

The Web Within: Leveraging Web Standards and Graph Analysis to Enable Application-Level Integration of Institutional Data

Luiz Gomes Jr.^(✉) and André Santanchè

Institute of Computing, University of Campinas (UNICAMP),
Campinas, SP 13083-852, Brazil
{gomesjr,santanche}@ic.unicamp.br
<http://www.ic.unicamp.br>

Abstract. The expansion of the Web and of our capacity of producing and storing information have had a profound impact on the way we organize, manipulate and share data. We have seen an increased specialization of database back-ends and data models to respond to modern application needs: text indexing engines organize unstructured data, standards and models were created to support the Semantic Web, Big Data requirements stimulated an explosion of data representation and manipulation models. This complex and heterogeneous environment demands unified strategies that enable data integration and, especially, cross-application, expressive querying.

Here we present a new approach for the integration of structured and unstructured data within organizations. Our solution is based on the Complex Data Management System (CDMS), a system being developed to handle data typical of complex networks. The CDMS enables a relationship-centric interaction with data that brings many advantages to the institutional data integration scenario, allowing applications to rely on common models for data querying and manipulation.

In our framework, diverse data models are integrated in a unifying RDF graph. A novel query model allows the combination of concepts from information retrieval, databases, and complex networks into a declarative query language that extends SPARQL. This query language enables flexible correlation queries over the unified data, enabling support for a wide range of applications such as CMSs, recommendation systems, social networks, etc. We also introduce *Mappers*, a data management mechanism that simplifies the integration of heterogeneous data and that is integrated in the query language for further flexibility. Experimental results from real data demonstrate the viability of our approach.

Keywords: Query model integration · Data integration · DB/IR Integration · Graph data models · Graph query languages · Complex data

1 Introduction

Digital data availability has grown to unprecedented levels and surpassed our capacity of storage and analysis. This has led to the Big Data and NoSQL

movements, aiming at tackling the increasing demands for scalability. A parallel development regards the proportional increase in data complexity. Capturing and processing greater amounts of data produces information that is correlated in diverse and intricate ways. Furthermore, recent developments in processing power, modeling, and algorithms enable the implementation of systems that better explore the increased complexity of data. All these factors influenced and enabled the dissemination of social networks and initiatives like the Linked Open Data¹.

Realizing the potential of the relationships inside the interconnected data and developing the means for their analysis fueled the development of the areas of complex networks [11] and link mining [14]. Related techniques have been applied in several scenarios, such as systems biology, neuroscience, communication, transportation, power grids, and economics [10].

In this article we aim to show how the focus on relationship analysis is important for institutional data and applications. Assessing properties of how data is correlated is the basis for several tasks, such as document retrieval, item recommendation and entity classification. We, therefore, advocate the vision that institutional data can be seen as a big complex network and, most importantly, several modern and commonplace applications can be specified in terms of link analysis tasks. A unified, relationship-centric framework for data and applications can enable a new level of integration, encompassing data from diverse sources (e.g. structured and unstructured) and applications (e.g. information retrieval, machine learning, data mining). This application-level integration allows developers to rely on common models for data interaction, simplifying development of applications with information needs that span multiple querying paradigms.

Institutional data and applications have, however, several requirements that make it hard to apply link mining techniques directly. The size of the data, its dynamic nature, and heterogeneity are not considered in traditional approaches. The focus of complex network techniques is typically on homogeneous networks with a single type of relationship (e.g. social networks), employing off-line algorithms to assess snapshots of the data. Modern applications, on the other hand, favor online access to subsets of the data, and must handle data heterogeneity seamlessly.

Here we introduce the Complex Data Management System (CDMS), which aims at providing query-based interaction for complex data. The proposed query model allows users to specify information needs related to the topology of the correlations among data. It also offers management mechanisms that are more adequate to the increased importance of the relationships in the data.

The increased expressiveness in the new framework provides a better match to the requirements in the described institutional settings. The online query mechanism allows the composition of queries that explore diverse aspects of how data is correlated. This not only allows the same query model to be used in diverse application scenarios, but also allows queries that encompass concepts

¹ <http://www.w3.org/standards/semanticweb/data>.

from multiple paradigms and data sources, such as in queries like “retrieve documents related to the keyword query ‘US elections’ and the topic *politics*, written by democrat journalists, ranked by relevance to the keyword query and reputation of the author”.

To enable this type of interaction, the underlying institutional data must be integrated. Here we describe how we are fostering web standards to enable the required integration. Once the data is integrated, the CDMS is used to provide the proposed querying infrastructure. To tackle the integration of data models, we employ an RDF graph that interconnects data from diverse sources and models. The flexibility of graph models allows easy mapping from otherwise incompatible models (e.g. unstructured text and structured databases). Figure 1 contextualizes the elements in our proposal: several data sources are integrated in a unifying graph, which allows our framework to enable a more expressive interaction between users and data.

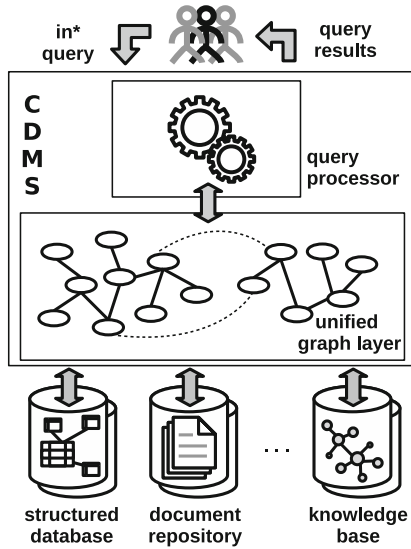


Fig. 1. Architecture of a CDMS deployed in a data integration scenario

As for integration at the query and application level, we acknowledge the importance of the Information Retrieval (IR) and Databases (DB) fields – which dominate data-driven applications in current settings – and describe how our new query model, which leverages complex network analysis, unifies concepts from these areas. To enable our query model over the unifying graph, we reinterpret querying concepts from diverse areas into graph analysis tasks. We implement this model in a new query language called *in** (in star), which is an extension grammar for existing languages such as SPARQL.

We also address architectural issues related to the integration process, introducing the concept of *mappers*, which aim at simplifying relationship management. Mappers are similar to *stored procedures* in databases, triggered when nodes are created to carry customized tasks such as adding appropriate relationships or even other new nodes. Our *mappers* are integrated in the query model for further flexibility.

We aim at contributing towards a more unified and expressive interaction between users and data through this relationship-centric querying and data management framework. Experiments with real data are presented to demonstrate the expressiveness and practicability of our framework.

This paper is organized as follows: Section 2 discusses the new challenges for the current heterogeneous technological landscape. Section 3 introduces the Complex Data Management System, which is the basis for our integration approach. Section 4 describes the requirements for data access and model integration in our framework as well as issues related to query model integration, a fundamental concept in our proposal. Section 5 details our integrated query model and discusses usage scenarios. Section 6 introduces related data management issues and describes our *mapper* mechanism. Section 7 demonstrates experiments for our query language and the use of mappers in scenarios based on a large and interlinked database of movies. Section 8 contextualizes related work in respect to our proposal. Finally, Sect. 9 concludes the paper.

2 New Challenges for Institutional Data and Application Integration

The new scenario of overwhelming accumulation of information has a profound impact on the way we organize and manipulate data. We have seen an increased specialization of database back-ends and data models to respond to modern application needs: text indexing engines organize data on the Web, standards and models were created to support the Semantic Web, Big Data requirements stimulated an explosion of data representation and manipulation models labeled under the NoSQL umbrella. This complex and heterogeneous environment demands unified strategies that enable data integration and, more importantly, cross-application, expressive querying.

Although data integration has been an active research topic for many decades, most proposals depart from environments that do not take into account the modern diversity of technological infrastructures. Federated databases, for example, usually adopt the relational model to integrate data sources, with limited capabilities when dealing with semi or unstructured data. Similarly, in typical OLAP implementations, the benefits of integration are restricted by the adopted query model: data analysts may answer complex questions, but there is no direct benefit to other applications inside the institution. For example, Web developers cannot leverage the potential of the integration in their implementations of recommendation systems because they typically work on very different query models. Similar issues also appear in other contexts, such as the Semantic Web,

which brings great benefits for data integration but querying capabilities do not match the diversity of Web applications.

