Almost Linear Semantic XML Keyword Search

Joe Tekli[†]
School of Engineering,
Dept. of Elec. and Compt. Eng.
Lebanese American University
36 Byblos, Lebanon
joe.tekli@lau.edu.lb

Gilbert Tekli
Fac. of Technology,
Mechatronics Dept.
University of Balamand
100 Tripoli, Lebanon
gtekli@gmail.com

Richard Chbeir
LIUPPA Lab., Univ. of Pau
and Adour Countries
IUT de Bayonne, CS Dept.
64000 Anglet, France
richard.chbeir@univ-pau.fr

ABSTRACT

Many efforts have been deployed by the IR community to extend free-text query processing toward semi-structured XML search. Most methods rely on the concept of Lowest Comment Ancestor (LCA) between two or multiple structural nodes to identify the most specific XML elements containing query keywords posted by the user. Yet, few of the existing approaches consider XML semantics, and the methods that process semantics generally rely on computationally expensive word sense disambiguation (WSD) techniques, or apply semantic analysis in one stage only: performing *query relaxation/refinement* over the *bag of words* retrieval model, to reduce processing time. In this paper, we describe the building blocks of a new approach for XML keyword search aiming to solve the limitations mentioned above. Our solution first transforms the XML document collection (offline) and the keyword query (on-the-fly) into meaningful semantic representations using context-based and global disambiguation methods, specially designed to allow almost linear computation efficiency. Consequently, the semantically augmented XML data tree is processed for structural node clustering, based on semantic query concepts (i.e., key-concepts), in order to identify and rank candidate answer sub-trees containing related occurrences of query key-concepts. Preliminary experiments highlight the quality and potential of our approach.

CCS CONCEPTS

Information systems - Data management systems - Database design and models - Data model extensions - Semi-structured data • Information systems - Information retrieval - Document representation • Information systems - Information retrieval - Information retrieval query processing • Information systems - World Wide Web - Web searching and information discovery.

KEYWORDS

XML, Semi-structured Data, Semantic Analysis, Semantic Disambiguation, Keyword Search, Query Processing.

ACM Reference format:

Joe Tekli, Gilbert Tekli, and Richard Chbeir. 2021. Almost Linear Semantic XML Keyword Search. In *Proceedings of the 13th International Conference on Management of Digital EcoSystems (MEDES'21)*. ACM, Hammamet, Tunisia.

1 Introduction

Various methods have been proposed for XML ranked retrieval. While most approaches consider content-and-structure features in specifying XML query constraints, few approaches have targeted semantic XML search based on simple keyword queries. Most approaches in this category exploit the concept of LCA (Lowest Common Ancestor) between two or multiple structural nodes to identify the most specific XML elements containing query keywords posted by the user. Yet LCA-based methods underline various limitations: i) each result candidate must contain all query keywords, which is not always intuitive since a candidate result (element or sub-tree) containing most (and not necessarily all) keywords might be deemed relevant by the user; ii) some meaningful results might be missed: as XML trees underline different nesting hierarchies, restricting results to the LCA encompassing all keywords might miss some more general, and yet relevant results; iii) few of the proposed approaches consider semantics: for instance, when submitting sample keyword query “Universities in Sao Paulo”, the user is probably interested in information concerning universities, academies and colleges in Sao Paulo, and cities in its vicinity such as Campinas, Sao Carlos, etc. Hence, semantic analysis becomes essential in such a context in order to improve search results; iv) the few existing methods that do target XML semantics generally rely on word sense disambiguation (WSD) and are computationally expensive, or v) apply semantic analysis in one stage only, performing *query relaxation/refinement* over the *bag of words* retrieval model, to reduce processing time.

In this paper, we describe the building blocks of a new approach for XML keyword search aiming to solve the limitations mentioned above. We propose to integrate *semantic analysis* and *structural clustering* in formulating an efficient solution to the problem. Our solution first transforms the XML document collection (offline) and the keyword query (on-the-fly) into meaningful semantic representations using context-based and global disambiguation methods, specially designed to allow almost linear computation efficiency. The semantically augmented XML data tree is processed for structural node clustering, based on semantic query concepts (i.e., key-concepts), in order to identify and rank candidate answer sub-trees containing related occurrences of query key-concepts. The overall architecture of our approach is depicted in Figure 1.

Section 2 reviews the background in XML and query semantic analysis. Section 3 provides an overview of our framework. Sections 4 and 5 respectively describe the XML semantic analysis and keyword query semantic analysis components. Section 6 describes the query processing component. Section 7 provides preliminary experimental results, before concluding in Section 8.

[†] Corresponding author.

MEDES '21, Nov. 1–3, 2021, Virtual Event, Tunisia
© 2021 Association for Computing Machinery. ACM ISBN 978-1-4503-8314-1/21/11…$15.00. https://doi.org/10.1145/3444757.3485079

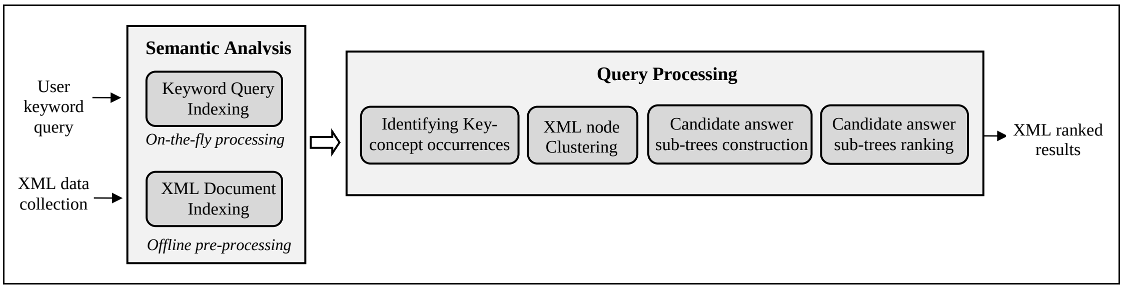


Figure 1: Overall architecture of our XML keyword search approach

2 Background

In this section, we briefly review the background in semantic information retrieval, while focusing on XML and keyword query semantic analysis and disambiguation.

2.1 Semantic Information Retrieval

The retrieval model for an information retrieval system specifies how documents and queries are represented, and how these representations are compared to produce relevant result estimates [7]. A core problem in this context is lexical ambiguity: a word may have multiple meanings (homonymy), a word maybe implied by other related words (metonymy), and/or several words can have the same meaning (synonymy) [34].

The lexical ambiguity problem becomes even more acute on the Web, with the latter's heterogeneous and unstructured nature which makes it even more difficult to query and retrieve meaningful information. Semantic IR is part of the Semantic Web vision [60] that promises to solve the retrieval ambiguity problem, by i) associating terms in Web pages and queries with explicit semantics (i.e., word senses or concepts), and then ii) performing search functions based on document/query concepts rather than plain terms [45]. A core challenge in this context is word sense disambiguation (WSD): how to resolve the semantic ambiguities and identify the intended meanings of document terms and query keywords [10]. Various methods have been proposed for WSD in the literature [34, 43, 59]. They fall in two main categories: *corpus-based* WSD and *knowledge-based* WSD. The corpus-based approach is data-driven, as it involves information about words previously disambiguated and requires supervised learning from sense-tagged corpora to enable predictions for new words. Knowledge-based methods are knowledge-driven, as they handle a sense inventory and/or a repository of information about words that can be exploited to distinguish their meanings in the text. Machine-readable knowledge bases (e.g., dictionaries or semantic networks: thesauri, taxonomies, or ontologies) provide ready-made sources of information about word senses to be exploited in knowledge-based WSD. While corpus-based methods have been popular in recent years [6, 30], they are generally data hungry and require extensive training, huge textual corpora, and/or a considerable amount of manual effort to produce a relevant sense-annotated corpus, which are not always available or feasible in practice. Therefore, knowledge-based methods have been receiving more attention [14, 34]. In the remainder of our study, we focus on knowledge-based WSD and semantic analysis.

