Università di Torino

XXII ciclo di dottorato in
Scienza e Alta Tecnologia
Indirizzo Informatica

Adaptation of Hierarchical Meta-Data for Efficient Large Data Set Exploration

Author:
Mario Cataldi

Advisor:
Prof. Maria Luisa Sapino

November 19, 2010

Abstract

Meta-data hierarchies are playing central roles in the development and deployment of many data applications, especially in the Web. They can be defined as structured information that describes, explains, locates, and makes it easier to retrieve, use and manage information resources. In fact, they embody formalized knowledge and define aggregations between concepts/categories (in a given domain) facilitating the organization of the data and making the contents easily accessible to the users of these applications.

Since hierarchical meta-data have significant roles in the data annotation, search, and navigation, they are often carefully engineered; however, especially in dynamically evolving domains, they do not necessarily reflect the content knowledge. In fact, the stagnant nature of meta-data may fail to timely capture the dynamic change of the relevant data contents. Moreover, when the users interests are highly focused, available meta-data, which are usually designed from domain experts for broad coverage of concepts in a given application domain, may fail to reflect details within the users foci of interest.

Thus, in this thesis, we ask and answer, in the positive, the following question: "*is there a feasible approach to efficiently and effectively adapt a given meta-data hierarchy to changing usage contexts?*".

Based on these considerations, we propose a set of novel adaptation approaches for re-structuring existing meta-data hierarchies to varying application contexts and different data formats, and we evaluate the proposed schemes relying on different data collections.

Moreover, we leverage these adapted meta-data structures for innovative data exploration methods that use domain-specific

concepts (taken from the properly adapted structures) as well as relevant corpus terms that are characterizing the data collection. These exploration methods provide novel navigation mechanisms and improve the efficiency of the standard exploration process, using the natural relationships expressed by the given contents in addition to those formalized by the associated adapted metadata structures.

Acknowledgments

I would like to express my gratitude to my supervisor, Prof. Maria Luisa Sapino, whose expertise, understanding, and patience, added considerably to my personal and professional activities. I appreciate her knowledge and skills in many areas and her continuous assistance in all my research activities. I also need to deeply thank the other members of my research group, Dr. Claudio Schifanella, and Luigi Di Caro for the assistance and support they provided at all levels of my life at the Department.

A very special thanks goes out to Prof. K. Selçuk Candan, who truly made a difference in my Ph.D. research works. It was through his persistence, understanding and obstinacy that I completed my Ph.D. and was encouraged to continuously searching for improvements in my research activities. I doubt I will ever reach his professional level, but I owe him my sincere gratitude.

I would like to thank the reviewers of this Ph.D. dissertation, Prof. Vincent Oria of the New Jersey Institute of Technology and Dr. Jia-Yu Pan of Google Research, for their insightful comments and suggestions for further developments of this work.

I would also like to thank all my friends at the Computer Science Department, particularly Sara, for our philosophical debates, movie discussions and venting of frustrations, which, in many cases, saved at least my mental sanity. I also need to thank my family for the support they provided me during my Ph.D. activities and, finally, I must acknowledge Giulia, without whose love, encouragement and assistance, I would have never, ever, finished this thesis.

Contents

Chapter 1

Introduction

Several Information Retrieval (IR) systems are used on an everyday basis by a wide variety of users. In fact, considering the enormous amount of available data sets in the web, often large in volume and complex in structure (multidimensional and/or hierarchical), it is necessary to assist the user in handling these large amount of data. Therefore, new access and exploration methods are needed in order to guide the users through the data, highlighting (if and when requested) hidden relationships among them and helping the users discover new knowledge previously hardly identifiable.

Data Management can be defined as the set of strategies and procedures that properly manage the data lifecycle in order to prepare them to be easily accessible by the users. This field includes various approaches and techniques that consider the input format data and the aims of their use. In particular, we distinguish among three important steps[1]: *Data analysis and design*, *Meta-data association and management* and *Data exploration* strategies.

The first step is the analysis of data; it consists of inspecting, cleaning, transforming, and modelling the given data with the goal of selecting the more suitable formats in order to make them easily accessible by the architecture that will allow the exploration process on them; within this preliminary phase, a cleaning procedure is generally needed in order to inspect and remove useless data. Many IR systems dealing with textual data also use stop word elimination and stemming in this phase; in fact in the literature, many investigations suggest that stop word removal improves retrieval sig-

[1] In this thesis we will not consider other important aspects of the data lifecycle, as for example data security or maintenance. We focus our research on innovative preprocessing and meta-data management operations for novel exploration strategies, relying on available techniques for other operations.

nificantly. Many techniques have been proposed in order to properly design the format of the data and formalize the content knowledge.

This set of processes defines the structure and the content of the information handled by the system (and, consequentely, by the users). But the data themselves (even if carefully designed and formalized) are often not enough to provide smart and fast access to them. In fact, most of the IR systems generally enrich the considered data by *meta-data*, that provide supervised knowledge about the contents that can help the retrieval process. Meta-data can be defined as structured information that describes, explains, locates, and makes it easier to retrieve, use and manage an information resource. For this reason, meta-data is usually defined as *data about data*.

Some of the most used meta-data types include controlled vocabularies, taxonomies, thesauri, data dictionaries and registries. Meta-data can be one dimensional, where each element is semantically separated from other elements, or hierarchical where evident relationships exist among the elements.

The degree of meta-data is generally referred as its granularity. Meta-data are generally created with the help of supervised systems (i.e. with the help of domain experts) in order to provide information that an unsupervised method is not able to identify and interpret. Meta-data can serve many important purposes, including data browsing, transfer and documentation. Meta-data can be organized into several levels ranging from a simple listing of basic information about available data to detailed documentation about an individual data set. An important reason for creating meta-data is to facilitate discovery of relevant information. They can organize text streams, facilitate interoperability among different systems, and support archiving and preservation. The importance of meta-data in running queries is absolutely central to the purpose of many IR systems both at design time and at run time. Indeed, by associating the data to one or many elements of the meta-data, they can easily index the contents and also highlight semantic relationships previously hardly identifiable.

Considering all these aspects, while there are many strategies for organizing data, hierarchical meta-data categorization, usually implemented through a pre-determined taxonomical structure, is often the preferred choice.

Hierarchy is a natural way of organizing semantics in natural languages and an important amount of work has been accomplished on defining semantic relationships between constituents in natural languages and referred to as taxonomies, thesauri, dictionaries etc. The relationships between concepts are intended to be combined to produce larger propositions that can then be used in a variety of interpretation paradigms and queries. Hence, semantic hierarchy is a natural way of querying complex data sets assuming

that some knowledge about the data (known as metadata) is organized in a hierarchical manner and carefully linked to the data set. However, defining a complete and correct hierarchy of metadata for a given application is extremely hard and time consuming.

In fact, in a taxonomy-based information organization, each element of a hierarchical meta-data represents a high-level *concept* that can be associated to data items that are relevant to it, facilitating the user in the retrieval of the available contents.

However, creating meta-data for the many and varied domains is a time-consuming process and meta-data construction is a major bottleneck to the wider deployment and use of semantic information. Since manual meta-data construction is costly, error-prone and inflexible to change, it is hoped that an automated (or semi-automated) process will result in a better meta-data construction and create knowledge structures that match a specific application. Considering these aspects, many resarchers tried to extract and organize relevant information from the data in order to automatically generate meta-data [165]. These meta-data learning approaches can be distinguished based on the type of input used for learning; they can learn from text, from a dictionary, from a knowledge base, from a semi-structured schema, or from a relational schema. Currently, few projects attempt to support the entire meta-data learning process, including automated support for tasks such as retrieving documents, classifying, filtering and extracting relevant information for the ontology enrichment; but most existing approaches for unsupervised meta-data construction require a large number of input documents for accurate results. Unfortunately, even if these methods can be very efficient in terms of computational costs, they can not guarantee high level quality.

For all these considerations, the importance of a well engineered meta-data structure is absolutely central in an IR application: in fact, with the enormous growth of the available data (for example, within the web), it is important to develop information discovery mechanisms based on intelligent techniques to make this creation process easier for any new specific domain application (or any user context).