A level of integration that covers a wide range of data models and, more importantly, data query models would not only allow applications to incorporate more relevant information, but would also allow more expressive queries that combine elements from different querying paradigms. For example, consider the following queries:

- retrieve documents related to the keyword query “US elections” and the topic *politics*, written by democrat journalists, ranked by relevance to the keyword query and reputation of the author;
- retrieve employees relevant to a given project ranked by their reputation among peers;
- retrieve profiles of people over 30 years old, ranked by similarity of hobbies on their profiles to hobbies on my own;
- retrieve products not yet purchased by the client Bob that are relevant to him.

These queries cover a broad range of data models (e.g. unstructured documents, relational, graph) and applications (CMSs, social networks, recommendation systems). The queries also combine concepts from diverse query models, such as relational predicates, keywords, ranking, and metrics of relevance and reputation. These and similar queries show up in many situations in typical institutions, both for internal, administrative purposes or for Web applications developed for external use. Answering these queries in current infrastructures typically demands substantial amount of resources and engineering to design ad-hoc subsystems.

To provide an overarching approach for querying, data model integration and query model integration must be tackled simultaneously. Querying is especially challenging, given the diversity of the data and the complexity of the information needs. The central observation underlying this article is that these issues can be mapped into complex network analysis tasks. Several tasks typically associated with the information retrieval and machine learning fields – including document retrieval, recommendation, and classification – draw inferences from how information pieces are correlated. Even though the correlations are often not explicit, it is intuitive to consider the data as a graph and notice the importance of the relationships and the underlying topology for each task. Our hypothesis is that an expressive query model that can capture topological properties in query time can be used to integrate these information needs in a single conceptual framework. We aim to show how the CDMS can be used in these scenarios, providing expressive querying and data management mechanisms that are appropriate to the heightened importance of relationships in the described scenarios.

3 Complex Data Management

The database framework used in our proposal is being developed to tackle issues associated with Complex Networks. In a complex network [11], the patterns

defined by the interconnections are non-trivial, deviating substantially from cases where connections have the same probability (e.g. lattices or random graphs). The techniques developed for complex network analysis have become important resources in diverse fields such as systems biology, neuroscience, communication, transportation, power grids, and economics [10]. These areas deal with complex structures that requires specific techniques for analysis. In all cases, relationship analysis is a major aspect for knowledge acquisition. Typically, these structures generate emergent behavior, which are determined by the complex interactions among their simple constituent elements.

As a result of increased capacity of data storage and processing, these scenarios have come forth in other areas, such as enterprise data management, our focus on this paper. A typical institution nowadays stores and processes many textual documents alongside traditional structured data, communication and transaction records, and fast changing data about market and competition. These data are highly interlinked, by design or through intricate (and potentially imprecise) data analysis procedures such as named entity recognition, sentiment analysis, and recommendation systems.

Our CDMS is aimed at enabling querying and management of what we define as *complex data*. Complex data is characterized when relationships are central to data analysis. In these cases, the graph formed by data entities (nodes) and relationships (links) present properties typical of complex networks. The CDMS is aimed at providing adequate support for handling and querying complex data. It differs from typical DBMSs in four main aspects: (i) data model, (ii) query language, (iii) query evaluation, and (iv) data management mechanisms. Each of these items is described below.

- **Data model:** The data in target CDMS applications typically do not comply to pre-defined schemas. The high number and diversity of the relationships require a model where relationships are first-class citizens. Graph models are obvious choices in these settings. Their flexible modeling characteristics enable easy mapping of most types of data. Nodes with immediate access to neighbors is also an important feature for the type of computation involved. The CDMS framework adopts weighted edge-labeled property multigraphs to encode complex data. In this article, we leverage the RDF model to integrate institutional data.
- **Query language:** Our CDMS query language is intended to be flexible enough to allow correlation of data when little is known about how they are linked and organized. We developed a declarative query language that extends existing graph languages by introducing ranking based on a set of flexible correlation metrics. The ranking metrics proposed are: relevance, connectivity, reputation, influence, similarity, and context. The proposed language is designed as an extension for existing graph languages. In this article we show how SPARQL can be extended to enable the new query model.
- **Query evaluation:** Our abstractions for query evaluation fully support the query language while allowing for under-the-hood optimizations. We adopt a variation of the spreading activation (SA) model as our main abstraction for

query evaluation. The model allows the specification of the ranking metrics that are the basis of our query language. The SA mechanism is based on traversing the network from a initial set of nodes, activating new nodes until certain stop conditions are reached. By controlling several aspects related to this activation flow, it is possible to infer and quantify the relationships of the initial nodes to the reached ones.

- **Data management mechanisms:** Relationship creation is an important and defining operation for the described application scenarios. For example, several text indexing tasks, such as topic modeling, derive relationships between the text and more general concepts. In machine learning applications, elements are associated with features or classification categories, for example. In our framework, the creation of relationships is encapsulated in mappers. Mappers are very similar to *stored procedures*. What sets them apart are (i) their integrated use in our ranking queries, and (ii) how they are hooked in the databases’s API so that any new data that matches the mapping criterion is passed through appropriate mappers.

The CDMS offers an architecture where relationships are central elements of the database. It enables queries to tap into properties derived from topological characteristics of the underlying graph. CDMS’s new query model and management mechanisms allow for new levels of expressiveness for several tasks, simplifying integration of data from diverse sources and allowing distinct applications to employ the same query model over the integrated data.

4 Data and Query Model Integration

The level of integration that we aim at requires solutions to three main issues: (i) unified data access, so that queries have access to all data, (ii) unified data model, so that queries can reference data from diverse formats; and (iii) unified query model, so that applications can have a single interface for interaction with data. This level of integration allows applications to be based on the same underlying models to interact with data, what we call application-level integration.

4.1 The Local Unified Graph

Institutions face similar challenges to that of the Web: data produced by diverse groups in distinct contexts must be integrated to allow for more capable and outreaching applications. Although several research and products were developed to address these issues, we argue that revisiting this problem through the perspective of the new developments in applications and standards of the Web would allow for a more adequate interaction with modern institutional data.

The Semantic Web initiative has advertised the benefits of treating the Web as an integrated Giant Global Graph (GGG) [5]. Similar benefits could be achieved inside institutions by integrating all their data in a Large Local Graph (LLG). A LLG lacks the diversity and magnitude of the GGG, but it

allows higher levels of control over data and local processing power, enabling better semantic integration among distinct data sources and more expressive querying. Another advantage of creating LLGs is that it facilitates transference of information to and from the GGG.

The framework proposed here assumes an underlying LLG. Although our solutions have interesting applications also in the context of the Semantic Web, we require levels of integration and processing power that are not currently available for the GGG. We, therefore, focus on institutional data but expect that in the future technological advances would allow similar interactions in a broader context.

A LLG is meant to integrate a broad range of data from an institution. Aggregation of external data from the GGG would also be important in many scenarios. Integrating data across domains and models is important to allow rich correlation queries between diverse data elements. The graph model is suiting for this scenario. Its simplicity and flexibility allows the representation of most of the popular data models [3, 6]. Figure 2 shows a graph containing data derived from documents and relational databases (more details on the mapping in Sect. 4.2).

Here we employ the RDF(S) model for the LLG for several reasons: it is a stable and popular model, it implements a flexible graph model, classes facilitate the mapping of other models (e.g. object, relational), integration with other standards (e.g. URI, XML), standardized query language (SPARQL), simplified data sharing, etc.

It is important to emphasize that the strategy to create the unified graph is environment-specific. Although we provide general guidelines on how data should be represented as nodes and edges, our framework assumes the data are converted and interlinked in a coherent graph. What we want to show in this paper, and our main contribution, is that popular query models can also be translated into graph concepts, employing graph analysis in query processing. To take full advantage of the model, users should be aware of the semantics of the elements composing the graph. In that regard, our strategy is similar to an OLAP environment, in which the query model assumes data are integrated in a multidimensional schema – according to whichever strategy is adequate for the specific environment.

4.2 Data Model Integration

There are several alternatives for mapping a given data model into graphs. Although our framework works independently of the strategy adopted, we provide guidelines on basic transformations of typical models.