2.2 XML Semantic Analysis and Disambiguation

While a considerable amount of research has been undertaken around (knowledge-based) WSD in flat textual data [43], yet few approaches have been developed in the context of XML and semi-structured information [59]. The main difference resides in the notion of XML (structural) contextualization. The context of a keyword, in traditional textual data, consists of the set of terms in the keyword's vicinity (i.e., terms occurring to the left and right of the considered keyword, within a certain predefined distance from the keyword [10]). However, there is no clear definition regarding the context of a node in an XML tree. The authors in [56, 57] consider the context of an XML data element to be efficiently determined by its parent element, and thus process a parent node and its children data elements as one unified (canonical) entity, using context-driven search techniques for determining the relationships between the different unified entities, so as to identify related semantic labels. In [54, 55], the authors extend the notion of XML node context to include the whole XML root path, i.e., path consisting of the sequence of nodes connecting a given node with the root of the XML document (or document collection). They consequently perform per-path sense disambiguation, comparing every node label in each path with all possible senses of node labels occurring in the same path (using a gloss-based WordNet similarity measure [8]) in order to select the most appropriate sense for the label at hand. Different from the notions of parent context and path context, the authors in [68] consider the set of XML tag names contained in the sub-tree rooted at a given element node, i.e., the set of labels corresponding to the node at hand and all its subordinates, to describe the node's XML context. The authors apply a similar paradigm to identify to contexts of all possible node label senses in WordNet. Consequently, they perform label sense disambiguation by comparing the XML label context to all candidate sense contexts in WordNet, identifying the sense (semantic concept) with the highest similarity. In [40], the authors combine the notions of parent context and descendent (sub-tree) context in disambiguating generic structured data (e.g., XML, web directories, and ontologies). The authors consider that a node's context definition depends on the nature of the data and the application domain at hand. They propose various edge-weighting heuristics (namely a Gaussian decay function) to identify *crossable* edges, i.e., nodes reachable from a given node through any *crossable* edge belong to the target node's context. Consequently, structure disambiguation is undertaken by comparing the target node label with each candidate sense (semantic concept) corresponding to the labels in the target node's context (using an edge-based semantic similarity measure [35], following the hypernymy/hyponymy relations in WordNet) in order to identify the highest matching semantic concept.

Another concern in XML-based WSD is how to effectively process the context of an XML node taking into account the structural dispositions of XML data. In fact, most existing WSD methods developed for flat textual data [34, 43], and those developed for XML-based data [54-57], follow the bag-of-words paradigm where the context is processed as a plain set of words surrounding the term/label (node) to disambiguate. In other words, all context nodes are treated the same, despite their structural positions in the XML tree. We encountered an approach in [40] which extends the traditional bag-or-words paradigm with additional information considering distance weights separating the context and target nodes (identified as *relational information model* [40]). The authors employ a heuristic Gaussian distance decay function estimating edge weights such that the closer a node (following a user-specified direction, e.g., ancestor, descendent, or both), the more it influences the target node's disambiguation [40]. The semantic contribution of each context node is weighted by its position in the context graph of the target node.

2.3 Query Semantic Analysis and Disambiguation

Semantic query analysis in information retrieval usually involves two steps: i) WSD to identify the user's intended meaning of query terms, and ii) semantic query representation/expansion in order to alter the query so that it achieves better (precision and recall) results [51]. As described in the previous section, traditional semantic analysis and disambiguation techniques usually rely on the notion of context such that terms (e.g., node labels in the context of XML) that appear together in the same context have related meanings [10]. While context-based solutions are applicable with classic IR queries which are rather lengthy (e.g., 15 terms on average for short queries [72], reaching up to 50-85 terms for long queries [11]), nonetheless, keyword queries on the Web are usually 2-3 words long [13] which is generally insufficient in identifying a meaningful context [34, 40]. In fact, lexical ambiguity with Web search is often the consequence of the low number of query words entered on average by Web users [32]. Therefore, some sort of user interaction is usually required to counter the lack of contextualization, and more accurately identify the intended senses of Web query terms [19, 71].

Various methods for interactive keyword querying have been proposed in the literature, e.g., [33, 50] [25, 58]. Most existing approaches are *corpus-based* in that they expand user queries by adding words that co-occur with the query terms in a given corpora, i.e. words that, on a probabilistic ground, are believed to describe the same *semantic concept* (e.g. *car* and *driver*). Here, expansion terms are usually identified from i) user feedback: extracting frequent terms occurring in previous results deemed relevant by the user [33, 50], and/or ii) query logs: identifying frequent terms in the document collection based on the associations between past queries and the documents downloaded by the user [25, 58]. Yet, the extensive training and huge corpora requirements of *corpus-based* methods makes them less practical in the context of Web search applications, which has led to a growing interest in *knowledge-based* solutions [29, 49]. The latter family of methods investigates the use of ontological information to assist the user in formulating and/or expanding keyword queries by: i) allowing user interaction to identify the intended senses of query-terms, and then ii) expanding/modifying query keywords via their most related semantic concepts in the reference semantic source (e.g., WordNet) [51].

Following [12], a keyword query is first processed for lexical normalization, and then presented to the user as a set of lexical tokens, where each token is associated with a set of possible semantic meanings (identified using WordNet and/or domain specific ontologies). Consequently, the user is asked to select the most relevant sense for each lexical token. The system then exploits the selected user senses to reformulate the query using dedicated heuristics (e.g., replacing actual keywords via their synonyms with highest frequency of usage in WordNet, identifying negative keywords, i.e., the terms corresponding to the highest frequency synset remaining beside the one selected by the user, etc.), thus obtaining a semantically augmented keyword query. A similar approach is adopted in [37] with a special emphasis on failed-query reformulation. The authors in [37] assume that the reformulation of a failed query without help from the system can be frustrating to the user, and thus suggest to assist the later by proposing semantically meaningful keywords selected from WordNet (using heuristics similar to those adopted in [12]). The method in [37] is developed in the context of the NALIX project for building an interactive natural language interface for querying XML [36].

A fully automated approach to *knowledge-based* query disambiguation is introduced in [44], where the authors exploit structural pattern recognition [20] in mapping query keyword senses. The proposed method creates a local semantic network for each keyword-sense in the query, including most semantic relations utilized in WordNet [23] (hypernymy, hyponymy, meronymy, etc.). Then, for each possible configuration of senses, the system identifies the intersections between corresponding pair-wise local semantic networks using an adapted structure pattern recognition algorithm. Common nodes are those that can be reached through both semantic networks being compared. The configuration with the highest intersection score (i.e., highest number of intersecting nodes) is selected as the one encompassing the most relevant keyword senses. In a subsequent step, the authors propose various heuristics to expand the query using synset, hyponymy and/or gloss information. Experimental results in [44] show a 26.85% improvement in retrieval precision over the plain query words.

Note that most existing studies targeting *knowledge-based* query semantic analysis, e.g., [29, 49] [12, 37, 46], do not evaluate the complexity (or execution time) levels of their proposed methods. Nonetheless, time complexity is critical for on-the-fly execution on the Web (in comparison with document semantic analysis which could be performed offline). The time complexity of query semantic analysis might even prove to be problematic in the case of the pattern recognition-based methods [16, 44], since traditional structure pattern recognition problems are usually of exponential complexity [20, 47].

3 Proposal Overview

Semantic similarity evaluation between two terms usually consists in looking up each term's lexical concept in a reference knowledge base (e.g., a semantic network such as WordNet), and consequently comparing the underlying concepts. Nonetheless, semantic similarity evaluation has been proven to be an expensive task: comparing two semantic concepts following one of the most prominent semantic similarity measures in the literature, i.e., [38], requires $O(|SN| \times Depth(SN))$ time where $|SN|$ is the size (i.e., cardinality in number of concepts) of the reference semantic network SN, and $Depth(SN)$ its maximum depth. Evaluating the semantic similarity between query keywords and each label/term in the XML document collection becomes extremely complex, and practically unfeasible.

A way of getting round the complexity problem would be to perform semantic analysis of the XML document collection, offline, and prior to the retrieval phase. This consists in transforming the XML documents into weighted semantic trees (graphs), and transforming and expanding the keyword query into a set of weighed semantic concepts. Consequently, an adapted XML IR engine (cf. Section 6) processes the semantically indexed documents and queries, so as produce more meaningful results. Our semantic analysis processes are depicted in Figure 2.

Note that while semantic query indexing is performed online, XML document indexing is performed offline, and does not affect the complexity of the approach. As shown in Figure 2, semantic indexing consists of three main phases: i) Linguistic Normalization, ii) Sense Disambiguation, and iii) Semantic Representation. While the first phase (Linguistic Normalization, including *tokenization*, *expansion*, *stop word removal*, and *stemming*) is similar for both document labels and query keywords, yet, we design the latter two (sense disambiguation, and semantic representation) differently following the data models and requirements at hand. Sense disambiguation usually relies on the notion of context, where terms that appear together in the same context have related meanings [10]. While context information is available for XML document nodes (e.g., the context of a node could be its parent node, its root path, the whole document tree containing the node, etc.), yet, keyword queries on the Web are usually two-to-three words long [13] which is generally insufficient in identifying a meaningful context [34, 40]. Hence, we introduce two different methods for document and query sense disambiguation: i) Context-based Sense Disambiguation (CSD) for XML documents, ii) Global Sense Disambiguation (GSD) for keyword queries.

In the following, Sections 3 and 4 present the XML document semantic analysis and the keyword query semantic analysis processes respectively.