Moreover, since meta-data have significant roles in data annotation, indexing and retrieval, they can also be succesfully used for data search and navigation purposes; in fact, a meta-data hierarchy is formed by a set of concepts (the number depends on how deeply the domain has been explored) and a set of edges (representing the relationships among concepts that exist in the considered domain) that, if used to explore data, can greatly help improve the efficiency of the exploration process. For example, in order to

classify musicians, based on their influences or style, it is possibile to choose a taxonomical meta-data that organizes the artists by music genres; but also, it is possible for a user to analyze the relationships that exist among the classified artists by navigating the meta-data and therefore discover new latent knowledge (i.e, influences among artists, similar styles, etc).

For all these considerations, hierarchical meta-data structures can represent highly focused alternatives for exploration processes into a large data set, supporting the standard query mechanisms where they fail in emphasizing users interests. In fact, considering also the enormous amount of data available through web-technologies and the very high number of users that can access them remotely, it is important to provide personalized strategies that can help the user in retrieval operations with very focused methodologies. Personalized search and navigation alleviates the burden of information overload by tailoring the information presented based on an individual user needs, and obviously, one of the key factors for accurate personalized information access is the user context. However, the representation of user preferences, search context, or the task context is generally missing in most search engines [80]. Indeed, contextual retrieval has been identified as a long-term challenge in information retrieval.
Considering these aspects, the meta-data hierarchies can represent possible usage contexts of the users, defining their foci of interest and leading the navigation into the data; in fact, by selecting a specific meta-data (or simply reporting preferences among the proposed concepts), users can clearly express some interests that can be used by the search and navigational system to retrieve data items relevant to the expressed preferences. Thus, a meta-data structure can also be used for contextualizing the navigation and retrieval process, providing to the user an intuitive mechanism to explore the data. Moreover, if the meta-data properly reflect the corpus knowledge and organizes its content based on the user's interests, the navigation experience can have a significant benefit. In this thesis we will evaluate all these aspects, providing novel algorithms to create a corpus-aware meta-data and we will use these automatically generated taxonomies for improving, from a user point of view, the data exploration process. In the next Section, we provide the details about the specific contributions of this work.

1.1 Contribution of the Thesis

In this thesis, given a data corpus and an associated hierarchical meta-data, we propose a set of innovative algorithms to analyze the given data and

adapt the meta-data structure to the semantics expressed by the data themselves. In fact, we believe that even if hierarchical meta-data structures are often carefully engineered by human domain experts, they do not necessarily reflect the content knowledge, especially in dynamically evolving domains. Moreover, when the user's interests are highly focused, available meta-data structures (which are often designed for broad coverage of concepts in a given application domain) may fail to reflect details within the users foci of interest. Thus, in this thesis, we start asking ourself the following question:

"*is there a feasible approach to efficiently and effectively adapt a given meta-data hierarchical structure to a usage context*?"

Therefore, we provide different mechanisms to adapt, re-size and remodel the given hierarchical meta-data structures based on statistical analysis on the original data, in order to reflect the most relevant information (properly restructured) in the hierarchies. This way, the resulting structures are adherent to the original data in such a way that they can lead more accurate exploration processes. Starting from different assumptions, we present two different hierarchical meta-data *adaptation* processes, which are based on different input data.

After discussing the state of the art in Chapter 2, in Chapter 3 we define the fundamental notations that we will use along all the thesis. In particular, we provide a set of methods to make explicit the knowledge that corpus and meta-data represent, and we introduce a novel meta-data formalization method that permits to highlight the correlations existing among the hierarchy nodes and the considered corpus [21].

In Chapter 4 we analyze text document collections, studying the existing structural relationships between a text corpus and its associated meta-data. In particular, we observe that, in a text environment, the primary role of a meta-data structure is to describe the natural relationships that exist between concepts (nodes in the meta-data hierarchy) in a given data corpus. Therefore, a corpus-aware adaptation of a hierarchical meta-data structure should essentially *distill* the structure of the existing taxonomy by appropriately segmenting and, if needed, summarizing this structures relative to the content of the corpus. Based on this key observation, we propose a novel adaptation method for re-structuring existing hierarchical meta-data structure to varying application contexts and we evaluate the proposed schemes using different text collections [24]. Metadata are pre-processed in order to eliminate irrelevant details and obtain a distilled version; pre-processing needs to be performed carefully to ensure that it does not cause significant

amounts of information loss. In other words, the hierarchy reduction process should eliminate the details in the meta-data that are not likely to be used by the users. Moreover, considering the different needs of each user, the reduction also has to let the user chose the detail level she is interested in, i.e., let her select how much data need to be preserved in the final structure.

Then, after adapting a given hierarchical meta-data structure to a usage context, in Chapter 5, we propose a novel Context-based Search and Navigation (CoSeNa) technique that leverages the relationships vehiculated by the adapted meta-data hierarchy to guide the user in a more effective exploration of the data [25, 26]. In particular we define a *keywords-by-concepts* (KbC) graph, which supports navigation using meta-data concepts as well as keywords characterizing the corpus of data. The KbC graph is a weighted graph, created by tightly integrating keywords extracted from documents and concepts obtained from domain meta-data structures. Documents in the corpus are associated to the nodes of the graph based on evidence supporting contextual relevance; thus, the KbC graph supports contextually informed access to these documents. The construction of the KbC graph relies on a spreading-activation like technique which mimics the way the brain links and constructs knowledge. This proposed system leverages the KbC model as the basis for document exploration and retrieval as well as contextually informed media integration. In fact, using CoSeNa, the user can navigate within the document space by leveraging navigational path that the system proposes. Moreover, the proposed system provides integration with three online popular media sources: Google Maps, Flickr, and YouTube.

The second part of the thesis is devoted to metadata adaptation to enhance the efficiency of the navigation in structured domains. Chapter 6 focuses on the cases where metadata are summarized and adapted to improve the navigation efficiency and effectiveness within dataspaces consisting of relational tables [20].

Considering data tables with millions of entries and dozens of different attributes, we optimize our approach in order to handle those large amount of data. Thus, we provide (as we did for the text corpora) optimized algorithms for extracting the most representative knowledge from the tuples and organize it in a meta-data structure, respecting as much as possible the original structured knowledge expressed by the given meta-data hierarchy. The proposed approach minimizes the information loss due to the reduction in details leveraging the redundancy in the data to identify value and tuple clustering strategies that can result in (almost) the same amount of information, but with a smaller number of data representatives. Therefore, this meta-data adaptation approach permits to reduce the size of the given

meta-data hierarchies (and re-format their internal structures) based on the distribution of the data in the considered tables.

Then, we apply the proposed adaptation method to hierarchical meta-data structures related to data tables that need to be reduced in their cardinality, in order to be easily explored in their largeness. In fact, exploration of large data tables is required in many scenarios where it is hard to display complete data sets, formed in many cases by millions of tuples and dozens of different attributes. Consider, for example, a scientist exploring a data base which archives and provides access to a large number of data collected by different researchers within the context of different projects. When this scientist poses a search query, without any knowledge about the specific data tables contents, her query might match many potentially relevant data tables. For this scientist to be able to explore the multitude of candidate data resources as quickly and effectively as possible, data reduction techniques are needed. Based on this key observation, the proposed exploration approach relies on the idea of summarization, and it takes as input a data table and (using the previously calculated adapted meta-data hierarchy) returns a reduced version of it, allowing the user to analyze only few entries that represent the general trends. The result provides tuples with less precision than the original, but still informative of the content of the database. This reduced form can then be presented to the user for exploration or be used as input for advanced data mining processes. In particular, with this method, each tuple in the original table is represented, in the summary, with a more generic tuple that *summarizes* its knowledge; on the other hand, each new tuple in the summary represents a maximal set of original tuples that can be expressed by it minimizing the information loss. In fact, our summarization approach leverages the underlying redundancy (such as approximate functional dependencies and other patterns) in the data to minimze this information loss.

All these methods are therefore evaluated by analyzing the performances of each proposed meta-data adaptation approach in order to quantify the benefits of using an adapted hierarchy (instead of the original one) and we compare our ideas to existing tree re-structuring methods. Moreover, we also analyze the computational costs of this pre-processing operations and we study their advantages and disadvantages against alternative adaptation methods.

Chapter 2

State of the Art

Considering the explosion of the accessible contents, especially on web communities, the problem of searching, indexing and navigating the available data is becoming very important. Moreover, with the advent of the new portable devices there is an emerging need for novel smart and flexible mechanisms to access and retrieve relevant information within the user's focus of interest.