Here we focus on the integration of text documents and the relational model. The mapping for other models, such as semi-structured or NoSQL variations, can be derived by similar approaches. There are several alternatives for mapping a relational scheme to an RDF graph [3, 6]. There is even a W3C working group² to define standards for these mapping languages. Here, to simplify the

² <http://www.w3.org/2001/sw/rdb2rdf/>.

discussion, we assume that (i) table descriptions become RDF classes, (ii) rows become instances of their respective tables, with their primary keys as identifiers, (iii) columns become properties of the instances, with values corresponding to literals and foreign keys becoming explicit links to other instances.

Graph representation of documents for IR purposes is also possible. An inverted index (in the bag of words model) can be readily mapped into a graph that connects terms and documents. More modern schemes to index documents such as topic models [8] and explicit semantic analysis [25] also fit nicely into this strategy, bringing the benefits of reduced dimensionality (i.e. avoiding creating an unnecessarily large graph containing entire postings list), less semantic ambiguity, and more cognitive appeal.

In our framework, a keyword query is also represented as a (temporary) node in the graph. The same indexing strategy used for the stored documents is applied to generate the relationships of the query node (Fig. 2). This graph representation of keyword queries allows them to be expressed alongside structured predicates in the queries (Sect. 5.3).

To simplify data management in the complex integrated graphs, our framework introduces mappers. Mappers play an important role in data model integration, being the mechanism that encapsulates the creation of relationships between elements of the graph.

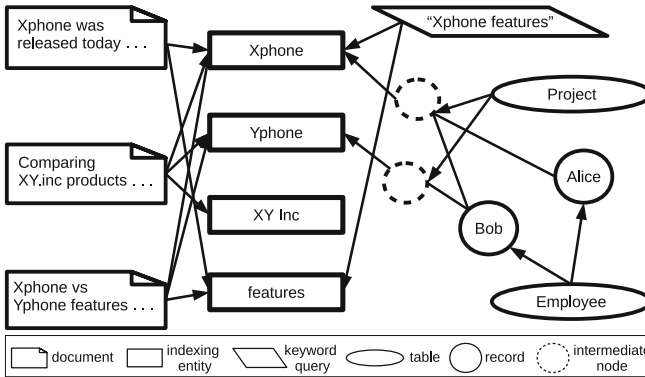


Fig. 2. Data elements represented as a unified graph

Figure 2 shows a simplified example to illustrate diverse elements represented as a unified graph. News articles about products are mapped into entities according to mappers that implement an indexing/annotation technique (e.g. topic modeling, named entity recognition, etc.). A keyword query is likewise mapped into these entities, using the same mapper in query time. Relational data from tables (Project, Employee) are also mapped into nodes in the graph and also connected to the entities. More details on the use of mappers to bridge data models are presented in Sect. 6.

4.3 Query Model Integration

Data access and model integration brings many benefits to institutions, providing a unified path for interaction with data. This interaction is, however, usually constrained by the data model and the query language employed for the integration. For example, in a typical OLAP setting, data are integrated in a data warehouse, but no direct benefit is gained by applications such as institutional search engines. The problem is that there is a conceptual gap between the interaction language in the integration infrastructure (OLAP) and the languages used by the applications (keyword queries, SQL, etc.).

Our query model, on the other hand, is built on the assumption that integration should begin at the query or application level. The goal is to specify a query model that can express concepts from diverse interaction models in a unified and intuitive way. We focus on the applications related to the areas of databases, information retrieval, and complex networks (Fig. 3). Our model can also be used in machine learning tasks, as discussed in [16].

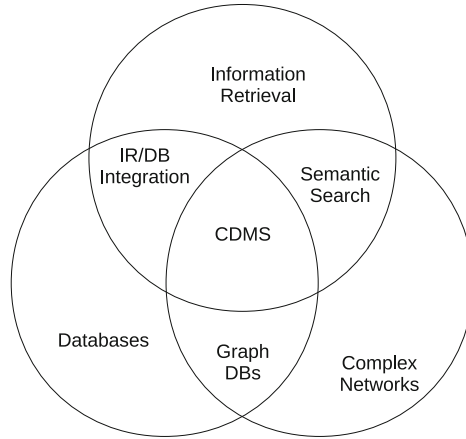


Fig. 3. CDMS in the intersection of multiple areas

The two main groups of models for data driven applications today are those associated with Information Retrieval and Database Systems. It is natural that these two areas attained such distinction over the last decades. They together cover a broad range of the data structuring spectrum – from unstructured data in documents to structured data in relations. Typical applications in IR include search engines, recommendation systems, social networks etc. Applications taking advantage of DBMSs are ubiquitous, being through traditional relational databases or the more recent models for document databases, XML and semi-structured databases, graph databases and the NoSQL movement.

Complex networks, which have gained strong momentum in the last decade, is the third area completing our picture. Complex networks, whose techniques

are often applied in typical IR and DB tasks, is an important area to cover in an integration framework. More importantly, we consider complex networks a fundamental piece to establish the basis of the integrated framework. To specify a query language that could be used in such a diverse scenario it is important to unify characteristics of the different interaction models.

Keyword queries and ranking are important concepts from IR, as other integration approaches have identified [2, 9, 33]. Significant research efforts have been dedicated to enable efficient ranking and keyword queries in a wider range of data models (e.g. relational, XML). In databases, declarative languages offer effective means for online interaction with data. Furthermore, the declarative approach offers opportunities for transparent query optimization. Complex networks offer a range of techniques to assess important characteristics of the data based on the underlying connections. These techniques are employed in diverse scenarios, such as the use of relevance metrics (e.g. HITS, PageRank) for IR purposes.

Here we defined a query model that embodies characteristics from all the discussed areas, providing a declarative query language that can express structured predicates, keyword queries, network topology-aware metrics, and compose results (optionally) as ranked lists. The challenge is to enable all these features over the unified graph model (LLG) presented.

Declarative querying and traditional database concepts like selections, projections and aggregations are already provided by RDF query languages such as SPARQL. The remaining issues are related to enabling IR-like ranking metrics that now have to be reinterpreted in an RDF graph setting. To enable this extended querying mechanism, we reinterpret this topology-aware metric in a common graph processing model that we call Targeted Spreading Activation (TSA), described in the following section.

5 Ranking Metrics and Language Integration

Correlating data is an important and defining characteristic for many of the applications we want to cover. To enable a high level of flexibility for correlations, we specify a set of ranking metrics which are influenced by information retrieval applications and complex networks concepts. The selection of the specific metrics aims at covering a wide range of applications while also being simple to use and understand. In the process of defining these metrics, we started with some popular metrics used in IR and then expanded the set according to the applications we wanted to cover. The set of metrics we define can be organized in the taxonomy presented in Fig. 4.

The basis of our taxonomy is the concept of comparison. Our metrics are meant to compare elements in the graph and generate a score that represents the strength of the association. The peculiar aspect about our metrics is that the scores are generated based on analysis of the topology of the graph, in contrast to most ranking approaches that are based on attributes of the elements.

There are two main groups of comparisons. Set comparisons corresponds to comparisons among elements from a finite set. Reputation and Influence are the

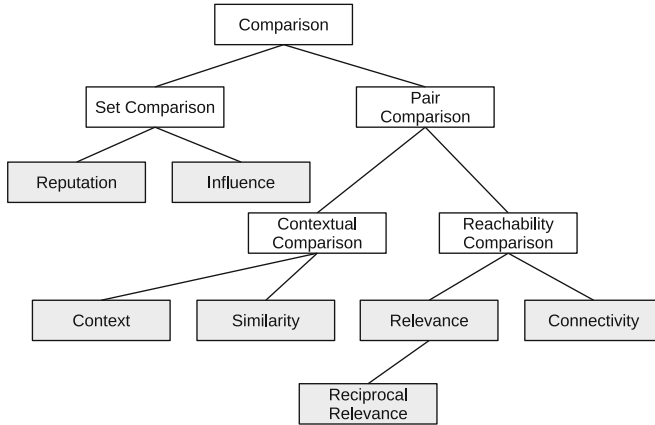


Fig. 4. Taxonomy for the adopted ranking metrics

metrics in this category. They assess, using different strategies, how well a node performs as a hub for information. The definitions of the metrics, as well as details on their interpretation, are presented in the next section.