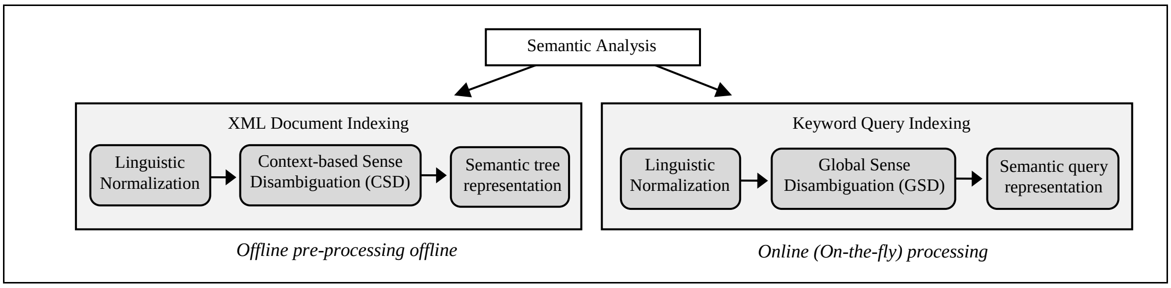

Semantic Analysis
XML Document Indexing
Linguistic Normalization
Context-based Sense Disambiguation (CSD)
Semantic tree representation
Keyword Query Indexing
Linguistic Normalization
Global Sense Disambiguation (GSD)
Semantic query representation
Offline pre-processing offline
Online (On-the-fly) processing

Figure 2: Semantic analysis of XML document and keyword query

4 Semantic XML Document Analysis

Our XML document semantic analysis process consists in: i) disambiguating each label following its context, to associate each label with the proper semantic concept in the reference knowledge base (e.g., WordNet), and ii) producing a semantically indexed XML tree, with the corresponding index structures and pair-wise concept semantic weights, to be consequently utilized in the query processing task. We describe the latter in the following sub-sections.

4.1 XML Context-based Sense Disambiguation

Our XML sense disambiguation approach was introduced in [15, 62]. Here, we only provide an overview of the approach describing the constructs and methods required in our current study.

Different from previous approaches which limit XML context to the parent node [56, 57], to the root node path [54, 55], to the node sub-tree [68], or to nodes reachable through heuristically identified crossable edges [40], we introduce the notion of XML *Sphere-Ring* context, inspired from the sphere-search paradigm in XML IR [26][1], to consider the whole structural surrounding of an XML node, including its ancestors, descendants, and siblings, tuned to better describe the node's context. An XML *ring* w.r.t. to a given node consists of the set of nodes situated at a specific distance from the center node. An XML sphere encompasses all rings contained at distances lesser or equal to the size (diameter) of the sphere. The size of the XML sphere is tuned following the nature of the XML data at hand (e.g., certain XML trees might underline specialized and domain-specific data, and thus would only require small contexts so as to achieve relevant WSD results, whereas more heterogeneous and generic XML data might require larger contexts to better describe the intended meaning of each node label).

In addition, we extend the traditional bag-of-words WSD paradigm, adopting a relational information approach, i.e., considering the interconnections among XML nodes in computing disambiguation scores (in contrast with the classic bag-of-words approach [54-57], where all context nodes are treated as a homogeneous set of words regardless of their proximity/relations with the target node). We consider the structural distance separating the center node and each of its context nodes, following the intuition that the farther the context node from the sphere center, the lesser should be its impact in determining the semantic meaning of the center node label. Formally, consider $R_d(n)$ to be the ring corresponding to the center node n at distance d, i.e. the set of all nodes whose distance from n is d. Hence, the context sphere $S_D(n)$ of node n, with size D, consists of all the rings contained in $S_D(n)$, such that $S_D(n)$ = {all $R_d(n)$ / $d \leq D$}. Following our *Sphere-Ring* context model, node scores can be weighted following the sizes of the sphere rings to which they correspond, such that the larger the sphere ring radius, the lesser the node weight. Hence, we can represent the context of a node n as a weighed vector, whose dimensions correspond to the all distinct nodes in its sphere context, weighted following their distances from the center node. In short, our approach:

- Integrates all notions of XML context, including ancestor, decedent, and sibling structural relations, which were considered separately in existing studies [54-57, 68],
- Allows the user/system administrator to manually and/or automatically tune the size of the XML context window following the nature and properties of the XML data at hand, in comparison with most existing static methods [54-57, 68],
- Extends the traditional bag-of-words WSD paradigm, adopting a relational information approach so as to consider the interconnections among XML nodes in computing disambiguation scores, in contrast with most existing methods using the traditional bag-of-words approach [54-57].

[1] While comparable to the concept of XML sphere exploited in [33], the latter consists of an XML retrieval paradigm for computing TF-IDF scores, selecting and ranking XML query answers, which is different in its use and objectives from our current XML semantic disambiguation proposal.

Once the contexts of all XML nodes have been determined, we process each target node label and its context node labels for WSD. Here, we evaluate the semantic similarity/relatedness between the target node label and each of its context node labels, by comparing the node's context with the context of each of its potential senses, extracted from the reference semantic source (a similar paradigm is utilized in [68] for XML node annotation). The idea is to first identify all possible senses of the target word node label in the reference semantic network. Consequently, we exploit the same notion of *Sphere-Ring*, which we adopted for XML trees (graphs), to identify the context of each potential sense in the reference semantic network (e.g., WordNet). Having computed the weighted context for the XML target node in the XML document tree (graph), and each of its possible senses in the semantic network, we compute the similarity between the node vector and each of its sense vectors. The sense vector yielding the highest similarity would underline the most meaningful sense describing the XML node label. This approach requires polynomial complexity: $O(|senses(\text{x}.\ell)|\times(|S_D(n)|+|S_D(s_P)|))$, where $|S_D(s_P)|$ designates the maximum context sphere cardinality for any sense (concept) in the semantic network.

Note that to our knowledge, existing approaches have seldom provide a complexity and time performance analysis of their WSD methods. Despite being performed offline, nonetheless, WSD time performance remains potent w.r.t. practicability, when indexing documents published on Web. The proposed approach has to be: i) effective in identifying the correct senses, but also ii) reasonably efficient in order to be practically applied to the large corpora of XML documents published online. Here, the complexity of our combined XML sense disambiguation approach is polynomial and simplified to $O(|X|\times|senses(\text{x}.\ell)|\times(|S_D(n)|+|S_D(s_P)|))$, where $|X|$ represents the number of nodes to be disambiguated in the target XML document.

4.2 XML Document Semantic Indexing

Having disambiguated all XML labels, the latter are replaced with their corresponding semantic concepts extracted from the reference semantic network (e.g., WordNet). Dedicated index structures (Concept-Doc and Concept-SN indexes [63-65], cf. Figure 3) are utilized to handle the mapping between XML document labels and knowledge base concepts. The output of the semantic document indexing process is a conceptual XML tree, i.e., an XML tree which labels consist of concepts with explicit semantic definitions (which is at the core of the vision of the Semantic Web: Extending the WWW by giving information well defined meaning [60]).

Consequently, we compute the semantic relatedness between each pair of node concepts in the XML tree. The idea is to produce a semantically weighted XML tree to be consequently exploited in keyword query processing (cf. Section 6). Here, various semantic similarity measures can be used (as briefly mentioned in the previous section): i) edge-based measures (computing semantic similarity based on the distance separating the concepts in the semantic network) [70], ii) node-based (computing semantic similarity based on the information content of each concept in the semantic network, w.r.t. a given text corpus) [38], and iii) gloss-based (comparing the glosses associated with each concept definition in the semantic network) [8]. Gloss-based approaches are particularly interesting in the context of WSD since they allow 'semantic relatedness' evaluation, which is a more general notion than '*semantic similarity*', including the latter as well as any kind of functional relation between terms [31] (e.g., *penguin* and *Antarctica* are not necessarily similar, but they are semantic related due to their *natural_ habitat* connection), particularly *antonymy* (e.g., *hot* and *cold* are semantically dissimilar since they have opposite meanings, but they are semantically related).

A simple example depicting the semantic indexing of a sample XML tree is shown in Figure 3. The sample XML document describes the movie *Rear Window*, one of *Alfred Hitchcock*'s masterpieces. While the XML labels seem meaningful and straightforward for a human user, nonetheless, they are highly ambiguous for a computer system. Most labels can be associated with more than 2 or 3 semantic senses (concepts) in WordNet reference. For instance, the label *Stewart* is associated with 2 semantic concepts: i) *James Stewart* (the leading actor who starred in *Rear Window*), and ii) *Dugald Stewart* (an 18th century Scottish philosopher). Likewise for most remaining labels in the input tree (e.g., *Kelly* underlines 3 semantic concepts, among which is *Grace Kelly*, the co-star of *Stewart* in *Rear Window*; *plot* underlines 4 different senses, among which *movie plot*, etc.).