Considering these aspects, we distinguish between two important objectives:

- information access and retrieval: we need to provide different retrieval mechanisms to the user, in order to have the useful information completely accessible in every condition and with every support;
- navigation into the document space: we need to provide novel mechanisms for an easy and smart exploration of the data, also highlighting hidden relationships among them and helping the user discover semantically relevant connections among them.

In the following Sections, we will analyze in details both aspects, providing a broad overview of the existing approaches in the research community.

2.1 Query Expansion and Re-Formulation

In the literature, the most popular retrieval approach is based on search queries. In fact, most information retrieval (IR) systems rely on a keyword search scheme, where queries are answered relying on the keyword contents of the text. Generally, the common way to search data information in huge

data sets is to provide search keywords to the system; but too often the user queries are under-specified. Users tend to provide at most two or three keywords but this is often insufficient to hone on the most relevant documents [149].

Considering the web space, where hyperlinks provide structural evidence to help identify authoritative sources, link analysis is used to help tackle this problem. But even then, query under-specification remains a significant challenge. In order to address this challenge, one of the most popular approaches is *query expansion* [94]; the goal of this approach is to modify the initial query by adding, removing and/or changing terms with similar ones.

Existing state-of-the-art query expansion approaches can mainly be classified into two classes: global and local analysis. The first one relies on corpus wide statistics such as the co-occurrences of every pairs of terms. To expand a query, the terms which are more similar to it are also considered. On the other hand, local analysis uses only some initially retrieved documents for further query expansion. An hybrid solution was introduced in [159], where both co-occurrences and local concepts selection are used for retrieval.

In [148], the authors present an example of global analysis approach: they propose a technique which adds terms obtained from term clusters built based on co-occurrences of terms in the document collection. A similar work was presented in [122]: in this work term similarities are used instead of term co-occurrences. These methods, however, cannot handle ambiguous terms: if a query term has different meanings, the clustering will add non-related terms thus making the expanded query ambiguous as well. In [122] the authors introduce a method for expanding target concepts of the whole query instead of a term-by-term change.

In [36] the authors exploit user logs for query expansion. Every search engine accumulates a large amount of query logs. The basic idea is that if a set of documents is usually selected for the same queries, then the terms in these documents are somehow related to the terms of the queries. Thus some correlations between query terms and document terms can be established based on such query logs.

Another common query reformulation method is *user's relevance feedback* [134]. The idea is to ask the users to mark relevant documents in search results and re-weighting the keywords of the initial query based on how effective they are according to such feedback. The obvious drawback of this technique is that it puts significant overhead on the users and assumes that the users know what they want and can provide consistent feedback.

Since this is rarely the case, the relevance feedback may by ineffective or may require significant amount of interactions. This mechanism has the big problem of charging the user with another, maybe hard, work on the data. One way to reduce the load on the user is to rely on *pseudo relevance* feedback [55], where the top ranked documents are assumed to be relevant and query enrichment is performed without user intervention, using these top-ranked documents. But this scheme works only if the initial query results are highly relevant and can degenerate if the first query results contain not-so-relevant documents.

Query reformulation can also be done by replacing items in a thesaurus with their longer descriptions. The thesaurus may be based on the used collection or based on a top domain knowledge like Wordnet [46]. However, these schemes still assume that user's initial query is highly precise and its expansion is sufficient to identify the relevant documents. Therefore, one needs to be very precise in formulating, or retrieving, the expansion of the keywords: in fact the system should guarantee the consistency of the re-formulated query by also analyzing the possibile ambiguities.

2.2 Information Retrieval by Exploring and Browsing

An alternative approach to retrieval is to rely on an exploratory process instead of data indexing and query matching [85]. As stated in [145] there exist three reasons for preferring this retrieval-by navigation approach over pure keyword-based text retrieval:

1. query formulation represents the most critical step in the whole retrieval process [131] because of a variety of factors such as user inexperience and lack of familiarity with terminology;

2. the scope of a user query, in many cases, is too broad to express precisely using a set of keywords;

3. the users prefer to navigate within a topic rather than being despatched to some system-relevant documents. Navigation process helps users understand the surrounding context to better hone on the relevant documents. This behavior is called orienteering [152].

Consequently, even if many existing retrieval systems continue to rely on the more traditional query-based IR model, there is a recent tendency towards relying on browsing in contrast to directed searching. In these

schemes, querying is nothing but an initial way of identifying starting points for navigation, and navigation is guided based on the context supplied in the query as well as any additional semantic metadata, such as taxonomies. These "semi-directed or semi-structured searching" processes [42, 145] help address the "don't-know-what-I-want" behavior [7] more effectively than relevance feedback schemes that assume that the user knows what she wants.

Considering this navigational approach, we distinguish two distinct important sub-problems:

1. finding starting points for exploration, and
2. providing guidance during navigation.

The problem of finding a useful starting point has been extensively studied: for example, Best Trails [156] selects starting points in response to queries, but it restricts them to be documents matching the query. The authors also propose an ad-hoc scoring function without considering navigability factors. Further, Best Trails' user interface departs substantially from the traditional query/browse interface and is difficult to use (as reported by the user study in [156]). The approach proposed in [145] closely adheres to familiar interfaces offered by popular query and browse tools. The hypertext retrieval paradigm known as topic distillation ([10], [27], [77]) aims to identify a small number of high-quality documents that are representative of a broad topic area. The basic idea in topic-distillation is to consider the structure of the data and propagate scores between documents in a way to organize topic spaces in terms of smaller sets of authoritative documents. Moreover, given a query, other methods propagate the term frequency values between neighboring documents [140] or the relevance score itself between documents connected with hyperlinks [144]. Much of the work to date on topic distillation uses as a primitive the HITS algorithm [77], which identifies sub-graphs consisting of hubs and authorities. Under the HITS model, good hubs are those that have many links to good authorities. Symmetrically, good authorities are those that are linked to by many good hubs. HITS-based approaches are inherently effective only for broad topic areas for which there are many hubs and authorities. The work of [75] aims to identify nodes of maximum influence in a social network, where the definition of influential nodes is related to the notion of *good* starting points.

But even considering good starting points (from a user perspective), as previoysly reported, it is necessary to provide to the user a guidance during her navigation process. For example, [72] highlights hyperlinks along paths taken by previous users who had posed similar queries. This approach

may not be suitable for open-ended search tasks where the query is not a good representation of the underlying search task. Other approaches ([86], [103]) do not rely on queries but instead passively observe the users browsing behavior in order to learn a model of her search task, and highlight links that match the inferred task.

Highlighting of hyperlinks based on an explicit query, as a method of navigation guidance, was evaluated, with a user study, in [109]. Guided navigation was found to result in significantly faster completion of certain search tasks compared to traditional query and browsing interfaces, assuming the user already knows of a suitable starting point.

2.3 Data Summarization: Reducing the Cardinality to Help the User Explore the Data

While the existing exploration systems help the user navigate within the data sets, most of them fail to orient her within it, because of the very large number of single data items; in fact, even the best search/navigation system can not efficiently handle millions of single data items and generally fails to effectively report the results of a user query (or they fail to efficiently organize them). Indeed, most of the users are disoriented by very large list of resulting documents (returned by standard keyword-based search query), or by too large data navigational path (in navigation-based IR system) and get confused by the cardinality of the presented results. Moreover, it has been proved that, in case of very large result lists, the user only takes into account the first ones, completely ignoring the majority of the reported data items.

From a user's point of view, the smart and flexible query methods can help provide a fast and direct access to the data but do not help the user in the understanding of the returned results; as deeply studied in the literature, if the visualization of the data results hard and complex, even the positive effect of efficient retrieval methods vanishes.

For all these considerations, many IR systems also rely on *cardinality reduction* processes, that minimize the information loss due to the reduction in details. In order to do this, many data reduction/summarization algorithms have been proposed. In fact, a summarization algorithm usually leverages the redundancy in the data to identify value and clustering strategies that represent the (almost) same amount of information with a smaller number of data representatives.

In the literature the summarization problem has been extensively stud-

ied; in the following Sections we will analyze the proposed approaches, distinguishing among them based on the format of the considered data.

2.3.1 Text Data Summarization

Considering the text data format, the summarization process is generally based on the study of *narrative topic development*; in particular, summarization of a text stream relies on the analysis of the evolution of the topics expressed by the sequence of sentences. Given a text document, the task of text summarization is to condense the information (minimizing the information loss) in the input document in a more concise output.