Pair comparisons are applied to individual pairs of nodes. They assess properties of the topology surrounding or connecting the two nodes. The similarity and context metrics, classified under contextual comparison, assess the commonalities in respect to elements (nodes or relationships) surrounding the comparing nodes. Relevance and connectivity, classified under reachability comparison, assess properties of the paths interconnecting the comparing nodes.

As far as we know, this is the first time that these metrics are considered and defined under the same conceptual framework. These metrics express cognitive processes or patterns that we use to assess correlation of entities in the real world, and which are the basis of many data-driven applications, as we intend to portray along the text. We now describe our metrics and define them from a graph analysis perspective.

5.1 Graph Interpretation of the Metrics

The translation of the ranking metrics to the unified graph strategy is a challenging task. Here we adopt a Spreading Activation (SA) [12] model for our novel interpretation of the metrics.

The Targeted Spreading Activation Model: Spreading Activation (SA) processes [12] were developed to infer relationships among nodes in associative networks. The mechanism is based on traversing the network from an initial set of nodes, activating new nodes until certain stop conditions are reached. By controlling several aspects related to this activation flow, it is possible to infer and quantify the relationships of the initial nodes to the reached ones.

This simple model has the fundamental requirements for the type of correlations we want to provide for complex data:

- (i) it can derive correlations among any two sets of initial nodes and destination nodes; This is important to enable modeling of several correlation metrics, as described in Sect. 5.1.
- (ii) the final value of the correlations decreases as the length of the contributing paths grows; This reflects the intuitive perception that closer elements are more correlated. The model allows tuning of this characteristic through a parameter for potential degradation.
- (iii) the degradation of the potential imposes boundaries to query processing;
- (iv) it can be implemented as graph traversal patterns [27]; The processing of these patterns are centered in origin nodes, resulting in localized processing. The computation of these patterns requires less memory than global ranking metrics such as PageRank and HITS. This type of computation is supported by several graph database systems³.

We tweak the basic SA model by adding mechanisms to (i) adapt the process to the labeled graph model used, (ii) consider relationship weights, (iii) add a more strict and predictable termination condition, and (iv) make the process aware of the target elements. The last point is key to the semantics of the SA process for querying complex data and also to improve optimization opportunities. We named the proposed SA variation as Targeted Spreading Activation (TSA).

The TSA model used here is defined by the parameters G , N , I , O , a , t , d , c , l , and dir described, alongside other definitions, in Table 1. A TSA process starts with origin nodes initially activated with potential a . Output potentials for each subsequent node are calculated by the function O . The output potential is spread through all relationships whose labels are in l that follow directions in dir . The potential for the reached nodes is calculated by function I . For the next iteration, the potential is spread to subsequent nodes, restarting the process, as long as the potential for reached nodes is higher than t and the number of iterations is lower than c .

Although simple in its definition, this is a very expressive model to build flexible correlation metrics. By specifying appropriate parameters and combining subsequent executions of TSAs, it is possible to define metrics that encompass concepts like relevance, reputation and similarity (Sect. 5.1). These metrics can be integrated in a declarative language with applications to a wide range of modern querying scenarios (Sect. 5.3).

Being the core of the querying process mechanism, the TSA process becomes the main target for query optimization strategies. Like with any other data or processing model, the practicability of TSA-based querying depends on architectural mechanisms to support data access optimizations and heuristics to provide approximate answers. Optimization issues were addressed in [15].

³ A good overview of applications and systems can be found in <http://markorodriguez.com/2013/01/09/on-graph-computing/>.

Table 1. Notation used in the definitions

Notation	Description
$SA(N)$	a set of activated nodes after the execution of the spread activation process
$SA(M)_n$	value for the potential of node n after the execution of the spread activation with initial activated nodes M
a, t, d, c	respectively, initial activation potential, firing threshold, decay factor, maximum number of iterations (depth)
l	set of labels that determine valid nodes for traversal
dir	$dir \subset \{inbound, outbound\}$; set of directions for traversal
\overline{dir}	$dir \cap \{inbound, outbound\}$; reversed directions of dir
$\overline{SA(m)_n}$	same as $SA(m)_n$ with reversed directions, i.e. $dir \leftarrow \overline{dir}$
$I(n)$	function that calculates the input potential of a node. $I(n) = \sum_{m \in ant(n)} O(i) \text{ in the default case}$
$O(n)$	function that calculates the output potential of a node. $O(n) = I(n) * d \text{ in the default case}$
$ant(n)$	set of antecedent nodes, i.e. nodes linked to n through relationships in l that follow the directions in dir
$sub(n)$	set of subsequent nodes, i.e. nodes linked to n through relationships in l that follow the directions in \overline{dir}
$p(SA(N))$	set of activation paths (for each node in $SA(N)$)
$ S $	number of elements in set S

IR Metrics According to the TSA Model: In the TSA model, to assess the rank of the relationship of nodes according to a metric, an activation potential is placed at the target elements defined in the query. The potential is spread across the topology of the graph, losing or gaining strength based on the IR metric, length of the path, or properties of the traversed elements. The metric-specific definitions of the TSA processes are presented below.

Definition 1. $relevance(m, n) = SA(\{m\})_n$,

with $O(n) = \frac{I(n) * d}{|sub(n)|}$

Relevance between two nodes is a measure that encompasses correlation and specificity. Correlation is proportional to the number of paths linking the two nodes and inversely proportional to the length of the paths. Specificity favors more discriminative paths (i.e. paths with fewer ramifications. It is easy to observe that this definition resembles the definition of relevance between queries and documents in a information retrieval setting. Traditional tf or idf term weighting can be readily emulated in our scheme when terms, queries and documents become nodes of a graph. Our definition is, however, a generalization of the concept that can be applied to any type of graph data and with any number or type of relationships in between m and n .

Definition 2. $rrelevance(m, n) = SA(\{m\})_n + \overline{SA(\{n\})_m}$,
 with $O(n) = \frac{I(n) * d}{|sub(n)|}$

Reciprocal Relevance (RRelevance) between two nodes aggregates the relevance in both directions. In an information retrieval setting, it would be equivalent to aggregating tf and idf in the same metric.

Definition 3. $connectivity(m, n) = SA(\{m\})_n$

Connectivity between two nodes is a measure that assesses how interconnected two nodes are. The score is proportional to the number of paths linking the nodes in the network activated by the SA algorithm.

Definition 4. $reputation(n, N) = SA(N)_n$

Reputation of a node measures how effective it is as a hub for information flow. Here the nodes of interest are activated at the beginning and the ranking scheme favors nodes that are revisited in the sequence of the SA process. This is a simple but convenient interpretation in scenarios where the reputation cannot be pre-calculated due to high update rates, variability in the types of relationships used for the queries, or need to bias the scores based on a set of initial nodes (as in [34]).

Definition 5. $influence(n) = |(SA(\{n\}))|$

Influence is a specialization of reputation where the only concern is the number of nodes reached from the origin. The topology of the graph – in/outdegree or cycles – do not influence the metric.

Definition 6. $similarity(m, n) = \frac{|p(SA(\{n\})) \cap p(SA(\{m\}))|}{|p(SA(\{n\})) \cup p(SA(\{m\}))|}$

Similarity measures the ratio of common relationships (same edge label linking common nodes) between two nodes.

Definition 7. $context(m, n) = \frac{|SA(\{n\}) \cap SA(\{m\})|}{|SA(\{n\}) \cup SA(\{m\})|}$

Context is a specialization of similarity where edge labels do not matter.

5.2 Semantics of Ranking Metrics in Queries

Having the ranking metrics interpreted as graph analysis tasks, there is now the need of integrating these metrics in a declarative language. As opposed to creating an entirely new query language, we decided to leverage existing languages by defining an extension language that can be integrated into other languages. To that extent, we first define the semantics of the intended integration.

In our model, the proposed ranking metrics are intended to be used with graph query languages that offer: (i) means to reference individual nodes in the graph, (ii) selection of match variables, and (iii) query results as a set of tuples (or a graph representation of). These are basic components of graph languages like SPARQL and Cypher. A ranking metric can refer to:

- a single match variable (set of vertices), e.g. “rank papers from EDBT 2013 according to first author reputation”, where first author is the match variable in question (e.g. “SELECT ?firstAuthor ...” in SPARQL);
- a given vertex⁴ and a match variable, e.g. “rank papers according to *relevance* of their first author (match variable) to the topic data integration (vertex)”;
- two match variables, e.g. “rank papers according to *relevance* of the first author to the topic in the first keyword of the paper”.