Recall that semantic XML document indexing is performed offline, as a pre-processing step prior to query evaluation, and does not affect the online computational complexity of the approach.

5 Semantic Keyword Query Analysis

While semantic XML document analysis relies on the notion of XML context (e.g., the surroundings of a given node) in identifying the meanings of XML labels, nonetheless, semantic keyword query analysis differs in the lack of sufficient contextualization (keyword queries on the Web are usually 2-3 words long [13], which might not be sufficient in identifying a meaningful context [34, 40], cf. background in Section 2). To get round the lack of keyword contextualization in identifying meaningful query keyword senses, we introduce a method to *global query sense disambiguation*. Our proposal is based on the following assumption: *A keyword query on the Web usually conveys a certain global semantic meaning, reflecting a certain global information need*. Hence, rather than analyzing the individual senses of each query-term separately, considering each term's context information (similarly to most existing approaches, e.g., [29, 49]), we evaluate the aggregate semantic meaning of the query as a whole such that: the higher the semantic homogeneity of the query, the higher the consistency of the unified global semantic meaning conveyed by the query, and thus the more likely the query reflects the user's need. This is in accordance with the traditional assumption in WSD: *the most plausible assignment of senses to multiple co-occurring words is the one that maximizes the relatedness of meaning among the chosen senses* [42].

In short, we disambiguate the query as a whole, by i) pinpointing all possible configurations of query-term senses, and ii) consequently estimating a global semantic relatedness score (given a reference information source, e.g., WordNet) for all senses combined in each configuration. The configuration with the highest score would underline the most semantically meaningful query. Global query sense ranking can also be performed to identify the top most meaningful query sense configurations.

A major problem with the above approach is its computational complexity. In fact, computing semantic similarity/relatedness for all possible sense configurations for a set of lexical terms was shown to be intractable [42] due to its best case exponential complexity (i.e., $O(senses(k)^N)$ where N is the number of query keywords, and $senses(k)$ is the maximum number of senses per keyword). A few approximation methods have been proposed, such as computing pair-wise keyword similarities [42], and evaluating the similarity between each keyword sense and all remaining node senses [9]. Nonetheless, in contrast with existing approximation solutions, e.g., [9, 48], we introduce a sense disambiguation method to solve the computational complexity described above, producing optimal results similarly to the initial (exponential complexity) approach, while confining to polynomial complexity. We do so by transforming the problem of identifying all possible sense configurations, into that of identifying the shortest (semantic) path in a (semantically) weighted graph, using an adaptation of Dijkstra's shortest path algorithm [17]. In short, we capitalize on Dijkstra's polynomial computation approach to eliminate all unnecessary similarity computations, while still considering all possible query sense configurations.

Our query semantic analysis approach is described in the following sub-sections (cf. Figure 2). Sub-section 5.1 presents our global query sense disambiguation approach, while Sub-section 5.2 describes our semantic query representation method. Recall that linguistic normalization (including *tokenization*, *expansion*, *stop word removal*, and *stemming*) is similar for both XML documents labels and query keywords, and will not be discussed hereunder.

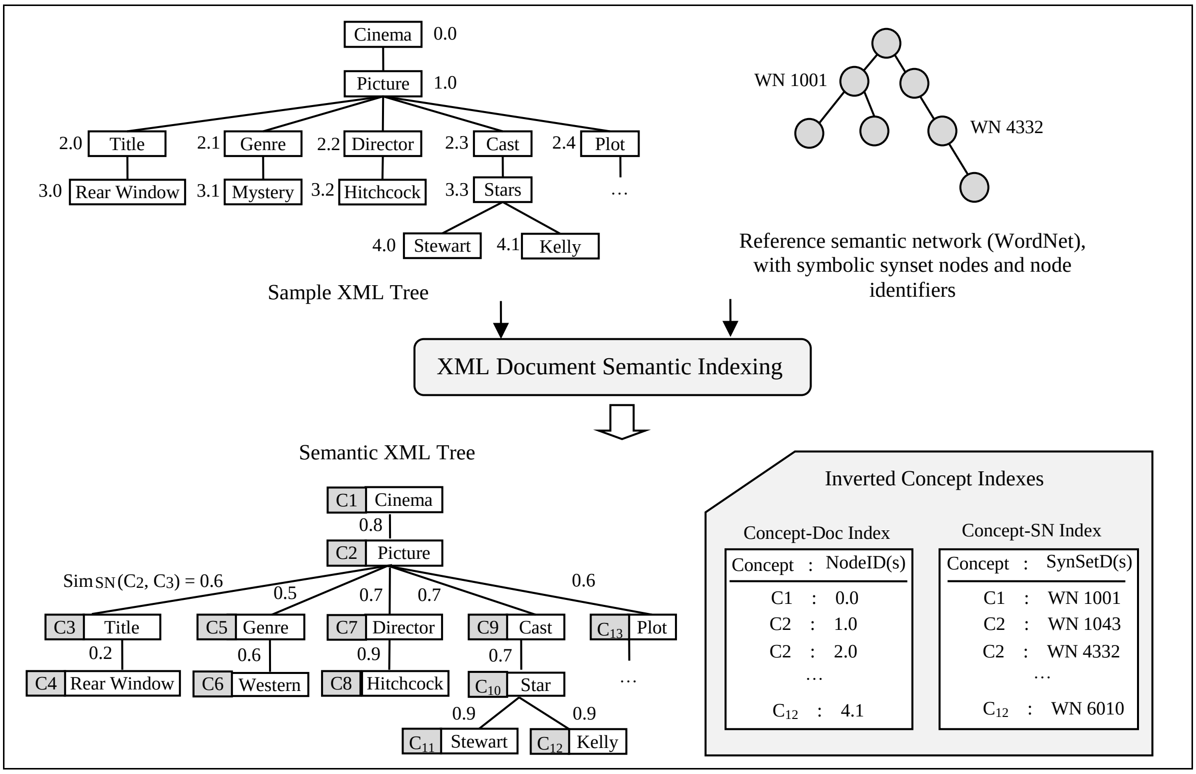


Figure 3: Semantic analysis of XML document

5.1 Query Global Sense Disambiguation

As mentioned previously, we assume that a keyword query on the Web conveys a certain global semantic information request. The main objective is to associate each query-term with the appropriate semantic sense (concept) maximizing global query sense homogeneity. To do so, we proceed as follows:

Step 1 – Identifying Keyword Senses: The first step consists in identifying the set of possible senses corresponding to each individual query-term (keyword). Formally, for each keyword k_r, we obtain a set of senses $S_r = \{s^r{}_1, s^r{}_2, \ldots, s^r{}_{|Sr|}\}$ where $s^r{}_i$ underlines the *i*th possible sense of keyword k_r extracted from the reference knowledge base (e.g., WordNet), and $|senses(k_r)|$ the maximum number of possible senses corresponding to k_r. This first step is similar to most existing semantic based approaches.

Step 2 – Building the Semantic Query Graph: Having identified all possible senses for each query-term, we construct a semantic graph where each node represents of a possible keyword sense. The graph is structured in different layers, such that:

i. Each layer corresponds to a query-term, and consists of nodes representing all possible semantic senses for that query-term,
ii. The layers are ordered following the order of appearance of the query-terms in the keyword query,
iii. Nodes within the same layer (i.e., representing possible senses for the same term) are not connected to each other. In fact, same layer nodes underline senses of the same query-term and thus should not appear simultaneously in the same path (i.e., same query sense configuration),
iv. Each pair of nodes corresponding to two consecutive layers (i.e., describing the possible meanings of two consecutive query-terms), are connected together via a weighted edge, underlining the semantic distance (as an inverse function of semantic similarity/relatedness) between node senses,
v. Two virtual *start* and *end* nodes are added to the graph, connected to the nodes of the first/last graph layers respectively, via edges of null distances. These are introduced to guide the execution process of our adapted shortest path discovery algorithm (described hereunder).

Step 3 - Identifying the Shortest Semantic Path: Consequently, the problem of identifying the most homogeneous configuration of query-term senses, simplifies to that of identifying the shortest semantic path in the semantic query graph. Here, we introduce an adaptation of Dijkstra's famous shortest path algorithm [17]. Our approach can be summarized as follows:

i. Initialize node distance scores such that: the *start node* score is set to zero, and all other node scores are set to infinity,
ii. Mark all nodes as *unvisited*, and set the *start node* as *current node*,
iii. For current node n_c, calculate the semantic distance with each of its connected nodes n_j in the consecutive layer, and preserve minimum distance scores, i.e., for each n_j, $Dist(n_j) = Min\{ Dist(n_c) + Weight(Egde(n_c, n_j)), Dist(n_j) \}$,
iv. When scores for all nodes connected to the current node n_c have been computed, n_c is marked as *visited*. A visited node would have a minimal and final distance score,
v. Select the *unvisited* node with the smallest distance score (from the initial node, considering all nodes in the graph) as the *current node* and continue from step 3,
vi. Terminate the algorithm when *end node* is deemed *visited*.

