimination:**
  $X \rightsquigarrow t^1 \preceq t^2 \;\Leftrightarrow\; t^1 \preceq t^2 \;\wedge\; t^1 \preceq X \;\wedge\; X \preceq t^2$
- **Descendant Elimination:**
  $\mathit{desc}\ t^1 \preceq l_2\{t_1^2, \ldots, t_m^2\} \Leftrightarrow t^1 \preceq l_2\{t_1^2, \ldots, t_m^2\} \vee \bigvee_{1 \leq i \leq m} \mathit{desc}\ t^1 \preceq t_i^2$
  if $m \geq 0$

Simulation unification is sound and complete for the notion of answer recalled above [15, 16, 30] . Like standard unification, simulation unification is symmetric since it can bind variables in the two terms considered. Unlike standard unification, however, it is asymmetric in the sense that it does not make the two terms considered equal, but instead makes the one simulate into the other what in general is no symmetrical relationship.

4 Work Related to Simulation Unification

How simulation unification relates to classical involved forms of unifications is addressed in [15] as follows:

> Several unification methods have been proposed that, like simulation unification, process flexible terms or structures, notably feature unification [1, 34] and associative-commutative-unification, short AC-unification, [23]. Simulation unification differs from feature unification in several aspects (discussed in [16]). Simulation unification might remind of theory unification [2]. The significant difference between both is that simulation unification is based upon an order relation, while theory unification refers to a congruence relation.

Simulation unification offers a decidable alternative to equational unification [2]. We argue in the following Section 5 that novel forms of simulation unification based on verious

embedding or similarity relationships would be useful in querying multimedia and semantic data as well as in searching data of all kinds.

Simulation unification has been developed for the textual query language and its visual companion visXcerpt. Between 2002 and 2006, research prototypes of Xcerpt and visXcerpt have been presented at database, Web, Semantic Web, logic programming and visual programming conferences [14, 6, 7, 31, 3, 13, 4, 5].

A subsumption referring to simulation unification, called simulation subsumption, and its use for query optimization have been introduced in [12]. Simulation subsumption expreses query containement for queries based on simulation unification. Simulation subsumption is useful for the query optimization, in particular for verifying the termination of recursive queries.

5 Beyond Querying Semistructured Data

Since the publication of our original article in 2002, the Web has undergone several major developments. First, the Semantic Web effort with its underlying technologies RDF and OWL has gained much momentum with the emergence of "Linked Data" as a means to publish semi-structured data using a uniform model for data representation and interlinking between datasets. Second, while the Web of 2002 was still mainly a static, text-based Web, the Web of 2012 is interactive and mostly consists of multimedia content. And third, with the success of Social Software, the amount of content and data on the Web has grown tremendously, making effective and efficient Web search more and more important. In the following, we will briefly describe how our ideas concerning simulation unification are more important than ever for addressing typical problems in these areas.

5.1 Generalising Simulation Unification

Simulation unification gives rise to queries retrieving structural sub-patterns within XML data. This can be generalized to other kind of data in two complementary ways:

- The first generalisation would build upon an "embedding" relationship on the data considered which, like simulation unification, would not be symmetric.
- The second generalisation would build upon a "similarity" relationship on the data considered which, in contrast to simulation unification, would be symmetric.

In the following sections, we describe how these two generalisations could help addressing open problems in several other areas.

5.2 RDF, RDFS and OWL

The Resource Description Framework, or "RDF" [37], is the primary model for publishing data on the Semantic Web. At its core, it defines a graph model where vertices represent Web resources (identified by URIs) or literal values and edges (so-called "triples") represent typed relations between Web resources. RDF also defines a number of different serialization formats for this graph data, e.g. RDF/XML, Turtle, or N-Triples. Schema information about an RDF graph can be defined using the schema languages RDFS [36] or OWL [35]. Both are capable of representing ontological knowledge about the schema in addition to specifying possible relations and are based on some form of logics.