One of the earliest summarization approaches by Luhn [88] uses word frequency counts in the text to detect important words and assigns significance scores to sentences based on the occurrence of significant words in the sentence and their position in the sentence. In [161], the authors use word frequency and word position in the document, as against sentence position to score sentences. Sentences are then selected so as to maximize the total sentence score in the summary. In TXTRACT [11], Boguraev and Neff utilize text discourse segmentation to aid a summarizer based on salience with a background document collection.

More recently, other approaches have investigated the discourse structure [95], the combination of information extraction and language generation [132], and machine learning techniques to find patterns in the given text [153].

In general, we can distinguish two different summarization problems: single document and multi-document summarization problems. In single document summarization, summary sentences are typically arranged in the same order as in the full document, although [70] reports that human summarizers do sometimes change the original order. In multi-document summarization, the summary consists of fragments of text or sentences that were selected from different documents. Thus, there is no complete ordering of summary sentences that can be found in the original documents. In domain dependent summarization, it is possible to establish possible orderings *a priori*. A priori defined simple ordering strategies are combined together by looking at a set of features from the input. [41] uses such techniques to produce patient specific summaries of technical medical articles. In [6], the authors proposed a strategy for ordering information that combines constraints from chronological order of events and topical relatedness. Another approach [104] is bottom-up reordering and it is used to group together stretches of text in a long, generated document by finding propositions that

are related by a common focus.

On a different dimension, there are two types of summarization: extractive and abstractive summarization. Extractive summarization usually ranks the sentences in the documents according to their scores calculated by a set of predefined features, such as term frequency-inverse sentence frequency [124], term/sentence position [160], [161], and number of keywords [161]. [54] proposes a method using latent semantic analysis to select highly ranked sentences. In [53], the authors describe a maximal marginal relevance method to summarize documents based on the cosine similarity between a query and a sentence. [155] is based on sentence-level semantic analysis (sentence-sentence similarities) and symmetric non-negative matrix factorization. Other methods include CRF-based summarization [146] and hidden Markov model based methods [35]. Instead, abstractive summarization involves information fusion, sentence compression and reformulation ([78], [71]).

Many other approaches also take into account a pre-process ordering method; the basic idea is to find a reasonable order among all the data entities of the considered data set and then, apply the reduction algorithm. Therefore, an ordering mechanism can be defined as a method that, given a set of data documents, should be able to provide an order of data portions that reflects, as much as possible, the knowledge structure represented in the whole set of documents.

Considering the text data field, the ordering task can be easily seen as a sentence ordering problem: this problem has been extensively investigated in the literature [96, 105, 63, 104].

2.3.2 Data Table Summarization

The summarization problem has been also extensively studied within the database field. Many systems in fact rely on database for the data organization and, due to the huge amount of stored data, it is not reasonable to provide a complete view of all of them. For example, [127, 128] present a table summarization system called *SaintEtiQ*, which computes and incrementally maintains a hierarchically arranged set of summaries of the input table. This way the system can select the most suitable summary to present to the user (depending on her need) providing a reduced view of the data. *SaintEtiQ* uses background knwoledge (i.e. metadata) to support these summaries. [3] also performs data summarization, but it relies on frequent patterns in the relational dataset.

TabSum [87] creates and maintains table summaries through row and

column reductions. To reduce the number of rows, the algorithm first partitions the original table into groups based on one or more attribute values of the table, and then collapses each group of rows into a single row relying on the available metadata, such as the concept hierarchy. For column reduction, it simplifies the value representation and/or merges multiple columns. [157] discusses a related approach for refinement of table summaries. Neither of these approaches, however, considers the impact of the imprecision of metadata during summarization.

Data compression techniques, like Huffman [64] or Lempel-Ziv [166], can also be used to reduce the size of the table. For example, [107] presents a database compression technique based on vector quantization. Buchsbaum et al. [15, 14] develop algorithms to compress massive tables through a partition-training paradigm. These methods, in general, are not directly applicable to our problem domain since the compressed tables are not human readable.

Histograms can also be exploited to summarize information into a compressed structure. Following this idea, Buccafurri et al. [13] introduce a quad-tree based partition schema for summarizing two-dimensional data. Range queries can be estimated over the quad-tree since the summarization is a lossy compression. Leveraging the quad-tree structure, [37] proposes approaches to processing OLAP operations over the summary. [37] first generates a 2-dimensional OLAP view from the input multidimensional data and then compresses the 2-dimensional OLAP view by means of an extended quad-tree structure. In fact, any multidimensional clustering algorithm can be used to summarize a table. Such methods, however, do not take into account specific domain knowledge (e.g. "what are acceptable summarizations, how do they rank?") that hierarchies would provide.

OLAP operations, *drill-down* and *roll-up*, which help users navigate between more general and more specific views of the data with the help of given value hierarchies, are also related to table summarization. The concept of imprecision in OLAP dimensions is discussed in [113]. In that framework, a fact (e.g., a tuple in the table) with imprecise data is associated with dimension values of coarser granularities, resulting in the dimensional imprecision. In schema design for traditional relational-OLAP systems, issues about heterogeneous dimensions have been discussed in [66, 65]. A dimension is called heterogeneous if two members in a given category are allowed to have ancestors in different concepts. Given a heterogeneous dimension, an aggregate view for a category may not be correctly derived from views for its sub-categories; in fact, the summarizability denotes the conditions under which a value or object can be summarized correctly from a more detailed

value or object, based on the given summarization rules (e.g., value mappings and value lattices). In [121], the authors supported OLAP operations over imperfectly integrated taxonomies.
The table summarization task is also related to the k-anonymization problem, introduced as a technique against linkage attacks on private data [137, 8, 83]. The k-anonymization approach eliminates the possibility of such attacks by ensuring that, in the disseminated table, each value combination of attributes is matched to at least k others. To achieve this, k-anonymization techniques rely on a-priori knowledge about acceptable value generalizations. Cell generalization schemes [2] treat each cell in the data table independently. Thus, different cells for the same attribute (even if they have the same values) may be generalized in a different way. This provides significant flexibility in anonymization, but the problem is extremely hard (NP-hard [97]) and only approximation algorithms are applicable under realistic usage scenarios [2]. Attribute generalization schemes [137, 8, 83] treat all values for a given attribute collectively; i.e., all values are generalized using the same unique domain generalization strategy. While the problem remains NP-hard (in the number of attributes), this approach saves significant amount of time in processing and may eliminate the need for using approximation solutions, since it does not need to consider the individual values. Most of these schemes, such as Samarati's original algorithm [137], however, rely on the fact that, for a given attribute, applicable generalizations are in total order and that all the generalization steps in this total order have the same cost. [137] leverages this to develop a binary search scheme to achieve savings in time. [83] relies on the same observation to develop an algorithm which achieves attribute-based k-anonymization one attribute at a time, while pruning unproductive generalization strategies. [8] assumes an attribute order and attribute-value order to develop a top-down framework with significant pruning opportunities. In [19] the authors have formulated the problem of table summarization with the help of domain knowledge lattices providing the outline of a fuzzy mechanism to express alternative clustering strategies.

2.4 Information Organization Supported by Hierarchical Meta-Data

An important help in searching and navigating into huge data sets could be also given by hierarchical meta-data structures; in fact, a meta-data based categorization is a crucial and well-proven instrument for organizing large

volumes of data information. Automatically categorizing documents into pre-defined topic hierarchies (or taxonomies) is a crucial step in knowledge and content management; indeed taxonomies embody formalized knowledge and define aggregations between concepts/categories in a given domain that could facilitate the organization of the data and make the contents easily accessible to the users.

A concept meta-data hierarchy is considered an effective representation that describes relevant categories related to a particular domain. In a web environment for example, a concept taxonomy can be also used to flexibly describe and index, with varying granularity, various web contents.

Currently there are many concept hierarchies available in the web; in fact they also enable sharing and integration of information from different domains and data sources. However, given a data set, it is not easy to find the appropriate categorization that best describes and indexes the contents. The available meta-data hierarchies are usually designed for broad coverage of concepts in a considered domain, failing to properly reflect important details within the considered data collection. Indeed, especially in dynamically evolving domains, the available structures could not necessarily reflect the content knowledge. For all these considerations, the research community has investigated the problem of automatically creating/distilling meta-data hierarchies that best reflect the considered data information. To cope with this, many different approaches have been studied, taking into account different constraints and needs. In the following Sections, we will provide a broad overview about the existing algorithms.