Consider keyword query *'Stewart Mystery Films'*. The corresponding semantic query graph, built based on query-term semantic senses extracted from WordNet [41], is depicted in Figure 5. Each graph layer corresponds to a query-term, and each node in a given layer underlines a semantic sense (concept) corresponding to the term at hand. The weight of an edge underlines the semantic distance between the connected nodes. Semantic distance can be computed as an inverse function of semantic similarity/relatedness, e.g., $Dist_{Sem} = 1 - Sim_{Sem}$. Recall that we adopt an aggregate semantic similarity/relatedness function combining *edge-based* methods [70], *node-based* methods [38], and *gloss-based* methods [8], w.r.t. WordNet. For ease of presentation, Figure 4 shows sample semantic weight values for some (and not all) of the graph edges (e.g., $weight(edge(n_1, n_4)) = 0.3$ indicating that semantic concepts *James Stewart* and *Mystery story* are more similar than *James Stewart* and *Enigma*, having $weight(edge(n_1, n_3)) = 0.5$).

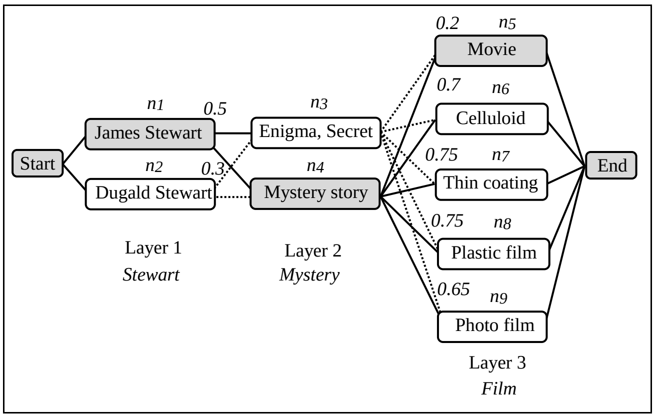


Figure 4: Semantic analysis of keyword query

The result of applying our adapted shortest path algorithm to the semantic query graph in Figure 4 is highlighted in the graph, and consists of nodes: n_1, n_4 and n_5. These underline the (WordNet) semantic concepts maximizing global query sense homogeneity: *James Stewart*, *Mystery story*, and *Movie*.

5.2 Semantic Query Representation

Having identified the best (i.e., most homogeneous) query sense configuration, we represent the query as a set of weighted semantic concepts (i.e., key-concepts), allowing the user to semantically expand the query, including additional concepts related to those originally conveyed by the query, in order to improve search result precision/recall.

Formally, a user keyword query Q consisting of a sequence of lexical keywords k_r, $Q = \prec k_1, k_2, \dots k_N \succ$ is transformed into a semantic query representation Q_{Sem} consisting of a set of weighted concepts, $Q_{Sem}(D)= \{(c_1, w_1), (c_2, w_2), \dots, (c_M, w_M)\}$ where c_i is a key-concept, w_i is the weight of c_i, and D is the query semantic expansion parameter. The number of resulting key-concepts $M \geq N$ since additional key-concepts can be added following the user-chosen D expansion parameter as explained in the following. Semantic query expansion is performed using our *Sphere-Ring* model (cf. Section 4.1) to consider the semantic context[2] of each query key-concept in the reference semantic network (e.g., WordNet). The idea is to expand the query with additional concepts within the semantic vicinity of the original query key-concepts. Following our *Sphere-Ring* model, a semantic *ring* $R_d(c)$ w.r.t. to a given concept c consists of the set of concept nodes, in the reference semantic network, situated at a specific distance d from the target concept node c. The semantic context sphere $S_D(c_i)$ encompasses all semantic rings contained at distances lesser or equal to the size (diameter D) of the sphere, such that $S_D(c) = \{\text{all } R_d(c) \ / \ d \leq D\}$. The sphere context size is specified by the user as a query semantic expansion parameter:

- For $D = 0$, the query is represented with it original key-concepts, associated maximum (unit, =1) weights,
- For $D > 0$, the query is expanded with concepts situated within each original key-concept's semantic sphere (in the reference semantic network). Expanded query concepts are weighted such that concepts farther away from the semantic sphere center have a larger semantic distance w.r.t. the sphere's center, and hence should have a lesser impact on the query's semantic meaning. Following our *Sphere-Ring* context model, concept weights can be computed following the sizes of the sphere rings to which they correspond, such that the larger the sphere ring radius, the lesser the concept weight (e.g., a given weight decay function could be computed as $weight(c_i) = w_i = \frac{1}{1+d} \in [0, 1]$ having $c_i \in R_d(c) \subset S_D(c)$). Note that parameter D can be normalized in the $[0, 1]$ interval, following the maximum depth of the reference semantic network SN at hand (e.g., $\frac{D}{Depth(SN)}$), to simplify the user's task in specifying the expansion threshold.

Consider for instance the sample keyword query Q = *'Stewart Mystery Films'*:

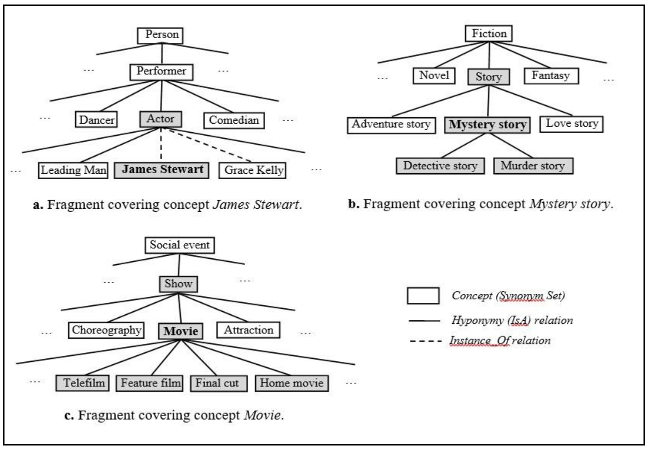
- For $D = 0$, $Q_{Sem}(0) = \{(James\ Stewart, 1), (Mystery\ story, 1), (Movie, 1)\}$,
- For $D = 1$, the resulting query representation includes all semantic concepts appearing in the unit ($D=1$) semantic context spheres of each original key-concept. Here, following the WordNet extracts in Figure 5, the semantic context of

[2] The semantic contexts of query concepts can be determined, since the latter have already been disambiguated (as opposed to the pre-disambiguation keyword query where the semantic meanings of query-terms were undefined).

concept *James Stewart* includes concept *Actor* (cf. Figure 5.a). Likewise, the semantic context of concept *Mystery movie* includes *Story*, *Detective story* and *Murder story* (Figure 5.b). The semantic context of concept *Movie* includes *Show*, and *17* children (hyponym) concepts including *Telefilm*, *Feature film*, *Final cut*, *Home movie*, etc., (the remaining child concepts are omitted here for ease of presentation, cf. Figure 5.c). The weights of all expanded concepts are equal to $\frac{1}{1+d} = \frac{1}{1+D} =$ 0.5, following our adopted decay function. Hence, the semantic query becomes:

$Q_{Sem}(1)$={ (*James Stewart*, 1), (*Actor*, 0.5), (*Mystery story*, 1),
(*Story*, 0.5), (*Detective story*, 0.5),
(*Murder story*, 0.5), (*Movie*, 1), (*Show*, 0.5),
(*Telefilm*, 0.5), (*Final cut*, 0.5), (*Home movie*, 0.5) }

The time complexity of our global query disambiguation approach comes down to that of the shortest path computation process, which comes down to almost linear $O(N\times\log(N))$ time where $N=|S_D(c)|\times|Q|\times|senses(k_r)|$. The latter simplifies to $N=|S_D(c)|\times|senses(k_r)|$ since $|Q|$ is usually limited to 2-3 keywords [13] and can be omitted as a fixed parameter.



a. Fragment covering concept *James Stewart*.

b. Fragment covering concept *Mystery story*.

c. Fragment covering concept *Movie*.

Figure 5: Taxonomic fragments extracted from WordNet, covering the key-concepts considered in our example

6 Semantic Keyword Query Processing

Having transformed the XML document collection and the keyword query into meaningful semantic representations, XML keyword query processing comes down to identifying and ranking the most relevant semantic XML sub-structures encompassing the semantic key-concepts in the query. Our querying method is based on a structural clustering technique to group together key-concept occurrences, in the XML data collection, which are structurally close. Our objective is to identify and rank the most prominent candidate answer sub-trees, in the XML data set, containing related occurrences of query key-concepts. Our query algorithm is shown in Figure 6 and is described below:

Step 1 - Identifying concept occurrences: The first step consists in pinpointing the XML nodes, in the data collection, containing occurrences of the query key-concepts.