Querying RDF Data. An important aspect of RDF is querying the graph data contained in a dataset. Typical RDF queries for example express RDF subgraphs to be found in the data queried. Given the graph model underlying RDF, pattern-based querying

of RDF is natural. In fact, the most widely used RDF query language SPARQL [39] uses so-called "triple patterns" specifying edges to look for in the dataset. Variables in triple patterns are bound to values when matching a pattern in the same style as in other logic programming languages.

While SPARQL is already a well designed and widely established query language, it is currently only defined in terms of a query algebra similar to the *relational algebra* and is not offering a *declarative calculus*. A query approach based on *(unrooted) simulation unification* could provide such a calculus for SPARQL in a style similar to the relational calculus behind SQL and Datalog. It would thus allow for a more declarative semantics and advanced reasoning services over RDF by opening up RDF querying to logic programming approaches. Note that this would also give rise to expressing RDFS and OWL ontology semantics in terms of logic programming rules. Querying RDF data corresponds to the "embedding relationship" of simulation unification described above.

Matching RDF Datasets. On the Semantic Web with many independent data publishers a common challenge is so-called "schema alignment" or "data alignment". In schema (or data) alignment, the goal is to create mappings between two different schemas (or datasets) to allow better interoperability and exchange of the data. A common way of doing schema alignment is to map concepts from the two schemas that are "similar" regarding different criteria.

For example, both schemas might define their own "Student" concept but with slightly different properties:

- Schema 1 defines a *Student* with *full name*, *email* and *inscription number*
- Schema 2 defines a *Student* with *first name* and *last name*, as well as *email*

Schema alignment could map between the Students of Schema 1 and 2 based on the name of the concept, the shared property *email*, and the similarity between *first name*/*last name* on the one hand and *full name* on the other hand.

A lot of research has been undertaken to investigate automatic means to carry out schema alignment. Nevertheless, many problems in this area today remain unsolved [33]. When representing the different attributes of a concept in terms of a graph structure, simulation unification in the second generalisation ("similarity relationship") could offer a new option for identifying similar concepts by trying to find a maximal simulation between the graphs representing two concepts.

5.3 Linked Data

Linked Data [8, 25] is a recent development within the Semantic Web effort to publish datasets of various sizes on the Web for anyone to use and combine, using the technologies developed in the Semantic Web context (mainly URIs and RDF). In his initial announcement, Tim Berners-Lee described four "Linked Data Principles" [8]:

1. Use URIs as names for things
2. Use HTTP URIs so that people can look up those names.
3. When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL)
4. Include links to other URIs. so that they can discover more things.

Since then, numerous datasets have been made available under these principles. As of September 2011, the known part of the "Web of Data" consists of about 300 datasets from various domains with more than 30 billion triples. Moreover, these datasets are connected with each other with about 500 million RDF links [9].

Conceptually, the Linked Data Cloud (or "Web of Data") can be seen as an RDF graph structure over distributed information systems. Since no dataset has information about the full graph, non-local parts can only be discovered by following RDF links that span across different servers.

Even though resources on Linked Data servers are typically interlinked and thus conceptually integrate data from many different sources, querying such data is still very cumbersome. The main reason is that existing query languages for RDF like SPARQL are rather dataset-centric and do not easily query over distributed or even unknown sources. There are currently four approaches to address this issue:

- a central index harvests the Web for RDF data and stores it in a central repository and offers it for querying, e.g. using SPARQL. This approach is followed e.g. by Sindice,[1] which offers a public SPARQL endpoint.
- a query is distributed over several query endpoints and the results are then combined. This approach is proposed in the SPARQL 1.1 Federation Extension [38].
- accepting the incompleteness of the results returned by the query and trying to improve the recall by different heuristics, as proposed e.g. by Hartig et.al. [24]

The first two approaches have obvious disadvantages: a central repository is not always recent and a single point of failure, while explicit federated queries are cumbersome to write and need exact information on how and where to access the SPARQL endpoint. They also require that all queried servers implement the SPARQL 1.1 Federation Extensions. The third approach is in our opinion not very user friendly, since the user cannot easily determine whether the results he will get are complete or not and important enterprise decisions might depend on that information.