2.4.1 Unsupervised or Semi-Supervised Meta-data Distillation Approaches

Many authors tried to automatically extract hierarchical categorizations from the considered data that have to be indexed. [16] presents an overview about the many methodologies that have been proposed to automatically extract structured information from a considered data set (reporting also procedures and metrics for quantitative evaluations). In [139] the authors present an unsupervised method to automatically derive from a set of documents a hierarchical organization of concepts (salient words and phrases extracted from the documents), using co-occurrence information.

One of the most critical tasks in unsupervised (or semi-supervised) categorizations is the definition of the semantical relationships among the retrieved concepts: [32] organizes the extracted concepts by analyzing the syntactic dependencies of the terms in the considered text corpus. [33] also

considers multiple and heterogeneous sources of evidence to improve the hierarchical relations between the selected terms. Many methods rely on preliminary supervised operations to limit the noise in the retrieved concepts: in [12], the user sketches a preliminary ontology for a domain by selecting the vocabulary associated to the desired elements in the ontology (this phase is called lexicalisation).

In the last few years, with the increase of semi-structured information repositories, many authors tried to leverage the information vehiculated by these sources to reduce the imprecision in the retrieved hierarchies: for example, in [115, 116], the authors investigate the problem of automatic knowledge acquisition from Wikipedia repositories. Moreover, [141] leverages the tag vocabulary extracted by Flickr to induce an ontology by using a subsumption-based model.

In [93] the authors present a meta-data hierarchy learning framework that extends typical meta-data engineering environments by using semiautomatic ontology construction tools.

As already reported, hierarchical categorizations, when available, can play a significant role in the organization and summarization of the data. [82, 81] generate hierarchies in order to summarize the documents retrieved by a search engine, while [79] proposes a hierarchical clustering algorithm to build a topic hierarchy for a collection of documents retrieved in response to a query. In a text environment, a concept meta-data hierarchy can be also used to flexibly describe a user/group's interests with varying granularity. However, the stagnant nature of the developed structure may fail to timely capture the dynamic change of the user's interest and the complex nature of the evolving contents. [151] addresses the problem of how to adapt a topic meta-data structure in order to reflect the change of a group's interest to achieve dynamic group profiling.

Moreover, researchers have attempted to construct meta-data hierarchies by examining the data domains. This is useful because implementers can quickly identify various techniques that can be applied to their domain of interest. [23] constructs a data-oriented taxonomy, visualizing several subcategories. Previously, [31] proposed a taxonomy of information visualization techniques based not only on data types, but also on the processing operators that are inherent in each visualization technique.

2.4.2 Meta-data Hierarchies Matching: Studying and Analyzing the Taxonomical Structures

Meta-data structures from different sources (and referring to the same domains) are rarely identical; in fact, their knowledge structure strictly depends on the domain expert that defined them and, in many cases, there is need for techniques to find alignments between concepts in different structures.

The problem of matching context-describing meta-data hierarchies has been investigated in various application areas, especially in scientific, business, and web data integration [126, 44, 100, 98, 39, 102, 92, 111, 22]. Different matching techniques focus on different dimensions of the problem, including whether data instances are used for schema matching, whether linguistic information and other auxiliary information are available, and whether the match is performed for individual elements (such as attributes) or for complex structures [126].

Cupid [91] is a schema-based approach that implements a sequential composition of different matchers. It consists of a first phase based on a linguistic matcher and a second phase based on a structural matching technique. The linguistic matcher calculates similarity coefficients between schema label nodes, while the structural matcher computes similarity values which measure the similarity between contexts in which elementary schema elements occur. A final phase aggregates these results by means of a weighted sum and compares them with a given threshold in order to generate the alignment. This algorithm operates only with trees: other schemas can be handled through a translation process. [100] uses schema graphs for matching; matching is performed node by node starting at the top; thus this approach presumes a high degree of similarity (i.e., low structural difference) between the taxonomies. Onion [102], the successor of SKAT [101], is a schema-based system that leverages logic rules to discover match and mismatch relationships between multiple ontologies, represented internally as labeled graphs. The matching algorithm proposes a sequential (and semi-automatic) approach that first performs a linguistic match and then applies a structure-based matching. The latter is based on the result of the first step and tries to match only the unmatched terms; it is based on a structural isomorphism detection technique between the subgraphs of the ontologies.

[22] and DIKE [112] use the distance of the nodes in the schemas to compute the mappings; while computing the similarity of a given pair of objects, other objects that are closely related to both count more heavily than those that are reachable only via long paths of relationships. Glue [40],

the successor of LSD [38], is an instance-based semi-automatic system that uses machine-learning techniques to discover one-to-one mappings between two taxonomies. It is based on the calculus of the joint distributions that are used for any similarity measures. This approach can be divided in three steps. First, a multi-strategy learning approach allows to compute the joint distribution of classes that are used in the second step to produce a similarity matrix. The latter is used in the final phase by a relaxation labeling technique in order to filter only the best matches contained in the similarity matrix. Differently from Glue, FCA-merge [150] takes as input two ontologies that share the same set of instances and produces a new ontology as result. It uses formal concept analysis techniques, through a process made up of three steps: instance extraction, concept lattice computation, and (interactive) generation of the final new ontology. Clio [98, 99] is a mixed schema-based and instance-based system that proposes a declarative approach to schema mapping between either XML and relational schemas. After the first phase in which input schemas are translated into an internal representation, the system combines sequentially an instance-based attribute classification (by using a Bayes classifier) with a string matching between elements name. These n-to-m value correspondences can be also entered by the user through a graphical user interface. After that, Clio produces a set of logical mappings with formal semantics, supporting also mappings composition. [44] provides a more detailed classification of matching techniques, based on other features including different similarity measures, matching strategies (such as name similarity or class similarity), and degrees of user involvement. [61] proposes an algorithm for ontology matching that combines standard string distance metrics with a structural similarity measure based on a vector representation. Despite such advances in structural mapping technologies, alignments across data sources are rarely perfect. In particular, imperfection can be due to homonyms (i.e., nodes with identical concept-names, but possibly different semantics, in the given taxonomy hierarchies) and synonyms (concepts with different names but same semantics). While structural-matching techniques help finding node-to-node alignments, they fall short when such scenarios arise.

2.4.3 Meta-Data Adaptation/Summarization: Re-Structuring the Hierarchical Knowledge for Reduction Purposes

The advent of the Web and the enormous growth of digital content in intranets, databases, and archives, have further increased the demand for meta-data categorization hierarchies. Obviously, manual categorization of-

ten lacks economic efficiency and automatic tools are indispensable to supplement human efforts. Thus, there is a need for novel instruments for supporting the creation of properly adapted (and in many cases also reduced) meta-data structures; in order to adapt/summarize hierarchical structures to represent (and eventually index) the available contents, various hierarchical clustering methods have been proposed.

Again, the basic idea is that, considering the novel visualization devices and the different user needs, it could be necessary to reduce the cardinality of the selected meta-data hierarchy. Obviously this reduction process has to leverage the underlying redundancy while preserving as much as possible the data information within the user's foci of interest.

To cope with this, many meta-data reduction algorithms have been proposed; most of them use a hierarchical clustering approach. There are two major approaches for hierarchical clustering: agglomerative clustering and divisive clustering. In [1], the centroids of each class are used as the initial seeds and then a projected clustering method is applied to build the hierarchy. During the process, the clusters with too few documents are discarded. Thus, the taxonomy generated by this method might have different categories than those predefined. In [84] a linear discriminant projection is applied to the data first and then the hierarchical clustering method UPGMA [69] is exploited to generate a binary tree. [118] applies a divisive hierarchical clustering; the authors generate a taxonomy with each node associated with a list of the categories. Each leaf node has only one category. This algorithm basically uses two centroids of the categories which are furthest away from each other as the initial seeds and then it applies spherical k-Means to divide the clusters into two sub clusters. [62] associates word distribution conditioned on classes to each node: the method uses a variance of the EM algorithm to cluster nodes. Similarly, [142] presents a method in which concepts are probabilistically modelled. The probabilistic classes are organized in hierarchies by relying on the KL divergence measure between the probability distributions associated to the concepts.