Conceptually, the ranking metrics are applied to query results, generating a ranking value for each returned tuple. In practice, to speed up query processing, results would be approximate and the rank would be generated for some of the nodes based on access pattern heuristics.

5.3 Extending Declarative Queries

Having the ranking metrics interpreted as graph analysis tasks, it is possible to integrate them in a declarative query language. As opposed to creating an entirely new query language, we decided to leverage existing languages by defining an extension language.

A convenient way to integrate the ranking metrics into existing query languages is to add a “RANK BY” clause. The clause should enable an arbitrary combination of metrics that expresses the global ranking condition defined by the user. We encode the clause in the extension query language that we denominated in* (or in star). in* can be used to extend other languages, for example, extended SPARQL becomes inSPARQL by convention. More details about the language and its design principles can be found in [17].

```

1 ExtendedQuery ::= RegularQuery RankClause
2 RankClause ::= 'RANK BY' RankMetric ( ',' RankMetric ) *
3 RankMetric ::= Weight? ( UnaryRMetricDesc | BinaryRMetricDesc )
4 UnaryRMetricDesc ::= UnaryRankMetric 'OF' MatchVariable
5     Modifiers *
6 BinaryRMetricDesc ::= BinaryRankMetric 'OF' MatchVariable 'TO'
7     ( MatchVariable | Vertex | KWQuery ) Modifiers *
8 UnaryRankMetric ::= Reputation | Influence
9 BinaryRankMetric ::= Relevance | Connectivity | Similarity | Context
10 Modifiers ::= Follow | Depth | Direction | Weighted
11 Follow ::= 'FOLLOW' EdgeSet
12 Depth ::= 'DEPTH' INTEGER
13 Direction ::= 'DIRECTION' ( 'INBOUND' | 'OUTBOUND' | 'BOTH' )
14 Weighted ::= 'WEIGHTED'
15 KWQuery ::= 'KWQUERY' '(' String ')'
16 EdgeSet ::= '(' Edge ( ',' Edge ) * ')' ...
```

Fig. 5. Simplified BNF grammar for the proposed extension (terminators omitted)

⁴ as defined previously, a keyword query would also be a node in the graph.

Note that the extension causes query semantics and result interpretation to change, therefore, any extended language would be more adequately described as new language based on the syntax of the original language. This suggests an incidental meaning for an acronym like inSPARQL: recursively, “inSPARQL is Not SPARQL”.

Figure 5 shows a simplified BNF grammar of the proposed extension. A ranking can be specified as mix of weighted ranking metrics (lines 2 and 3). Weights capture the relative importance of each metric. The scores generated by the metrics are normalized before the calculation of the final weighted score.

Ranking metrics are unary or binary. Unary ranking metrics are applied to a single match variable (lines 4 and 5). Binary ranking metrics can be applied to a match variable and a named vertex or between two match variables.

The language allows for modifiers (lines 10 to 14) to be applied to the ranking definitions. These modifiers define the parameters for the execution of the SA algorithm. FOLLOW specifies valid edges for the algorithm to traverse. DEPTH defines the maximum length for the traversal paths. DIRECTION sets the direction of traversal as outbound, inbound or both (default) edges. WEIGHTED makes edge weights influence the degradation of the activation potential (the potential is multiplied by the weight).

The combination of the IR-inspired metrics in a declarative querying setting enables a high level of flexibility and expressiveness for the applications to explore. In the next section we show and discuss some examples of queries that can be used for practical applications.

5.4 Applications

This section presents examples of queries in the extended SPARQL language. These queries are meant to demonstrate the expressiveness of the approach in a wide range of applications.

Search engines/CMSs: Figure 6a shows a possible implementation for a document retrieval query using topic modeling. The keyword query is expressed by the function KWQUERY⁵ and the relevance is assessed as if the query was a node in the graph. The query also takes into account the reputation of the authors and the relevance of documents to the topic :Politics (assessed based on the connections between the query node and documents that are created by a Topic Modeling algorithm such as LDA). Data management aspects discussed in the next section would be interesting matches to implement novel CMS architectures like in Ngomo et al. [26]. Our metrics would also allow query answering based on the context of the user or a context defined by the user, implementing a query model such as the one proposed by [28]. Graph-based term weighting [7] could also be simulated in our query model.

Recommendation systems: Figure 6b shows a product recommendation query that finds products that the client Bob (with uri :bob) has not purchased.

⁵ KWQUERY is a syntactical shortcut that represents an underlying mapper as in Sect. 6.1.

The query traverses Bob's friendship network to find products purchased by his friends that might be relevant to him. The spreading activation interpretation of this query evaluation also implies that products purchased by Bob, even though they do not appear in the results, will be traversed on the way to customers that have co-purchased these products, which in turn will activate other products from these customers.

Social Networks: Figure 6c shows a query that could be used for friend suggestion on a social network application. It ranks the top 5 persons over a given age based on the similarity of hobbies and movie preferences of user Alice.

Collaborative filtering: Figure 6d shows a query that filters posts from pages that friends of user Carol follow. The posts are ranked based on their influence in the network.

Decision support: Figure 6e shows a query that can be used to prospect for employees that would be good candidates to replace a manager (Charlie) in his post. The query favors employees strongly related to a (presumably important)

- a**

```
SELECT ?doc, ?author
WHERE { ?doc :type :BlogPost }
RANK BY (2 KW, 1 PoliticsTopic, 3 AuthorRep)
2 RELEVANCE OF ?doc TO KWQUERY("`US elections")
1 RELEVANCE OF ?doc TO :Politics
3 REPUTATION OF ?author
```
- b**

```
SELECT DISTINCT ?product
WHERE { ?product :type :Product .
FILTER NOT EXISTS (:bob :purchased ?product))}
RANK BY RELEVANCE OF ?product TO :bob
FOLLOW (:friendsWith, :purchased)
DEPTH 3 DIRECTION BOTH
```
- c**

```
SELECT ?person
WHERE { ?person :hasAge ?age .
FILTER (?age > 30)}
LIMIT 5
RANK BY SIMILARITY OF ?person TO :alice
FOLLOW (:hasHobby, :favoriteMovie)
```
- d**

```
SELECT ?post, ?page
WHERE { :carol :closeFriend ?closeFriend .
?closeFriend :follows ?page .
?page :publishes ?post .
?post :date ?date .
FILTER(?date = "2012-12-12"^^xsd:date)}
RANK BY INFLUENCE OF ?post
```
- e**

```
SELECT ?employee
WHERE { ?employee :worksFor :research}
RANK BY
2 RELEVANCE OF ?employee TO :yPhone
1 CONTEXT OF ?employee TO :Charlie
```

Fig. 6. Examples of extended SPARQL queries (namespaces have been omitted)

product (yPhone) and also those that have professional contexts similar to the current manager.

Other applications: Similar queries could be used in several other scenarios, especially the ones with richly interconnected data and that require complex analysis of the correlations. Some examples are Semantic Web inference applications, where assessing correlations between classes and candidate instances can be complex [1]. The scientific domain is another interesting application field. For example, in a database with food network relationships, a query could identify relevant species or areas for conservation efforts.

6 Relationship Management in the CDMS

We have so far discussed our data and query models, with little focus on implementation or architectural aspects. The proposed query model implies new requirements for user interaction, query processing and data management. The CDMS is responsible for encompassing all these aspects in a coherent architecture.

The querying mechanism presented so far is based on the observation that relationship analysis is central to several applications and the basis for evaluating the metrics introduced here. Besides providing a query language that enables expressive correlation clauses, it is important to provide the CDMS with mechanisms to manage diverse aspects of relationship life cycle. Here we show how such mechanisms could provide better support for data integration tasks and increase the expressiveness of the query language.

6.1 Mappers

Relationship creation is an important and defining operation for the described application scenarios. For example, several text indexing tasks, such as topic modeling, derive relationships between the text and more general concepts. To support these types of task, an integrated framework must provide mechanisms to facilitate the creation of these relationships in the unified graph. The same type of mapping between source data and the unified graph is required for other types of data such as relational or semi-structured.