In [32], we therefore proposed a path-based approach that is more suitable for querying Linked Data. However, a path language only allows binding one variable at a time (the "end" of the path) and is therefore rather limited in its expressivity and performance. Rooted simulation unification as described previously for XML could give rise to a novel kind of query language for Linked Data that does not share the problems of SPARQL and goes beyond the expressivity of simple path navigations. A query pattern could use a context resource from a local dataset as query root, follow links to other remote datasets and then bind multiple variables at the same time, reducing the number of network requests and providing a convenient way for formulating a query.

As an example, consider users publishing their basic profile information using the FOAF (friend-of-a-friend) vocabulary. Each user publishes on his website an RDF file with his name and email address, as well as links (`foaf:knows`) to the FOAF files of his friends. A query based on simulation unification could then select the first name, last name and email addresses of each friend in a single query by starting at the local FOAF file, following `foaf:knows`, and binding the three variables at the same time. Such a query is currently neither possible using SPARQL nor using a path-based approach.

5.4 Multimedia

An embedding realtionship can be specified for multimedia data expressing that, for example, a given visual pattern can be found in a picture or in a video. Such a relationship can be defined in terms of either geometrical image recognition algorithms, of features extracted from the multimedia content, or of symbolic metadata associated with picture. Rather

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

different embedding relationships can be thought of that would fullfil the needs of different applications. For example:

- Existing metadata could be queried. In the scenario described in [32], we are working with cliff-diving videos provided by Red Bull that are accompanied with precise descriptions of the scene, transcripts of interviews, as well as music cue sheets and general metadata about a video (persons, locations, editor, description). A query based on Simulation Unification could query for all videos with a certain person at a certain location.
- Multimedia information extraction could automatically extract faces of persons (e.g. using an Eigenfaces algorithm) as well as prominent structures (e.g. edges with sharp contrast) from a large collection of images and videos and store them as features. Simulation Unification could be used to provide a query with some sample features (the face of a person and a tower in the background) and the be evaluated over the image collection to retrieve matching images.

Similarity relationships are often used in retrieving multimedia data. Indeed, multimedia applications require to retrieve images similar to some given images. Image similarity can be specified in many different manners, with and without simliarity threshold to be fulfiled by the selected data.

5.5 Web Search and Enterprise Search

With the tremendous increase in content, Web Search and Enterprise Search are nowadays the most important way of finding and accessing information. The most important difference between Web Search and Enterprise Search is that Web Search can make use of the hyperlinks between documents (e.g. in Google's PageRank [29]) and the novelty of documents, while enterprise content is typically not connected and novelty is not necessarily a good measure for relevance.

Web Search and Enterprise Search could benefit from generalisations of simulation based on emebedding or similarity relationships in the following typical search tasks:

- *Search:* Both Web and Enteprise Search build in its core build upon the occurring of a words, or phrase, or of an ordered list of words or phrases in documents. Such a relationship could be replaced by embedding or similarity relationships for multimedia or semantic data of the afore mentioned kind. This would result in multimedia and semantic search engines at the the cost of indexing a well-chosen selection of patterns. For example, this would allow searches like "the fantasy book with the blue cover".
- *Grouping:* Search results often contain many similar documents, e.g. different versions of the same document in an enterprise setting. When displaying the search results, such documents should be grouped and displayed together. Detecting such groups can be a very hard task. A simulation unification for similarity relationships could be used for clustering similar documents based on various document properties.
- *Ranking:* Ranking of search results in the result list is the real art of search engines. For example, Google considers over 300 features in their ranking algorithm to determine the relevance of documents with respect to the search query and the user context (e.g. location, previous searches, social networking profile). When so many aspects are taken into account, simulation unification could provide a conceptual framework for calculating the similarity between search results and the query and user context.