Step 2 - Performing XML node clustering: Having identified the XML nodes encompassing key-concept occurrences, we perform structural clustering [45] to group together the XML nodes which are closest in the XML tree. The algorithm is applied on the weighted distances separating concept occurrences (cf. Section 4).

Step 3 – Constructing Answer Trees: We construct candidate answer trees based on the XML node clusters. An answer tree consists of the sub-tree rooted at the lowest common ancestor of all concept occurrences in the corresponding cluster.

Step 4 – Ranking Answer Trees: Having identified the candidate answer sub-trees, we rank them following their relevance to the query. Here, we utilize an integrated function combining various ranking criteria including i) weights of semantic concepts; ii) answer tree size (compactness), iii) common usage of senses (e.g., WordNet estimates the average usage frequency of word meanings in the English language, following the Brown corpus [24]), where the most commonly used senses are deemed more relevant in ranking results [40]. Other weighting functions can be used.

```
Algorithm SemanticQueryProcessing
Input:  T      // Semantic XML tree
        K      // Set of query selection terms
        D      // Sphere diameter designating query context size
Ouput: N_Out   // List of ranked trees from T designating query answers
Begin
  N_Out = ϕ
  Step 0:  S = getSemanticQuerySenses(K)  // Global disambiguation
  For each term s_i ∈ S                   // For each keyword sense
  {
     Step 1: n_In = getNodeID(s_i, T)     // Identify concept occurrences
     Step 2: SP = PerformClustering(n_In, D, T)
     Step 3: N_init = constructAnswerTree(SP, T)
     Step 4: N_Out = rankAnswerTree(N_init, T)
  }
  Return N_Out
End
```

Figure 6: Pseudo-code of semantic keyword query processing

Note that the complexity of the query processing algorithm comes down to the complexity of the structure clustering algorithm in Step 2. We utilize Lloyd's heuristic algorithm [39] to bound clustering complexity to $O(N \times C \times I)$ where N is the number of XML nodes to be clustered, C the number of produced clusters, and I the number of iterations to reach convergence.

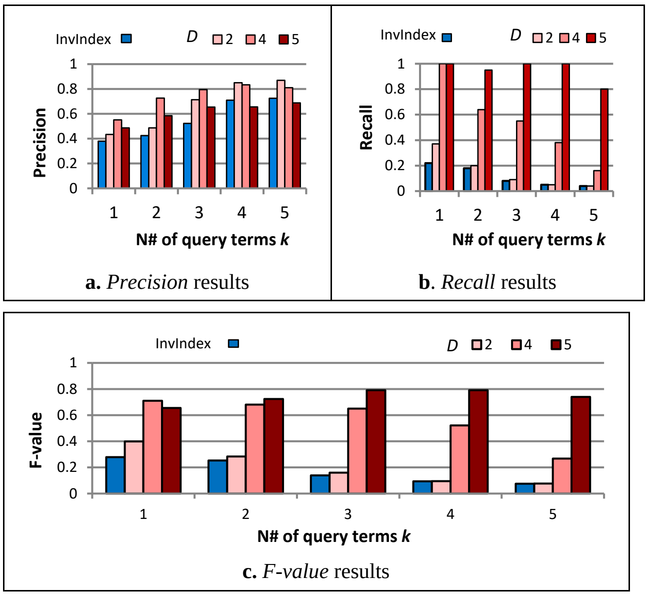
7 Preliminary Experiments

We conducted preliminary experiments to test and evaluate our approach. We used a collection of 80 test documents gathered from several data sources[3] having different properties. Target XML nodes were first subject to manual disambiguation (12-to-13 nodes were randomly selected per document, yielding a total of 1000 target nodes, allowing human testers to annotate each node by choosing appropriate senses from WordNet) followed by automatic disambiguation. We then compared user and system generated senses to compute *precision*, *recall* and *f-value* scores.

[3] Shakespeare collection http://metalab.unc.edu/bosak/xml/eg/shaks200.zip, Amazon product files simply-amazon.com/content/XML.html SIGMOD Record http://www.acm.org/sigmod/xml Niagara collection http://www.cs.wisc.edu/niagara/

Table 1: Sample test queries used in our experiments

Query *Q1*		Query *Q2*	
ID	**Terms**	**ID**	**Terms**
Q1_1	“music”	Q2_1	“play”
Q1_2	“music”, romance”	Q2_2	“play”, “theator”
Q1_3	“music”, romance”, “dinner”	Q2_3	“play”, “theator”, “scene”
Q1_4	“music”, romance”, “dinner”, “trip”	Q2_4	“play”, “theator”, “scene”, “hero”
Q1_5	“music”, romance”, “dinner”, “trip”, “Paris”	Q2_5	“play”, “theator”, “scene”, “hero”, “climax”



a. *Precision* results

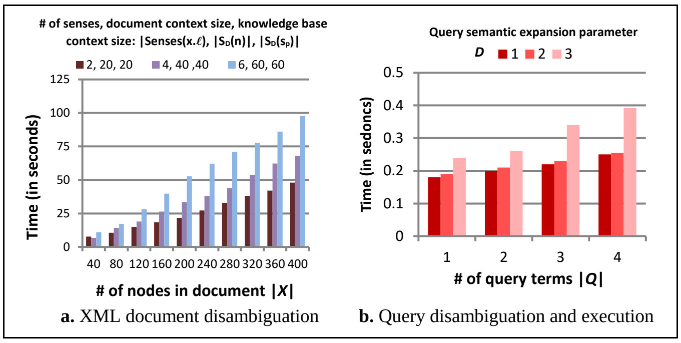
b. *Recall* results

c. *F-value* results

Figure 7: Comparing precision (PR), recall (R) and F-value results with legacy inverted index syntactic search

We first tested the effectiveness of our approach considering its different features and configurations: i) the properties of XML data (w.r.t. ambiguity and structure), and ii) context size (sphere neighborhood radius). Results in Figure 7 show that precision levels increase with the number of query terms *k*. This is due to the human testers’ expectations: given that queries are expanded versions of one another, result quality is evaluated based on the user’s intent: which is expressed with the most expanded (i.e., most expressive) query (e.g., *Q1_5* and *Q2_5*). One can realize that using fewer query terms produces lower precision levels, which is due to the system returning more results which are (semantically related to the query terms but which are) not necessary related to the user’s intent. As for recall, one can realize that levels steadily increase with concept depth *D*, where the number of correct (i.e., user expected) results returned by the system increases as more semantically related terms are covered in the querying process. F-value results increase with the increase of context depth *D*, and they slightly decrease with the increase of the number of query keywords *k*. This confirms the precision and recall results, where the determining factor affecting retrieval quality remains context depth *D*. An increase in the number of keywords *k* tends to reduce system recall with higher values of *k* (queries becoming very selective, thus missing some relevant results). F-value levels are significantly higher than those obtained with the legacy inverted index, highlighting a clear improvement over syntactic retrieval quality.

We also evaluated our solution’s almost linear efficiency. Results in Figure 8 highlight the polynomial (almost linear) complexities of both our (offline) XML document disambiguation and (online) global query disambiguation approaches, considering different parameter configurations for both processes. Results in Figure 8.b show total query execution time including online disambiguation, by varying both the number of keywords and query semantic depth *D* (i.e., semantic context size).



a. XML document disambiguation

b. Query disambiguation and execution

Figure 8: XML document disambiguation time (a) and query processing (disambiguation and execution) time (b)

8 Conclusion

In this paper, we introduce the building blocks of a new approach for XML keyword search allowing to transform both XML documents and keyword queries into semantic representations, using semantic concepts in a reference knowledge base. We describe two approaches for i) offline context-based XML document disambiguation and ii) online global keyword query disambiguation, both designed to run in almost linear time. Our solution is: i) fully automated, compared with existing interactive solutions which require user input to manually identify the intended query senses e.g., [29, 49], and ii) tractable (of almost linear time) and thus reasonably applicable on the Web, compared with polynomial or exponential solutions, e.g., [18, 47].

We are currently investigating the integration of semantic-aware indexing capabilities [63-65] and different clustering algorithms to form XML answer trees [27, 61]. This would provide more opportunities toward both speed-ups and semantic-based filtering. We are also investigating the use of alternative knowledge sources such as Google [1], Wikipedia [69], and FOAF [4] to acquire a wider word sense coverage, and explore our approach in practical applications, namely semantic-aware document and schema matching [66, 67], RSS news feed merging [52, 53], affective blog analysis [21, 22], social event detection [3, 5], and semantic relations’ identification from social media data [2]. On the long run, we aim to investigate word embeddings and learning statistical distributions in a corpus [28, 73], to infer semantics without the need for predefined knowledge bases.