In our framework, the creation of relationships is encapsulated in *MAP* statements. Figure 7 shows an example (detailed in the experiments section) of such DML (Data Manipulation Language) query. The MAP statement (that triggers a *mapper*) is also encoded as an extension of a graph query language (SPARQL, in this case). The query selects all nodes of type ‘film’ and their respective labels. The selected elements are used to call the mapper *TokenMapper*.

Mappers are very similar to *stored procedures*. What sets them apart are (i) their integrated use in our ranking queries, and (ii) how they are hooked in the databases’s API so that any new data that matches the mapping criterion is passed through appropriate mappers. Point (ii) is not yet supported by our framework. (i) is discussed in the experiment section.

```

MAP DISTINCT ?movie, ?label
WITH TokenMapper
WHERE {
    ?movie a movie:film .
    ?movie rdfs:label ?label }

```

Fig. 7. MAP statement applying mapper *TokenMapper* to movies and their labels

Mappers are the mechanism that underlie the creation of the LLG. In a wider perspective, mappings are however not restricted to model transformations, but also allow transformations of data already in the unified graph, for example, to infer new relationships based on correlations between nodes. These ad hoc mappings are especially important for querying and analyzing data, enabling users to manipulate the underlying data at query time without the obligation to materialize the new relationships. For this reason, our approach addresses query and mapping as an interdependent and symbiotic process of data analysis and exploration.

A mapping process stores metadata related to the creation of the relationships, which can be explored at query time. Since relationships are associated with their mappers, multiple mappers can be used for the same type of relationship. For example, multiple text indexing strategies can be used simultaneously, then queries can specify the strategy that best fits the information need or simply take advantage of the multiple connections created by the diverse mappers.

Metadata about creation time and usage statistics of the relationships can also be used in a more expressive version of the extended query language presented here. Query modifiers could refer to this metadata to favor *novelty* or *popularity* (also important concepts in IR) of the relationships.

7 Experiments

We now show experiments that aim at demonstrating what we envision as a typical usage scenario for our framework. The database used in the experiments is the Linked Movie Data Base (LinkedMDB) [19], which we think is a good representative for the type of unified graph we aim at. The database integrates data from several sources (FreeBase, OMDb, DBpedia, Geonames, etc.). The process used to semantically integrate the distinct sources is similar to what is done in a typical Data Warehouse and precisely what we envision to be the workflow for the usage scenarios of our framework (i.e. integration of institutional data). The database contains 3,579,616 triples. The dataset has other important characteristics: (i) it encompasses the bulk of the production in an important area of human activity, (ii) data elements have clear semantics, (iii) data elements are organized based on several characteristics (type, genre, subject, etc.) and correlated in a complex graph topology. These characteristics support the applicability of our framework in real scenarios.

We implemented a basic mapper (*TokenMapper*) that maps input nodes into tokens present in their labels. The tokens are themselves stored as nodes in the

```

PREFIX movie: <http://data.linkedmdb.org/resource/movie/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT DISTINCT ?movie where{
  ?movie a movie:film .
  ?movie movie:initial_release_date ?date .
  FILTER ( fn:substring(?date, 1, 4) > "1990" ) .
  FILTER EXISTS { ?movie foaf:page ?page }
}
RANKED BY
2 RELEVANCE OF ?movie TO
  TokenMapper("My favorite films are Matrix and Avatar")
  DEPTH 2 FOLLOW TokenMapper:hasToken,
1 RELEVANCE OF ?movie TO movie:film_subject/461
  DEPTH 5 FOLLOW skos:subject

```

Fig. 8. Query that ranks movies according to relevance to a preferences text and to the subject “Virtual Reality” (film_subject/461)

graph database. The mapper receives as arguments the source node for the mappings and the text content to be mapped. This mapping uses a standard query analyzer (from Lucene’s library) that simply lowercases, removes stop words, and tokenizes the text. We are using this strategy for its simplicity and didacticism. In a real scenario, more modern techniques such as NER (Named Entity Recognition), LSA (Latent Semantic Analysis) or ESA (Explicit Semantics Analysis) would provide more efficient and meaningful mappings.

Table 2. Top-15 ranked results for the first query

top 15	name
0.67	Avatar
0.67	Avatar
0.55	The Matrix
0.53	The Matrix Revolutions
0.53	The Matrix Reloaded
0.33	The Thirteenth Floor
0.33	EXistenZ
0.33	Lawnmower Man 2: Beyond Cyberspace
0.33	Storm Watch (aka Code Hunter)
0.33	Strange Days
0.33	The Lawnmower Man
0.33	Welcome to Blood City
0.21	The Favorite
0.19	The Matrix Online
0.19	The Matrix Revisited

The DML query used for our mapping is shown in Fig. 7. We are mapping the label of movies to tokens. Executing the query triggers the mapping of each selected movie, generating the appropriate nodes for the tokens when needed. This same mapper can be used in the ranking clause, as will be shown below.

We now present analysis of queries to demonstrate the use of the metrics and mappers. At this point of development of our framework, we are focusing on the relevance and connectivity metrics, which we regard as having widespread applications and presenting the biggest challenges for query processing.

The first query (Fig. 8) retrieves and ranks movies relevant to a text stating movie preferences and that are also relevant to the subject ‘Virtual Reality’. This type of query can be derived from user’s profiles, for example. The query also selects elements based on structured predicates, specifying that returned movies should have a home page relationship and have been released after 1990.

The evaluation of this query can be divided in two phases: (i) graph matching and structured selection, and (ii) ranking based on topological properties. This separation in two distinct phases is only conceptual however. A query processor is free to combine the phases in any fashion to optimize the evaluation.

In the graph matching and selection phase, the triple pattern is matched against the data graph, the results are filtered according to the structured predicates. In the second phase, the multiple ranking criteria are evaluated.

The first ranking criterion is relevance to the text. To assess the scores for this metric, the query processor creates a temporary node and appropriate mappings are made using the specified mapper (TokenMapper). The spreading activation process is then executed to assess the correlation between each movie and the temporary node. The process is set to follow the relationships created by the mapper (hasToken). This is a typical mechanism for a keyword query type of interaction in our framework. A system-wide default keyword query mapper can be set so that queries can use the reserved KWQuery element, so that the parser automatically assigns the appropriate mapper and relationships to follow (as in Fig. 6a).

The second ranking criterion assesses correlation between movies and the subject “Virtual Reality”. The resulting scores from each criterion are normalized and aggregated, according to the specified weights, to generate the final score. The results (Table 2) show contributions from both ranking criteria. The movie Avatar is not directly related to the subject “Virtual Reality”, owing its high ranking to the high $tf*idf$ value of its name (note that $tf*idf$ s are not calculated, this is an emergent property of the relevance metric which is not restricted to text-based rankings)⁶. All movies from the Matrix franchise have scores combining both criteria. Lower ranking results also provide insight into the interplay between the rankings for this query (e.g. ‘The Favorite’ matches the keyword query with a lower $tf*idf$ -like score).

The second query (Fig. 9) uses the connectivity metric to discover films correlated to ‘The Silence of the Lambs’. The query specifies that the analysis

⁶ The second Avatar record refers to a lesser known Singaporean film (introducing a *reputation* metric in the query would certainly lower its score).

```
PREFIX film: <http://data.linkedmdb.org/resource/film/>
PREFIX resource: <http://data.linkedmdb.org/resource/movie/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
SELECT DISTINCT ?movie WHERE{
  ?movie a resource:film .
  ?movie resource:initial_release_date ?date .
  FILTER ( fn:starts-with(?date, "199")
}
RANKED BY
CONNECTIVITY OF ?movie TO film:38145
DEPTH 4 FOLLOW (skos:subject, resource:director)
```

Fig. 9. Query that retrieves films correlated to ‘The Silence of the Lambs’ (film:38145)

Table 3. Top-10 ranked results for the second query

top 10	name
1.0	The Silence of the Lambs
0.81	Man Bites Dog
0.80	Natural Born Killers
0.80	Butterfly Kiss
0.80	Freeway
0.79	Seven
0.79	Aileen Wuornos: The Selling of a Serial Killer
0.79	Serial Mom
0.79	Copycat
0.79	The Young Poisoner’s Handbook

Table 4. Top-10 ranked results for the first query

top 10	name
1.0	The Silence of the Lambs
0.34	Philadelphia
0.34	Cousin Bobby
0.25	Man Bites Dog
0.25	Natural Born Killers
0.25	Butterfly Kiss
0.24	Freeway
0.23	Seven
0.23	Aileen Wuornos: The Selling of a Serial Killer
0.23	Serial Mom

should consider only the relationships ‘movie:director’ and ‘skos:subject’. It is interesting to note that setting the modifier *depth* to 4 means that indirect correlations are also considered. For example, a film could receive a positive score even though it does not share a subject with ‘The Silence of the Lambs’, as long as it is correlated with a film that does share a subject.