6 Conclusion

Thias article has first recalled what led its authors to develop simulation unification for querying semistructured data on the Web. Simulation unification has been presenteed in 2002 at the 18th International Conference on Logic Programming (ICLP 2002) [15], in the long version [16] of that article, and in more detail in the doctoral thesis [30].

This article then has given a brief reminder of simulation unification as presented in the afore mentioned ICLP 2002 article.

Finally, this article has suggested novel directions for Web and Semantic Web research building upon the idea of simulation unification and generalising it in various manners.

Generalising simulation unification as suggested in this article would anchor logic programming in promising fields of research of considerable practical importance: Querying and Web search for multimedia and semantic data.

Acknowledgements

The authors are thanksful to the program comittee of the 28th International Conference on Logic Programming (ICLP 2012) and to its chairs, Agostino Dovier and Vitor Vitor Santos Costa, for having selected their article [15] amongst the mot influencial logic programming publications of the last decade and for their invitation to present the content of the present article at ICLP 2012.

References

1 Hassan Aït-Kaci, Andreas Podelski, and Seth Copen Goldstein. Order-Sorted Theory Unification. Technical report, digital – Paris Research Laboratory, 1993.

2 Franz Baader and Wayne Snyder. Unification theory. In Alan Robinson and Andrei Voronkov, editors, *Handbook of Automated Reasoning*. Elsevier Science Publishers, 1999.

3 Sacha Berger, François Bry, Oliver Bolzer, Tim Furche, Sebastian Schaffert, and Christoph Wieser. Xcerpt and visXcerpt: Twin Query Languages for the Semantic Web (Demonstration). In *Proceedings of 3rd International Semantic Web Conference (ISWC)*, 2004. `http://pms.ifi.lmu.de/publikationen/PMS-FB/PMS-FB-2004-23.pdf`.

4 Sacha Berger, François Bry, Oliver Bolzer, Tim Furche, Sebastian Schaffert, and Christoph Wieser. Querying the standard and Semantic Web using Xcerpt and visXcerpt (Demonstration). In *Proceedings of the 2nd European Semantic Web Conference (ESWC)*, 2005.

5 Sacha Berger, François Bry, Tim Furche, Benedikt Linse, and Andreas Schröder. Beyond XML and RDF: The Versatile Web Query Language Xcerpt (Poster Paper). In *Proceedings of 15th International World Wide Web Conference (WWW)*, pages 1053–1054, 2006. `http://pms.ifi.lmu.de/publikationen/PMS-FB/PMS-FB-2006-21/PMS-FB-2006-21.pdf`.

6 Sacha Berger, François Bry, and Sebastian Schaffert. A Visual Language for Web Querying and Reasoning. In *Proceedings of Workshop on Principles and Practice of Semantic Web Reasoning (PPSWR)*, number 2901 in LNCS. Springer, 2003. `http://pms.ifi.lmu.de/publikationen/PMS-FB/PMS-FB-2003-6.pdf`.

7 Sacha Berger, François Bry, Sebastian Schaffert, and Christoph Wieser. Xcerpt and visXcerpt: From Pattern-Based to Visual Querying of XML and Semistructured Data (Demonstration). In *Proceedings of 29th Intl. Conference on Very Large Data Bases (VLDB)*. `http://pms.ifi.lmu.de/publikationen/PMS-FB/PMS-FB-2003-2.pdf`.

8 Tim Berners-Lee. Linked Data, 2006.

9 Chris Bizer, Anja Jentzsch, and Richard Cyganiak. State of the lod cloud. Technical report, Freie Universität Berlin / DERI, NUI Galway, http://www4.wiwiss.fu-berlin.de/lodcloud/state/, September 2011.

10 Scott Boag, Don Chamberlin, Mary F. Fernández, Daniela Florescu, Jonathan Robie, and Jérôme Siméon. *XQuery 1.0: An XML Query Language (Second Edition)*. W3C Recommendation, December 2010. `http://http://www.w3.org/TR/xquery/`.