Considering the current state of the art, there are a lot of possible uses for summarized categorization structures. In fact, summarization has been used to support various reasoning tasks. Fokouel *et al.* [48] focus on the problem of summarizing OWL structures while *KAON2* [67] reduces an ontology to a disjunctive datalog program and makes it naturally applicable to reasoning with Aboxes stored in deductive databases. Another interesting *RDF*-based approach was proposed by Zhang et al. [163]. An RDF Sentence Graph is proposed to characterize the links between RDF sentences derived from a given ontology. The salience of each RDF sentence is assessed in

terms of its *centrality* in the graph. Zhang et al. propose to summarize an ontology by extracting a set of salient RDF sentences according to a re-ranking strategy. Since many metadata types, such as value hierarchies and taxonomies, are hierarchical, researchers also experimented with tree summarization algorithms [52, 125]. For example, Davood *et al.* [125] observed that summaries of XML trees did a much better job in document clustering tasks than methods using edit distance values, e.g. [108], on the original trees.

DataGuides [52] was one of the first approaches which attempted to construct structural summaries of hierarchical structures to support efficient query processing. Though this and similar methods work fine for tree based structures, in the sense that the number of nodes in the summary are less than in the original tree and that they capture the structure well, the constructed summaries are not trees, but graphs. Various other summarization algorithms, such as [125, 60, 30], focus on creating summaries suitable for efficient similarity-search in tree-structured data. Since, once again, the goal of these algorithms is not to obtain a smaller tree representing the larger one provided as input, but to find a representation that will speed up query processing, the resulting summaries are in the form of strings, hash sequences, and concept/label vectors. Our goal, in this thesis, however is to reduce the size of the input taxonomy tree to support table summarization process, therefore these and similar algorithms are not applicable.

2.4.4 Evaluation of Meta-Data Structures

Evaluation of the quality of automatically generated (or adapted) meta-data hierarchies is a very important and non-trivial task. In the literature, many evaluation measures have been introduced. In [162], the authors determine the precision of the clustering algorithm by manually assigning a relevance judgment to the documents associated to the clusters. In [164], the authors use the F-Score to evaluate the accuracy of the document associations (but the approach requires a ground truth, which is hard to determine in many cases). In [138] the authors perform a user study to evaluate the qualities of the relationships between concepts and their children and parent concepts. In [82], authors estimate the goodness of concepts when compared to the top TF-IDF terms and measure the quality of the concepts by evaluating their ability to find documents within the hierarchy (the "reach time" criterion measures the time taken to find a relevant document).

Chapter 3

Definition of the Hierarchical Meta-Data Knowledge

In this chapter, we define the fundamental knowledge and the notations that we will use along all the other chapters of the thesis. Given as input data a hierarchical meta-data (defining the considered domain) and a related data corpus, as originally proposed in [21] and [25], we apply standard knowledge extraction techniques and innovative statistical analysis methods in order to make explicit the knowledge they both represent and highlight the correlations existing among them. In particular, as explained in [21], we introduce novel mechanisms to formalize the meta-data hierarchy knowledge that not only leverage the structural information but also enrich them by analyzing the considered corpus contents.

In order to do that, we introduce three main operations that formalize the knowledge expressed by the input data:

- *data vectorization*: given an input taxonomy and a related data corpus, we formalize the knowledge expressed by them in a vector space, making this knowledge accessible, usable and comparable by other statistical analysis processes.

- *data association process*: this process assigns each given data element (from the given corpus) to one or more concept-categories (and vice-versa) based on the semantical similarity; we propose different approaches that take into account different requirements.

- *concept-keyword relationship*: in order to improve the quality of the association process (that can properly define each considered concept),

we introduce a novel relevance feedback-based method that highlitghs new hidden relationships between relevant terms (contained in the given data corpus) and the original concept/categories of the given taxonomy, making the latent semantic connections among them explicit.

In the next Sections we will analyze in details each of these operations. Finally, in Section 3.4, we will study the efficacy of our meta-data knowledge formalization approach by comparing the proposed approach against alternative representation methods.

3.1 Data vectorization

In order to analyze the knowledge expressed by the input taxonomy and compare such information against the data corpus, we need to formalize the relevant knowledge they express in a shared common space.

3.1.1 Structural Vectorization of a Meta-Data hierarchy

Taxonomies (also referred to as hierarchies in this thesis) have played a central role in the development and deployment of many applications and have significant roles in the data annotation, search and navigation. They are generally defined (or extracted) by human domain experts and they represent a general knowledge about a specific domain organized in such a way to be easily understandable by human users. They embody formalized knowledge and define aggregations between concepts/categories (expressed by nodes in the hierarchical structure) in a given domain and could facilitate the organization of the data making the contents easily accessible to the users (by using the structure to index the available contents). However, in order to express the knowledge defined by the hierarchy itself, we need to formalize it by explicitating the information defined by the structure in a new vector space. Our basic assumption is that a meta-data hierarchy $H(C, E)$ is a tree structure, composed by $n = |C|$ concept-nodes, which satisfies two basic properties:

- a more general concept-node in the hierarchy subsumes its children concepts.
- The concepts subsumed by a concept-node are usually non-overlapping.

Given this type of structure, without loss of generality, for the structural vectorization process, we rely on the `CP/CV` mapping algorithm proposed in [76].

Definition 3.1.1 (Concept-Vector $\vec{cv}$) *Given a taxonomy $H(C, E)$, we map each concept-node as a concept-vector $\vec{cv}$ with n dimensions, in such a way to encode the structural relationship (defined by the edges in E) between this node and all the other nodes in the hierarchy. The concept-vectors are obtained by propagating the weight of each concept on the taxonomy tree according to their* semantic *contributions to the definition of the other concepts (dictated by the edges of the taxonomy).*

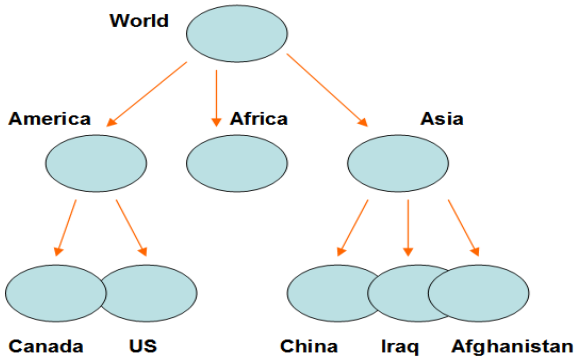


Figure 3.1: Example of a geographical taxomomy fragment.

The purpose of propagation is to identify the weighted concept-vectors that can represent the given nodes. Thus, for a taxonomy H with n nodes, `CP/CV` calculates n vectors that represent the hierarchy nodes, as follows.

The vectors dimensions are positionally associated to the nodes of the given taxonomy, according to any traversal order of the taxonomy. Before the propagation process starts, each concept-vector is simply initialized by setting to 1 the weight corresponding to itself, and 0 all the other elements; i.e., considering the node c_i in the given hierarchy, the initial concept-vector of this node is

$$\vec{cv_{c_i}} = [0, 0, ..., 1, ..., 0]$$

where the only non-zero weight is associated with the i-th dimension, the one associated to the node c_i. The total number of dimensions is equal

to the number of the nodes in $H(C, E)$. Then, the process repeatedly enriches the concept-vectors of the nodes by enabling neighboring nodes to exchange concept weights. The propagation of the weights works by adding to each concept-vector the weights of the neighbour ones (parent and children), multiplied by a *propagation degree*[1] that sets how much information has to migrate from one node to the neighbours. This process is iterated until all nodes are informed of all the others.

Consider, for example, the taxonomy fragment (containing nine concept nodes) presented in Figure 3.1. `CP/CV` maps each concept into a 9-dimensional vector (Figure 3.1). Vectors' elements are associated to the taxonomy nodes. For example, the root is represented by the vector

$$\langle 0.450, 0.169, 0.141, 0.158, 0.018, 0.018, 0.018, 0.021, 0.021 \rangle,$$

in which the first component (the one associated to the category "*world*"), dominates over the others that contribute to the definition of the concepts. The second, third and fourth components reflect the weights of "*asia*", "*africa*" and "*america*" respectively in the semantic characterization of "*world*", while the remaining components represent the weights of the three descendants of "*asia*" and of the two descendants of "*america*".