The results for the query are shown in Table 3. The output is strongly biased towards scores generated by correlations through common subjects (which tend to form tighter clusters). If the user wants to increase the importance of the ‘director’ relationship to retrieve more movies correlated to the director of ‘The Silence of the Lambs’, the user can separate the relationships into two ranking criteria. This type of user interactivity is another important advantage our declarative querying scheme. The results of splitting the rankings in such a way are presented in Table 4. The results are for a version of the query that used a 3:1 weight division favoring the *director* relationship.

8 Related Work

We now discuss related work on data integration in various levels: from data access integration, through syntactic/semantic integration, and up to application or query model integration. Integration at any level is highly dependent on the lower levels.

8.1 Data Access Integration

The first level of integration must provide a unique access point for the data. This can be accomplished by basically two approaches: centralizing the data or connecting the data sources in an infrastructure that simulates a centralized repository. Centralized integration of institutional data is typically related to the deployment of data warehouses or data marts [21]. Data centralization approaches have also been proposed in the context of the Semantic Web [19], and the DBpedia project⁷ is a notable example of this type of approach.

The research on Federated Databases aims at providing a unified view of the data while maintaining the autonomy of the data sources [32]. In the context of the Semantic Web, Schwarte et al. [31] have proposed a federation layer for Linked Open Data. Schenk and Staab [30] have proposed a mechanism for the specification of views over Linked Data, enabling declarative federation of data sources.

Our framework is independent of the specific strategy chosen for data access integration. The requirement is that all interaction is done as if the data was integrated in a unified graph. Whether this integration is done through federation or physically integrating the data is an architectural decision based on expectations of performance and requirements for preserving the autonomy of data sources.

⁷ <http://dbpedia.org/>.

8.2 Data Model Integration

Data integration requires enabling data manipulation under a unified model. Federated databases frequently employ the relational model (common among data sources) for the integration. Data mining, which has application-specific requirements, favors the multidimensional model [22].

In the Semantic Web, the adopted unifying model is the RDF graph. The Resource Description Framework (RDF)⁸ is a general-purpose language created for representing information about resources in the Web. The basic unit of information is a statement triple, which contains a subject, a predicate, and an object. All elements in a triple are identified by URIs (except for objects that can also be literal values). Triples can refer to each other, forming a graph. The advantage of the RDF model comes from its simplicity, enabling the representation of data from a wide range of domains.

There has been a substantial amount of research in mapping other data models into RDF [3, 6]. The W3C RDB2RDF Working Group⁹ is defining languages and standards for mapping relational databases into RDF.

Besides having the data in a unified representation model, it is important to correlate data from the diverse sources into unified concepts. In the relational world, this process is known as record deduplication or linkage and is part of the ETL (Extraction Transformation Loading) workflow [18]. In the Semantic Web, the usual way to represent these correlations is the creation of *sameAs* relationships between entities. These relationships can be created manually or by automated processes. Hassanzadeh and Consens [19] employ several string matching techniques to correlate Linked Open Data from diverse sources to create an interlinked version of a movies database.

In this proposal, we assume that the institutional data is integrated in an RDF graph. This allows us to take advantage of other standardized technologies developed in the context of the WWW and the Semantic Web, such as universal identification through URIs, semantic integration through *sameAs* relationships, and the SPARQL query language.

8.3 Query Model Integration

Once data is integrated, it becomes possible to pose queries that could not be answered before, producing more valuable information for institutions and the public. The integration approaches, however, typically focus on integrating data under a specific query model, such as the relational or OLAP. This usually constrains the range of data models that can be integrated and, foremost, restricts direct querying of the integrated data from applications that use other query models.

Recently, there have been initiatives aimed at tackling integration at the application/query level. The research community has identified the interplay between

⁸ <http://www.w3.org/RDF/>.

⁹ <http://www.w3.org/2001/sw/rdb2rdf/>.

the fields of Databases (DB) and Information Retrieval (IR) as a means to improve data integration and query expressiveness across applications [2, 9]. The drive to integrate the areas stems from the fact that they represent the bulk of data stored and processed across institutions. Furthermore, either field has been very successful by their own but still faces challenges when dealing with interactions typical to the other field.

The integration of the IR and DB areas has been an important topic in the agenda of the research community for many years. Following the initial identification of challenges and applications, several successful approaches were proposed and implemented [33]. Most prominent research focuses on keyword queries over structured data and documents, top-k ranking strategies and extraction of structured information from documents.

Keyword query research draws from the simple yet effective keyword query model to allow integrated querying over documents and structured data. Most of the frameworks match keywords to documents, schema and data integrated in a graph structure. The connected matches form trees that are ranked based on variations of IR metrics such as $tf*idf$ and PageRank. Some of the research focus on optimizing the top-k query processing [23] while others implement more effective variations of the ranking metrics [24].

Keyword queries over structured data are intended for tasks where the schema is unknown to the user. The techniques are effective for data exploration, but there is no support for more principled interactions. There are conceptual and structural mismatches among queries, data and results that make returned matches hard to predict and interpret. Furthermore, the queries can only express relevance between the provided keywords and database elements. Our query model can represent many more correlation criteria that can be combined arbitrarily in user-defined expressions. More importantly, the queries can correlated any type of data in the graph database.

The research on Top-k queries focus on enabling efficient processing of ranked queries on structured and semi-structured data. Ranking is based on scores derived from multiple predicates specified in the query. The main challenge is to compute results avoiding full computation of the expensive joins. The proposals vary on adopted query model, data access methods, implementation strategy, and assumptions on data and scoring functions (see [20] for a contextualized survey).

Scoring functions enable ranking based on properties of data elements. There is, however, no simple means to rank results based on the context of elements or how they are correlated, typical requirements for IR-like applications and a defining characteristic of our SA-based ranking scheme.

This type of contextual ranking could only be implemented in an ad-hoc fashion through complex scoring functions. Since the query processor would be unaware of the semantics of the queries and the topology of the relationships, there would be no opportunity for the optimizer to make sensible execution plans. Furthermore, the relational model assumed in most research has no means to reference individual data elements, an important requirement for effective data correlation. Our focus is on offering predicates that allows ranking based on contextual metrics not readily available as attributes. The proposal described

here is complementary to regular top-k querying. It is important to support both types of ranking, since they are recurrent to many applications.

Information Extraction refers to the automatic extraction from unstructured sources of structured information such as entities, relationships between entities, and attributes describing entities [29]. Information Extraction systems employ two main techniques to harvest relationships (or facts) from text: extrapolating extraction patterns based on example seeds [13] and employing linguistic patterns to discover as many types of relationships as possible, task known as Open Information Extraction [4]. Loading the extracted facts on a DBMS allows declarative querying over the data. This is a one-way, data-centric type of integration of DB and IR. The integration proposed here focuses on unified querying and data models. The framework proposed allows easy integration of Information Extraction system's output, maximizing the benefits of both approaches.

We argue that the mentioned approaches tend to focus on infrastructure issues related to extremes of enabling the type interaction present in one area over the data model of the other. In this paper we take a top-down approach to modeling the integration, questioning what are the main and defining properties of each area, and how to offer a unified, non-modal interaction over data and query models.

9 Conclusion

We showed how modern standards and technologies developed to solve integration issues on the Web can be applied in a unifying framework for institutional data. Representing the integrated data as a graph is a good strategy for data model integration. Our main contribution is on extending this type of integration to a higher level of abstraction, tackling integration of query models.