11 Oliver Bolzer, François Bry, Tim Furche, Sebastian Kraus, and Sebastian Schaffert. Development of Use Cases, Part I – Illustrating the Functionality of a Versatile Web Query Language. Technical report, Institute for Informatics, Ludwig-Maximilian University of Munich, 2004. `http:/pms.ifi.lmu.de/publikationen/PMS-FB/PMS-FB-2005-23.pdf`.

12 François Bry, Tim Furch, and Benedikt Linse. Simulation Subsumption or Déjà vu on the Web. In Diego Calvanese and Georg Lausen, editors, *Proceedings of the 2nd International Conference on Web Reasoning and Rule Systems (RR)*, number 5341 in LNCS, pages 28–42. Springer, 2008. `http://pms.ifi.lmu.de/publikationen/PMS-FB/PMS-FB-2008-8/PMS-FB-2008-8-paper.pdf`.

13 François Bry, Paula-Lavinia Pătrânjan, and Sebastian Schaffert. Poster Presentation: Xcerpt and XChange: Logic Programming Languages for Querying and Evolution on the Web. In *Proceedings of the 20th International Conference on Logic Programming (ICLP)*, number 3132 in LNCS. Springer, 2004. `http://pms.ifi.lmu.de/publikationen/PMS-FB/PMS-FB-2004-11.pdf`.

14 François Bry and Sebastian Schaffert. A Gentle Introduction into Xcerpt, a Rule-based Query and Transformation Language for XML. In *Proceedings of International Workshop on Rule Markup Languages for Business Rules on the Semantic Web (RuleML)*, 2002. `http:/pms.ifi.lmu.de/publikationen/PMS-FB/PMS-FB-2002-11.pdf`.

15 François Bry and Sebastian Schaffert. Towards a Declarative Query and Transformation Language for XML and Semistructured Data: Simulation Unification. In Peter J. Stuckey, editor, *Proceedings of the 18th International Conference on Logic Programming (ICLP)*, number 2401 in LNCS, pages 255–270. Springer, 2002. Short version of [16], `http://pms.ifi.lmu.de/publikationen/PMS-FB/PMS-FB-2002-2_short.pdf`.

16 François Bry and Sebastian Schaffert. Towards a Declarative Query and Transformation Language for XML and Semistructured Data: Simulation Unification. Research Report PMS-FB-2002-2, Institute for Informatics, Ludwig-Maximilian University of Munich, 2002. Long version of [15], `http:/pms.ifi.lmu.de/publikationen/PMS-FB/PMS-FB-2002-2.pdf`.

17 François Bry and Sebastian Schaffert. An Entailment Relation for Reasoning on the Web. In *Proceedings of Rules and Rule Markup Languages for the Semantic Web (RuleML)*. Springer, 2003. `http://pms.ifi.lmu.de/publikationen/PMS-FB/PMS-FB-2003-5.pdf`.

18 François Bry, Sebastian Schaffert, and Andreas Schröder. A contribution to the Semantics of Xcerpt, a Web Query and Transformation Language. In *Applications of Declarative Programming and Knowledge Management – Proceedings of 15th International Conference on Applications of Declarative Programming and Knowledge Management and 18th Workshop on Logic Programming (INAP/WLP)*. `http:/pms.ifi.lmu.de/publikationen/PMS-FB/PMS-FB-2004-5.pdf`.

19 Peter Buneman, Mary Fernandez, and Dan Suciu. UnQL: A Query Language and Algebra for Semistructured Data Based on Structural Recursion. *VLDB Journal*, 9(1):76–110, 2000.

20 Donald D. Chamberlin and Raymond F. Boyce. SEQUEL: A Structured English Query Language. In *Proceedings of the SIGMOD Workshop (SIGMOD)*, volume 1. ACM.

21 James Clark and Steve DeRose. *XML Path Language (XPath) Version 1.0*. W3C Recommendation, November 1999. `http://www.w3.org/TR/xpath/`.