REFERENCES

[1] Mehmet Ali Abdulhayoglu and Bart Thijs, 2017. *Use of ResearchGate and Google CSE for Author Name Disambiguation.* Scientometrics, 111(3): 1965-1985.

[2] Minale Ashagrie Abebe, Joe Tekli, Fekade Getahun, Richard Chbeir, Gilbert Tekli, 2020. *Generic Metadata Representation Framework for Social-based Event Detection, Description,& Linkage.* Knowledge Based Systems, 188.

[3] Minale Ashagrie Abebe, Joe Tekli, Fekade Getahun, Richard Chbeir, Gilbert Tekli, 2017. *Overview of Event-Based Collective Knowledge Management in Multimedia Digital Ecosystems.* International Conference of Signal Image Technology and Internet-based Systems (SITIS'17), pp. 40-49.

[4] Muhammad Amith, Kayo Fujimoto, Rebecca Mauldin, Cui Tao, 2020. *Friend of a Friend with Benefits Ontology (FOAF+): Extending a Social Network Ontology for Public Health.* BMC Medical Informatics and Decision Making, 20-S(10): 269.

[5] Minale Ashagrie Abebe, Joe Tekli, Fekade Getahun, Gilbert Tekli, Richard Chbeir, 2016. *A General Multimedia Representation Space Model toward Event-based Collective Knowledge Management.* IEEE Inter. Conf. on Computational Science and Engineering (CSE 2016), pp. 512-521

[6] Antonia Azzini, Célia da Costa Pereira, Mauro Dragoni, Andrea Tettamanzi, 2012. *A Neuro-Evolutionary Corpus-based Method for Word Sense Disambiguation.* IEEE Intelligent Systems, 27(6): 26-35.

[7] Ricardo Baeza-Yates and Berthier Ribeiro-Neto, 2011. *Modern Information Retrieval: The Concepts and Technology behind Search.* ACM Press Books, p. 944.

[8] Satanjeev Banerjee and Ted Pedersen, 2003. *Extended Gloss Overlaps as a Measure of Semantic Relatedness.* Inter. Joint Conf. on Artificial Intell. (IJCAI), 805-810.

[9] Mustapha Baziz, Mohand Boughanem, and Salam Traboulsi, 2005. *A Concept-based Approach for Indexing Documents in IR.* INFORSID'05, pp. 489-504.

[10] Carlos Bobed and Eduardo Mena, 2016. *QueryGen: Semantic Interpretation of Keyword Queries over Heterogeneous Information Systems.* Information Sciences, 329: 412-433.

[11] Hamed R. Bonab, Mohammad Aliannejadi, John Foley, and James Allan, 2019. *Incorporating Hierarchical Domain Information to Disambiguate Very Short Queries.* Inter. Conf. on the Theory of Information Retrieval (ICTIR'19), pp. 51-54.

[12] Andrew Burton-Jones, Veda C. Storey, Vijayan Sugumaran, and Sandeep Purao, 2003. *A Heuristic-Based Methodology for Semantic Augmentation of User Queries on the Web.* International Conference on Conceptual Modeling (ER'03), pp. 476–489.

[13] Andrea Calì, Davide Martinenghi, Riccardo Torlone, 2016. *Keyword Queries over the Deep Web.* International Conference on Conceptual Modeling (ER'16), pp. 260-268.

[14] Devendra Singh Chaplot and Ruslan Salakhutdinov, 2018. *Knowledge-based Word Sense Disambiguation using Topic Models.* AAAI Conference on Artificial Intelligence (AAAI'18), pp. 5062-5069.

[15] Nathalie Charbel, Joe Tekli, Richard Chbeir, and Gilbert Tekli, 2015. *Resolving XML Semantic Ambiguity.* Inter. Conf. on Extending DB Technology (EDBT), 277-288.

[16] Dunren Che, Tok Wang Ling, and Wen-Chi Hou, 2012. *Holistic Boolean-Twig Pattern Matching for Efficient XML Query Processing.* IEEE Trans. on Knowledge and Data Engineering (TKDE), 24(11): 2008-2024.

[17] Thomas Cormen, Charles Leiserson, Ronald Rivest, and Clifford Stein, 2001. *Introduction to Algorithms (Second ed.) - Section 24.3: Dijkstra's Algorithm.* MIT Press and McGraw-Hill, pp. 595–601.

[18] Luis M. de Campos, Juan M. Fernández-Luna, Juan Huete, and Eduardo Vicente-López, 2013. *XML Search Personalization Strategies using Query Expansion, Reranking and a Search Engine Modification.* ACM Symposium on Applied Computing (SAC'13), pp. 872-877.

[19] Elena Demidova, Xuan Zhou, Irina Oelze, and Wolfgang Nejdl, 2010. *Evaluating Evidences for Keyword Query Disambiguation in Entity Centric Database Search.* Inter. Conf. on Database and Expert Systems Applications (DEXA'10), pp. 240-247.

[20] Angelo Di Iorio, Silvio Peroni, Francesco Poggi, and Fabio Vitali, 2012. *A First Approach to the Automatic Recognition of Structural Patterns in XML Documents.* ACM Symposium on Document Engineering, pp. 85-94.

[21] Mireille Fares, Angela Moufarrej, Eliane Jreij, Joe Tekli, and William Grosky, 2019. *Difficulties and Improvements to Graph-based Lexical Sentiment Analysis using LISA.* Cognitive Computing (ICCC'19), pp. 28-35

[22] Mireille Fares, Angela Moufarrej, Eliane Jreij, Joe Tekli, and William Grosky, 2019. *Unsupervised Word-level Affect Analysis and Propagation in a Lexical Knowledge Graph.* Knowledge-Based Systems. 165: 432-459.

[23] Kostas Fragos, 2013. *Modeling WordNet Glosses to Perform Word Sense Disambiguation.* International Journal of Artificial Intelligence Tools, 22(2).

[24] Nelson Francis and Henry Kucera, 1982. *Frequency Analysis of English Usage.* Houghton Mifflin, Boston, 561 p.

[25] Jianfeng Gao, Shasha Xie, Xiaodong He, and Alnur Ali, 2012. *Learning Lexicon Models from Search Logs for Query Expansion.* Empirical Methods in Natural Language Process. (EMNLP'12), pp. 666-676.

[26] Jens Graupmann, Ralf Schenkel, Gerhard Weikum, 2005. *The SphereSearch Engine for Unified Ranked Retrieval of Heterogeneous XML and Web Documents.* Inter. Conf. on Very Large Databases (VLDB), pp. 529-540.

[27] Ramzi Haraty, Mohamad Dimishkieh, and Mehedi Masud, 2015. *An Enhanced k-Means Clustering Algorithm for Pattern Discovery in Healthcare Data.* Intelligent Journal on Distributed Sensor Networks, 11: 615740:1-615740:11.

[28] Ramzi Haraty and Rouba Nasrallah, 2019. *Indexing Arabic Texts using Association Rule Data Mining.* Library Hi Tech, 37(1): 101-117.

[29] Donna Harman, 2017. *Towards Interactive Query Expansion.* SIGIR Forum, 51:79-89.

[30] Martin Holub, Vincent Kríz, Silvie Cinková, and Eckhard Bick, 2012. *Tailored Feature Extraction for Lexical Disambiguation of English Verbs Based on Corpus Pattern Analysis.* Inter. Conf. on Computational Linguistics (COLING), 1195-1210.

[31] Pascual Julián Iranzo and Fernando Sáenz-Pérez, 2012. *Implementing WordNet Measures of Lexical Semantic Similarity in a Fuzzy Logic Programming System.* Theory and Practice of Logic Programming, 21(2): 264-282.

[32] Maryam Kamvar and Shumeet Baluja, 2006. *A Large Scale Study of Wireless Search Behavior: Google Mobile Search.* SIGCHI Conference on Computer-Human Interaction, pp. 701–709.

[33] Ritesh Kumar, Bhanodai Guggilla, and Rajendra Pamula, 2019. *Book Search using Social Information, User Profiles and Query Expansion with Pseudo Relevance Feedback.* Applied Intelligence, 49(6): 2178-2200.

[34] Sunjae Kwon, Dongsuk Oh, and Youngjoong Ko, 2021. *Word Sense Disambiguation based on Context Selection using Knowledge-based Word Similarity.* Information Processing and Management, 58(4): 102551.

[35] Claudia Leacock and Martin Chodorow, 1998. *Combining Local Context and WordNet Similarity for Word Sense Identification.* The MIT Press, pp. 265-283.