In our approach, the key to achieve more expressiveness at the query level is the combination of flexible metrics in a declarative model. Our query model redefines several metrics that rank entities based on the topology of their correlations. To the best of our knowledge, this is the first time the metrics presented are considered and formalized under the same model. Similarly, we are not aware of other ranking strategies that enable the level of expressiveness offered by the combination of our metrics and a declarative language. This combination allows data correlation queries that cover a wide range of applications. The introduced mappers play an important role as a data management mechanism to support this high level of integration.

As suggested by the query examples presented (Fig. 6), it is possible to represent information needs that would require a level of data analysis that is beyond current implementations of typical DB or IR systems. In fact, answering the type of queries introduced here in a typical technological environment nowadays would require substantial engineering for the implementation of *ad-hoc* solutions. The expressiveness of the queries allowed by the extended languages sometimes blurs the line between declarative queries and data analysis. Given the computational requirements of such settings, it is important to introduce optimization mechanisms and heuristics to compute approximate answers. Our declarative querying

scenario opens many opportunities for query optimization. Details about the mechanisms we are currently adopting are described in [15].

We expect query-level integration to become increasingly important as our technological landscape continues to diversify. We showed how our model can cover a broad range of models and applications. Our experiments indicate the practicability of our approach, especially regarding the use of mappers to simplify data integration and enable more expressive querying.

Acknowledgments. The authors would like to thank Prof. Frank Wm. Tompa for feedback and encouragement in earlier stages of this work. This work was partially financed by the Microsoft Research FAPESP Virtual Institute (NavScales project), CNPq (MuZOO Project and PRONEX-FAPESP), INCT in Web Science (CNPq 557.128/2009-9) and CAPES, with individual grants from CAPES and FAPESP (process 2012/15988-9).

References

1. Alves, H., Santanchè, A.: Abstract framework for social ontologies and folksonomized ontologies. In: SWIM. ACM (2012)
2. Amer-Yahia, S., Case, P., Rölleke, T., Shanmugasundaram, J., Weikum, G.: Report on the DB/IR panel. SIGMOD Record **34**(4), 71–74 (2005)
3. Auer, S., Dietzold, S., Lehmann, J., Hellmann, S., Aumüller, D.: Triplify: lightweight linked data publication from relational databases. In: Proceedings of the 18th International Conference on World Wide Web, WWW 2009 (2009)
4. Banko, M., Cafarella, M.J., Soderland, S., Broadhead, M., Etzioni, O.: Open information extraction from the web. In: IJCAI, pp. 2670–2676 (2007)
5. Berners-Lee, T.: Giant global graph. Online posting, 2007. <http://dig.csail.mit.edu/breadcrumbs/node/215>
6. Bizer, C.: D2rq - treating non-rdf databases as virtual rdf graphs. In: Proceedings of the 3rd International Semantic Web Conference (ISWC2004) (2004)
7. Blanco, R., Lioma, C.: Graph-based term weighting for information retrieval. Inf. Retr. **15**(1), 54–92 (2012)
8. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. J. Mach. Learn. Res. **3**(4–5), 993–1022 (2003)
9. Chaudhuri, S., Ramakrishnan, R., Weikum, G.: Integrating DB and IR technologies: what is the sound of one hand clapping? In: CIDR, pp. 1–12 (2005)
10. Costa, L., Oliveira Jr., O., Travieso, G., Rodrigues, F., Boas, P., Antiqueira, L., Viana, M., Rocha, L.: Analyzing and modeling real-world phenomena with complex networks: a survey of applications. Adv. Phys. **60**, 329–412 (2011)
11. Costa, L.D.F., Rodrigues, F.A., Travieso, G., Boas, P.R.V.: Characterization of complex networks: a survey of measurements. Adv. Phys. **56**(1), 167–242 (2007)
12. Crestani, F.: Application of spreading activation techniques in information retrieval. Artif. Intell. Rev. **11**(6), 453–482 (1997)
13. Etzioni, O., Cafarella, M., Downey, D., Kok, S., Popescu, A.-M., Shaked, T., Soderland, S., Weld, D.S., Yates, A.: Web-scale information extraction in Know-ItAll. In: WWW, pp. 100, 26 March 2004
14. Getoor, L., Diehl, C.P.: Link mining: a survey. SIGKDD Explor. Newsl. **7**(2), 3–12 (2005)

15. Gomes Jr., L., Costa, L., Santanchè, A.: Querying complex data. Technical Report IC-13-27, Institute of Computing, University of Campinas, October 2013
16. Gomes Jr., L., Jensen, R., Santanchè, A.: Query-based inferences in the Complex Data Management System. In: *Structured Learning: Inferring Graphs from Structured and Unstructured Inputs (SLG-ICML)* (2013)
17. Gomes Jr., L., Jensen, R., Santanchè, A.: Towards query model integration: topology-aware, ir-inspired metrics for declarative graph querying. In: *Graph Q-EDBT* (2013)
18. Han, J., Kamber, M.: *Data Mining: Concepts and Techniques*. Morgan Kaufmann, San Francisco (2006)
19. Hassanzadeh, O., Consens, M.: Linked movie data base. In: *Proceedings of the 2nd Workshop on Linked Data on the Web (LDOW2009)* (2009)
20. Ilyas, I.F., Beskales, G., Soliman, M.A.: A survey of top- k query processing techniques in relational database systems. *ACM Comput. Surveys* **40**(4), 11:1–11:58 (2008)
21. Imhoff, C., Galembo, N., Geiger, J.G.: *Mastering Data Warehouse Design: Relational and Dimensional Techniques*. Wiley, Chichester (2003)
22. Jarke, M., Lenzerini, M., Vassiliou, Y., Vassiliadis, P.: *Fundamentals of Data Warehouses*. Springer, Heidelberg (2003)
23. Kimelfeld, B., Sagiv, Y.: Finding and approximating top- k answers in keyword proximity search. In: *PODS* (2006)
24. Luo, Y., Wang, W., Lin, X., Zhou, X., Wang, J., Li, K.: SPARK2: Top- k keyword query in relational databases. *TKDE* **23**(12), 1763–1780 (2011)
25. Markovitch, S., Gabrilovich, E.: Computing semantic relatedness using wikipedia-based explicit semantic analysis. In: *IJCAI* (2007)
26. Ngonga Ngomo, A.-C., Heino, N., Lyko, K., Speck, R., Kaltenböck, M.: SCMS – Semantifying content management systems. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) *ISWC 2011, Part II. LNCS*, vol. 7032, pp. 189–204. Springer, Heidelberg (2011)
27. Rodriguez, M.A., Neubauer, P.: The graph traversal pattern. *CoRR*, abs/1004.1001 (2010)
28. Rodriguez, M.A., Pepe, A., Shinavier, J.: The dilated triple. In: Badr, Y., Chbeir, R., Abraham, A., Hassanien, A.-E. (eds.) *Emergent Web Intelligence: Advanced Semantic Technologies*, pp. 3–16. Springer, London (2010)
29. Sarawagi, S.: Information extraction. *Found. Trends Databases* **1**(3), 261–377 (2008)
30. Schenk, S., Staab, S.: newblock Networked graphs: a declarative mechanism for SPARQL rules, SPARQL views and RDF data integration on the web. In: *WWW* (2008)
31. Schwarte, A., Haase, P., Hose, K., Schenkel, R., Schmidt, M.: FedX: A federation layer for distributed query processing on linked open data. In: Antoniou, G., Grobelnik, M., Simperl, E., Parsia, B., Plexousakis, D., De Leenheer, P., Pan, J. (eds.) *ESWC 2011, Part II. LNCS*, vol. 6644, pp. 481–486. Springer, Heidelberg (2011)
32. Sheth, A., Larson, J.: Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surveys* **22**(3), 183–236 (1990)
33. Weikum, G., Kasneci, G., Ramanath, M., Suchanek, F.: Database and information-retrieval methods for knowledge discovery. *Commun. ACM* **52**(4), 56–64 (2009)
34. White, S., Smyth, P.: Algorithms for estimating relative importance in networks. In: *SIGKDD* (2003)

Transactions on Large-Scale Data- and
Knowledge-Centered Systems XIX

Special Issue on Big Data and Open Data

Hameurlain, A.; Küng, J.; Wagner, R.; Bianchini, D.; De
Antonellis, V.; De Virgilio, R. (Eds.)

2015, IX, 129 p. 40 illus., Softcover

ISBN: 978-3-662-46561-5