22 Alin Deutsch, Mary Fernandez, Daniela Florescu, Alon Levy, and Dan Suciu. *XML-QL: A Query Language for XML*. W§C Submission, 1998.

23 François Fages. Associative-commutative Unification. In R. E. Shostak, editor, *Proceedings of the Seventh International Conference on Automated Deduction (Napa, CA)*, volume 170, pages 194–208, Berlin, 1984. Springer-Verlag.

24 Olaf Hartig, Christian Bizer, and Johann-Christoph Freytag. Executing SPARQL Queries over the Web of Linked Data. In *Proc. 8th International Semantic Web Conference (ESWC20099)*, Washington DC, USA, 2009.

25 Tom Heath and Christian Bizer. *Linked Data: Evolving the Web into a Global Data Space. Synthesis Lectures on the Semantic Web: Theory and Technology, 1:1*. Morgan & Claypool, 1st edition, 2011.

26 Monika R. Henzinger, Thomas A. Henzinger, and Peter W. Kopke. Computing Simulations on Finite and Infinite Graphs. Technical report, Computer Science Department, Cornell University, July 1996.

27 Robin Milner. An Algebraic Definition of Simulation between Program. Technical Report CS-205, Computer Science Department, Stanford University, 1971. Stanford Aritifical Intelligence Project, Memo AIM-142.

28 Dan Olteanu, Holger Meuss, Tim Furche, and François Bry. Xpath: Looking forward. In *XML-Based Data Management and Multimedia Engineering – Proceedings of the EDBT EDBT 2002 Workshops XMLDM, MDDE, and YRWS, Revised Papers*, volume 2490 of *Lecture Notes in Computer Science*, pages 109–127. Springer, 2002. `http://pms.ifi.lmu.de/publikationen/PMS-FB/PMS-FB-2002-4.pdf`.

29 Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

30 Sebastian Schaffert. *Xcerpt: A Rule-Based Query and Transformation Language for the Web*. Doctoral dissertation, Institute for Informatics, Ludwig-Maximilian University of Munich, October 2004. `http://pms.ifi.lmu.de/publikationen/dissertationen/PMS-DISS-2004-1/Schaffert_Sebastian.pdf`.

31 Sebastian Schaffert and François Bry. Querying the Web Reconsidered: A Practical Introduction to Xcerpt. In *Proceedings of Extreme Markup Languages 2004*, 2004. `http:/pms.ifi.lmu.de/publikationen/PMS-FB/PMS-FB-2004-7.pdf`.

32 Sebastian Schaffert, Thomas Kurz, Dietmar Glachs, Christoph Bauer, Fabian Dorschel, and Manuel Fernandez. The linked media framework: Integrating and interlinking enterprise media content and data. In *Proceedings of I-Semantics 2012*, 2012.

33 Pavel Shvaiko and Jérôme Euzenat. Ontology matching: state of the art and future challenges. In *IEEE Transactions on knowledge and data engineering*, 2012.

34 Gert Smolka. Feature Constraint Logics for Unification Grammars. *Journal of Logic Programming*, 12:51–87, 1992.

35 W3 Consortium. *OWL Web Ontology Language*, February 2004. W3C Recommendation, `http://www.w3.org/TR/owl-ref/`.

36 W3 Consortium. *RDF Vocabulary Description Language 1.0: RDF Schema*, February 2004. W3C Recommendation, `http://www.w3.org/TR/rdf-schema/`.

37 W3 Consortium. *Resource Description Framework*, February 2004. W3C Recommendation, `http://www.w3.org/TR/rdf-primer/`.

38 W3 Consortium. *SPARQL 1.1 Federation Extensions (W3C Working Draft)*, June 2010. `http://www.w3.org/TR/sparql11-federated-query/`.

39 W3 Consortium. *SPARQL 1.1 Query (W3C Working Draft)*, May 2011. `http://www.w3.org/TR/sparql11-query/`.