[36] Yunyao Li, Huahai Yang, and H. V. Jagadish, 2005. *NaLIX: an Interactive Natural Language Interface for Querying XML.* International ACM Conference on Management of Data (SIGMOD'05), pp. 900-902.

[37] Yunyao Li, Huahai Yang, and H. V. Jagadish, 2006. *Term Disambiguation in Natural Language Query for XML.* International Conference on Flexible Query Answering Systems (FQAS'06), pp. 133–146.

[38] Dekang Lin, 1998. *An Information-Theoretic Definition of Similarity.* International Conference on Machine Learning (ICML'98), pp. 296-304.

[39] Stuart Lloyd, 1982. *Least Squares quantization in PCM.* IEEE Transactions on Information Theory, 28(2):129-137.

[40] Federica Mandreoli and Riccardo Martoglia, 2011. *Knowledge-based Sense Disambiguation (almost) for all Structures.* Information Systems, 36(2): 406-430.

[41] George Miller, 1990. *WordNet: An Online Lexical Database.* International Journal of Lexicography, 3(4).

[42] Saif Mohammad, Graeme Hirst, and Philip Resnik, 2007. *Tor, TorMd: Distributional Profiles of Concepts for Unsupervised Word Sense Disambiguation.* SemEval@ACL'07, pp. 326-333.

[43] Roberto Navigli, 2009. *Word Sense Disambiguation: a Survey.* ACM Computing Surveys, 41(2):1–69.

[44] Roberto Navigli and Paola Velardi, 2005. *Structural Semantic Interconnections: A knowledge-based Approach to Word Sense Disambiguation* IEEE Trans. on Pattern Analysis and Machine Intelligence, 27(7):1075–1086.

[45] Roberto Navigli and Giuseppe Crisafulli, 2010. *Inducing Word Senses to Improve Web Search Result Clustering.* Inter. Conf. on Empirical Methods in NLP, 116–126.

[46] Roberto Navigli and Paola Velardi, *An Analysis of Ontology-based Query Expansion Strategies.* Inter. Joint Conf. on Artificial Intelligence (IJCAI'03), pp. 42-49.

[47] Amjad Qtaish and Mohammad Alshammari, 2019. *A Narrative Review of Storing and Querying XML Documents using Relational Database.* Journal of Information & Knowledge Management, 18(4): 1950048:1-1950048:28.

[48] Philip Resnik, 1995. *Disambiguating Noun Groupings with Respect to WordNet Senses.* In Proceedings of the 3rd Workshop on Large Corpora, pp. 54-68.

[49] Tony Russell-Rose, Philip Gooch, and Udo Kruschwitz, 2021. *Interactive Query Expansion for Professional Search Applications.* CoRR abs/2106.13528.

[50] Sanasam Ranbir Singh, Hema Murthy, and Timothy Gonsalves, 2010. *Dynamic Query Expansion based on User's Real Time Implicit Feedback.* Knowledge Discovery and Info. Retrieval (KDIR'10), pp.112-121.

[51] Nadia Soudani, Ibrahim Bounhas, and Sawssen Ben Babis, 2018. *Ambiguity Aware Arabic Document Indexing and Query Expansion: A Morphological Knowledge Learning-Based Approach.* Florida AI Research Society Conf. (FLAIRS), 230-235.

[52] Fekade Taddesse, Joe Tekli, Richard Chbeir, Marco Viviani, and Kokou Yétongnon, 2010. *Semantic-based Merging of RSS Items.* World Wide Web, 13(1-2): 169-207.

[53] Fekade Getahun, Joe Tekli, Richard Chbeir, Marco Viviani, and Kokou Yétongnon, 2009. *Relating RSS News/Items.* Inter. Conf. on Web Engineering (ICWE'09), 44-452.

[54] Andrea Tagarelli and Sergio Greco, 2010. *Semantic Clustering of XML Documents.* ACM Transactions on Information Systems, 28(1):3.

[55] Andrea Tagarelli, Mario Longo, and Sergio Greco, 2009. *Word Sense Disambiguation for XML Structure Feature Generation.* European Semantic Web Conf., pp. 143–157.

[56] Kamal Taha and Ramez Elmasri, 2008. *CXLEngine: A Comprehensive XML Loosely Structured Search Engine.* EDBT workshop on Database Technologies for Handling XML Information on the Web (DataX'08), pp. 37-42.

[57] Kamal Taha and Ramez Elmasri, 2010. *XCDSearch: An XML Context-Driven Search Engine.* IEEE Transactions on Knowledge and Data Engineering, 22(12):1781-1796.

[58] Wolfgang Tannebaum and Andreas Rauber, 2014. *Using Query Logs of USPTO Patent Examiners for Automatic Query Expansion in Patent Searching.* Information Retrieval, 17(5-6): 452-470.

[59] Joe Tekli, 2016. *An Overview on XML Semantic Disambiguation from Unstructured Text to Semi-Structured Data: Background, Applications, & Ongoing Challenges.* IEEE Transactions on Knowledge and Data Engineering (TKDE), 28(6):1383-1407.

[60] Joe Tekli, Antoine Abou Rjeily, Richard Chbeir, Gilbert Tekli, Pelagie Houngue, Kokou Yétongnon, and Minale Ashagrie Abebe, 2013. *Semantic to Intelligent Web Era: Building Blocks, Applications, and Current Trends.* International Conference on Managment of Emergent Digital EcoSystems (MEDES'13), pp. 159-168.

[61] Jimmy Tekli, Bechara al Bouna, Youssef Bou Issa, Marc Kamradt, and Ramzi Haraty, 2018. *(k, l)-Clustering for Transactional Data Streams Anonymization.* Information Security Practice and Experience, pp. 544-556.

[62] Joe Tekli, Nathalie Charbel, and Richard Chbeir, 2016. *Building Semantic Trees from XML Documents.* Journal of Web Semantics (JWS), 37–38:1–24.

[63] Richard Chbeir, Yi Luo, Joe Tekli, Kokou Yétongnon, Carlos Raymundo Ibañez, Agma J. M. Traina, Caetano Traina Jr., and Marc Al Assad, 2015. *SemIndex: Semantic-Aware Inverted Index.* Symposium on Advances in Databases and Information Systems (ADBIS), pp. 290-307.

[64] Joe Tekli, Richard Chbeir, Agma J. M. Traina, and Caetano Traina Jr., 2019. *SemIndex+: A Semantic Indexing Scheme for Structured, Unstructured, and Partly Structured Data.* Knowledge-Based Systems, 164:378-403.

[65] Joe Tekli, Richard Chbeir, Agma J. M. Traina, Caetano Traina, Kokou Yétongnon, Carlos Raymundo Ibañez, Marc Al Assad, and Christian Kallas, 2018. *Full-fledged Semantic Indexing and Querying Model Designed for Seamless Integration in Legacy RDBMS.* Data and Knowledge Engineering, 117: 133-173.

[66] Joe Tekli, Richard Chbeir, and Kokou Yétongnon, 2007. *A Fine-grained XML Structural Comparison Approach.* International Conference on Conceptual Modeling (ER), pp. 582-598.

[67] Joe Tekli, Richard Chbeir, and Kokou Yétongnon, 2007. *Structural Similarity Evaluation between XML Documents and DTDs.* Inter. Conf. on Web Info. Systems Eng. (WISE), pp. 196-211.

[68] Martin Theobald, Ralf Schenkel, and Gerhard Weikum, 2003. *Exploiting Structure, Annotation, and Ontological Knowledge for Automatic Classification of XML Data.* ACM SIGMOD International Workshop on Databases (WebDB'03), pp. 1-6.

[69] Hai-Lun Tu, Peichen Ho, Jason S. Chang, and Li-Guang Chen, 2019. *Word Sense Disambiguation Using Wikipedia Link Graph.* IEEE BigData'19, pp. 6235-6236.

[70] Zhibiao Wu and Martha Stone Palmer, 1994. *Verb Semantics and Lexical Selection.* 32nd Annual Meeting of the Associations of Computational Linguistics, pp. 133-138.

[71] Dan Yang, Derong Shen, Ge Yu, Yue Kou, and Tiezheng Nie, 2013. *Query Intent Disambiguation of Keyword-Based Semantic Entity Search in Dataspaces.* Journal of Computer Science and Tech., 28:382–393.

[72] Jeonghee Yi, Farzin Maghoul, and Jan Pedersen, 2008. *Deciphering Mobile Search Patterns: a Study of Yahoo! Mobile Search Queries.* Web Wide Web Conference (WWW'08), pp. 257-266.

[73] Hanwang Zhang, Xindi Shang, Huan-Bo Luan, Meng Wang, and Tat-Seng Chua, 2017. *Learning from Collective Intelligence: Feature Learning using Social Images & Tags.* ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 13(1):1.