	World	Asia	Africa	America	Afgh.	Iraq	China	Canada	US
$\vec{cv}_{world}$	0.450	0.169	0.141	0.158	0.018	0.018	0.018	0.021	0.021
$\vec{cv}_{asia}$	0.052	0.469	0.006	0.006	0.156	0.156	0.156	0.0003	0.0003
$\vec{cv}_{africa}$	0.100	0.012	0.873	0.012	0.0006	0.0006	0.0006	0.0007	0.0007
$\vec{cv}_{america}$	0.057	0.007	0.007	0.520	0.0003	0.0003	0.0003	0.204	0.204
$\vec{cv}_{afgh.}$	0.004	0.100	0.0002	0.0002	0.872	0.012	0.012	0	0
$\vec{cv}_{iraq}$	0.004	0.100	0.0002	0.0002	0.012	0.872	0.012	0	0
$\vec{cv}_{china}$	0.004	0.100	0.0002	0.0002	0.012	0.012	0.872	0	0
$\vec{cv}_{canada}$	0.006	0.0003	0.0003	0.165	0	0	0	0.806	0.023
$\vec{cv}_{us}$	0.006	0.0003	0.0003	0.165	0	0	0	0.023	0.806

Table 3.1: Concept-vectors associated to the taxonomy fragment in Figure 3.1.

Once the process is completed, since all the concepts are mapped into the same vector space, the knowledge expressed by each node is comparable with all the others; i.e, it is possible to compute semantic similarities of the concepts by a similarity measure, for example the cosine similarity measure. In fact, [76] showed that cosine similarity (measuring the angles between the vectors)

[1]Note that, in [76], in the absence of any prior or external/corpus-based knowledge, the authors set the propagation degree based on the pure hierarchy structure (for example, a concept c with n children will have a propagation degree wrt. its children equal to $1/n$)

$$sim_{cosine}(\vec{v_1}, \vec{v_2}) = cos(\vec{v_1}, \vec{v_2}),$$

leads to highly precise similarity measurement across concepts within the taxonomy in comparison with other approaches. For example, average KL divergence (which treats the vectors as probability distributions and measures the so-called relative entropy between them),

$$\frac{\Delta_{KL}(\vec{v_1}, \vec{v_2}) + \Delta_{KL}(\vec{v_2}, \vec{v_1})}{2} = \frac{1}{2}\sum_{i=1}^{n} v_{2_i} log\frac{v_{2_i}}{v_{1_i}} + v_{1_i} log\frac{v_{1_i}}{v_{2_i}},$$

and intersection similarity (which considers to what degree $\vec{v_1}$ and $\vec{v_2}$ overlaps along each dimension)

$$sim_{intersection}(\vec{v_1}, \vec{v_2}) = \frac{\sum_{i=1}^{n} min(v_{1_i}, v_{2_i})}{\sum_{i=1}^{n} max(v_{1_i}, v_{2_i})}$$

will essentially normalize each dimension, thus they will likely give equal weight to all concepts, independently of where they occur. Instead cosine similarity will give more weight to the dimensions with higher values. [30], on the other hand, showed that KL Divergence performs better than the cosine and the intersection similarity in similarity search of hierarchical data, such as XML. Comparisons against other approaches on available human-generated benchmark data [49, 129] showed that `CP/CV` improves concept similarity measurements in terms of the correlation of the resulting concept similarity judgments to human common sense. Thus, without loss of generality, we use `CP/CV` to measure the semantic similarities between the nodes in the hierarchy.

3.1.2 Extraction of Document-Vectors

In this step, given a data corpus D of documents (also referred to as contents), we analyze and extract a representative document-vector for each of them.

The $m = |D|$ documents are represented with a **document-vector** in which each component represents a keyword. As usual, the keyword extraction includes a preliminary phase of stop-word elimination and stemming. For the stemming process, we use Wordnet [46].

Therefore, we calculate the weight $u_{j,x}$ of the $x - th$ vocabulary term in $j - th$ document by using the augmented normalized term frequency [136]:

$$u_{i,x} = 0.5 + 0.5 \cdot \frac{tf_{i,x}}{tf_i^{max}}$$

where $tf_{j,x}$ is the term frequency value of the $x-th$ vocabulary term in $i-th$ document and tf_i^{max} is the highest term frequency value of the $i-th$ document. In fact, as in the standard `TF` formula, we try to give credit to any term that appears in the corpus, but we only add some additional credit to terms that appear more frequently. In this way we preserve the keywords that appear less frequently. In fact, too often the most relevant terms related to a topic (or category, in our case) are specific keywords that do not appear so frequently in a big corpus of documents: the augmented normalized term frequency permits us to preserve this information.

Thus, given a corpus document d_i, we define the related document-vector as

Definition 3.1.2 (Document-Vector $\vec{d}$)

$$\vec{d_i} = \{u_{i,1}, u_{i,2}, ..., u_{i,v}\}$$

where v is the size of the considered vocabulary, and $u_{i,x}$ is the normalized term frequency of the $x-th$ vocabulary term in the $i-th$ document.

3.2 Data Association Process

The task of the document association process is to assign (or classify), if it is possible and supported by the contents, each given data element to one or more categories (concepts in our case) based on its contents. Document association problems can be divided into two major categories:

- *unsupervised document association*, where the classification must be done without any reference to external information;
- *supervised document association*, where some external mechanism (such as human feedback) provides information to improve the documents' classification process.

Our aim is to associate each document in D to at least one of the concepts/categories of the given taxonomy. In fact, as introduced above, a taxonomical structure could be seen as a domain categorization structure, that could also index the related documents and help the user navigate and search within the contents space.

In this thesis, the association process is based on the two structures previously defined:

- *concept-vectors*: they describe each concept of the taxonomy by quantifying the relationship among the nodes in the original structure;

- *document-vectors*: they define each document as a set of weighted keywords.

The main idea behind the association operation is that, while concept-vectors help us capture and leverage the structural information embedded in taxonomies, they can also be used to improve the association process. In fact, each concept-vector defines the related concept leveraging the taxonomical structure, and that helps us express the meaning of the category based on the relationships with all the other concepts in the taxonomy.

Let us consider for example the meta-data hierarchy fragment in Figure 3.1: the node "*asia*" could be naively defined by only considering its label or, in a more accurate way, we can leverage the relationships expressed by the structure. We can for example infer that the concept "*asia*" is related to (and also defined by) its ancenstors and descentants. Thus, using the concept-vectors, we also take into account this information; we call the set of documents that a concept relates to as its *association*. Therefore, the concept-vectors assigned to the concept nodes provide a convenient way to identify associations. In particular, we rely on a association module which takes as input the set

$$CV = \{c\vec{v}_1, \ldots, c\vec{v}_n\}$$

of the concept-vectors representing the taxonomy, and the set

$$DV = \{d\vec{v}_1, \ldots, d\vec{v}_m\}$$

of vectors representing the documents to be associated. Document-vectors are defined in the space of the entire vocabulary; each dimension corresponds to a keyword, and the weights in the vector represent the relevance of the corresponding keyword in the document represented by the vector.

Thus, the goal of the data association process is to *associate the documents to their best representative concepts in the taxonomy*. We capture this notion of representativeness through the similarity among the concept- and document-vectors representing the taxonomy concepts and the documents, respectively. Semantic similarities (at the basis of the association process) between the concepts and the documents being associated are computed by

- unifying the vector spaces of the concept-vectors and the document-vectors. The unification of the spaces consists in unioning the dimensions and representing every vector in the new extended space by setting to 0 the values corresponding to those dimensions that were not appearing in its original vector space (while keeping all the other components unchanged);

- calculating the cosine similarity of the resulting vectors in the unified space.

In the following discussions we will always assume to deal with vectors sharing the same space. In the next two Sections we analyze two different association approaches.

Document-to-Concept Association

For every document in the corpus, the document-to-concept association identifies the taxonomy concepts that best match with it. In other words, each concept in the original taxonomy is considered as belonging to the associations of those documents whose similarities with it are above an adaptively computed threshold. The association steps are the following:

For each document $d_j \in D$:

1. consider the document-vector $\vec{dv_j}$

2. compute its cosine similarity wrt. all the concept-vectors describing the given taxonomy, i.e., the concept-vectors associated to each node in the taxonomy.

$$sim(\vec{cv_i}, \vec{dv_j}) = cos(\vec{cv_i}, \vec{dv_j})$$

3. sort the concepts-vectors in decreasing order of similarity wrt. $\vec{dv_j}$;

4. choose the cut-off point to identify the concepts which can be considered *sufficiently similar* to justify the association of the object under them.

   Our method adaptively computes the cut-off as follows: It

   Algorithm 1 (Adaptive Cut-Off)

- ***1:*** *first ranks the concepts in descending order of match to* $\vec{dv_j}$*, as previously calculated;*
- ***2:*** *computes the* maximum drop *in match and identifies the corresponding drop point and,*
- ***3:*** *computes the* average drop *(between consecutive entities) for all those nodes that are ranked before the identified maximum drop point.*
- ***4:*** *The first drop which is higher than the computed average drop is called the* critical drop*. We return concepts ranked better than the point of critical drop as candidate matches.*

At the end of this phase, a document $d \in D$ has an **association**, $A_{d \to c}(\vec{cv})$ defined as

Definition 3.2.1 (Association $A_{d \to c}(\vec{cv})$)

$$\forall c \in C, c \in A_{d \to c}(\vec{cv}) \iff sim(\vec{dv_d}, \vec{cv_c}) > drop_d$$

where $sim(\vec{cv_c}, \vec{dv_d})$ *is the cosine similarity between the document-vector* $\vec{dv_d}$ *and the concept-vector* $\vec{cv_c}$*, and* $drop_d$ *represents the critical drop computed as in Algorithm 1.*

Notice that in general the associations of different concepts are not disjoint, since the same object can be assigned to multiple (similar) concept-vectors. The number of concepts associated to a given document depends on the corresponding adaptive threshold value computed by the association algorithm. Notice that, at this point, given all the calculated associations, for each concept c_i, we can infer the set of documents associated to it.

Concept-to-Document Association

Depending on the document data and their degree of matching with the different concept-vectors of the taxonomy, the above association process may not result in a uniform distribution of documents across the different associations. In particular, there can be cases where some concept nodes of the taxonomy do not appear in the list of the "best candidates" (i.e., the list of concepts above the adaptively computed threshold) for any document. For such concept node, their corresponding associations are thus empty at the end of the association process.

These cases might affect the quality of the navigation and search, which are based on the evaluation of mutual relationships among the associations of

the concepts. In fact, considering that the taxonomy is usually generated by a domain expert, we believe that each concept-node in it should essentially represents a portion of the knowledge expressed by the considered contents (and if it is not case, the domain expert should basically remove it). Thus, we also consider a dual, concept-to-document association approach in order to determine a new association, $A_{c \to d}(\vec{cv})$, that avoids this problem.

The algorithm differs from the previous one only as far as the external control loop is concerned: for every concept-vector, the association is computed by finding the best matching document-vectors.

Again, for each concept c_i, we dynamically set the relevant documents related to it by applying the previously reported cut-off (Section 3.2). Thus, we can define $A_{c \to d}(\vec{cv})$ as

Definition 3.2.2 (Association $A_{c \to d}(\vec{cv})$)

$$\forall d \in D, d \in A_{c \to d}(\vec{cv}) \iff sim(\vec{cv_c}, \vec{dv_d}) > drop_c$$

where $sim(\vec{cv_c}, \vec{dv_d})$ returns the cosine similarity between the concept-vector $\vec{cv_c}$ and the document-vector $\vec{dv_d}$, and $drop_c$ represents the critical drop computed as in Algorithm 1.

In the concept-to-document approach, the association of a concept is empty only if the associations of all the concepts are empty. Thus, in almost all cases, at the end of this phase, all of the concepts nodes have a non-empty association. Notice that, once again, associations are not necessarily disjoint, since the same object can be assigned to multiple (similar) concept-vectors.

3.3 Discovery of Concept-Keyword Relationships

Using the previously described methods, we can associate the documents to their best representative concepts in the taxonomy. Both methods can be considered as *unsupervised document associations*, where each association has been entirely created just considering the information contained in the documents and the concept vectors.

At this step, depending on the user needs, we may be interested in improving the quality of the associations through an additional step, where a relevance feedback mechanism provides more precise information that can improve the documents classification process. Moreover, this relevance feedback-based approach allows the system to discover new hidden relationships among the relevant terms (contained in the given corpus of documents)

and the original concept/categories of the given taxonomy, creating new semantic connections between both input data.

Considering a concept c_i and its association, we aim to search for the most contextual informative keywords. For this, we treat each document in the related association as a bag of words (the keywords extracted from the orginal text). As discussed in [133], we compute the degree of matching between the keyword and the concept by treating each document contained in the association as a positive relevance feedback and each document containg the keyword but not in the concept association as a negative relevance feedback.

In other words, this phase aims to find those keywords better describing the concept in the chosen context documents. Therefore, given a concept-vector and a corresponding association, this process aims to identify keywords (and their weights) that are significant for the characterization of the concept in the given context. For this purpose, we treat (a) the node document-vector as a query and (b) the association set as a contextual feedback, and we apply a probabilistic feedback process.

Definition 3.3.1 (Concept-Keyword Relationship) *We define the relationship that exists between a keyword k_i, extracted from the considered corpus, and a hierarchy concept c_i, as the weight u_i computed as follows [134]:*

$$u_i = log \frac{r_i/(R - r_i)}{(n_i - r_i)/(N - n_i - R + r_i)} \times \left| \frac{r_i}{R} - \frac{n_i - r_i}{N - R} \right|$$

where:

- *r_i is the number of document in the association containing the keyword i*
- *n_i is the number of documents in the corpus containing the keyword i*
- *R is the cardinality of the association*
- *N is the number of documents in the collection*

Intuitively, the first factor increases when the number of the documents containing the keyword k_i increases, while the second factor decreases when the number of the irrelevant documents (i.e., not belonging to the considered association) containing the keyword k_i increases. Therefore, keywords that are highly common in a specific association and not much present in others will get higher weights. For each concept, we consider all keywords contained

in at least one document of the concept association that have a positive weight value. Similarly to association phase, we apply the adaptive cut-off to this set in order to select the most relevant keywords with the highest weights that will form the *enriching-keyword* vector $\vec{ekv}_{c_i}$.

At this point, for each concept c_i, we have two vectors:

1. the concept-vector $\vec{cv}_{c_i}$ representing the concept-concept relationships in the corresponding taxonomy
2. the enriching-keyword vector, $\vec{ekv}_{c_i}$, consisting of keywords that are significant in the current context defined by the corpus.

In order to combine the concept and the enriching-keyword vectors into a single **extended-concept vector**, defined as

$$\vec{ecv}_{c_i} = \alpha_{c_i} \cdot \vec{cv}_{c_i} + \beta_{c_i} \cdot \vec{ekv}_{c_i},$$

we need to first establish the relative impacts (i.e. α_{c_i} and β_{c_i}) of the taxonomical knowledge versus real-world background knowledge.

Therefore, given concept c_i, let

- $S(\vec{cv}_{c_i})$ be the set of documents associated to the concept c_i (i.e. the documents retrieved from querying the database using the concept-vector, $\vec{cv}_{c_i}$); and
- $S(\vec{ekv}_{c_i})$ be the set of documents obtained by querying the database using the enriching-keyword vector, $\vec{ekv}_{c_i}$.

We quantify the relative impacts, α_{c_i} and β_{c_i}, of the concept and enriching-keyword vectors, $\vec{cv}_{c_i}$ and $\vec{ekv}_{c_i}$, by comparing how well $S(\vec{cv}_{c_i})$ and $S(\vec{ekv}_{c_i})$ approximate $D_{d\to c}(\vec{cv}_{c_i})$. In other words, if

- $C_{c_i} = D_{d\to c}(\vec{cv}_{c_i}) \cap S(\vec{cv}_{c_i})$ and
- $EK_{c_i} = D_{d\to c}(\vec{cv}_{c_i}) \cap S(\vec{ekv}_{c_i})$,

then we expect that

$$\frac{\|\alpha_{c_i} \cdot \vec{cv}_{c_i}\|}{\left\|\beta_{c_i} \cdot \vec{ekv}_{c_i}\right\|} = \frac{|C_{c_i}|}{|EK_{c_i}|}.$$

If the concept and enriching-keyword vectors are normalized to 1, then we can rewrite this as

$$\frac{\alpha_{c_i}}{\beta_{c_i}} = \frac{|C_{c_i}|}{|EK_{c_i}|}.$$

Also, if we further constrain the extended concept vector $\vec{ecv}_{c_i}$ to be also normalized to 1. I.e.,

$$\left\| \alpha_{c_i} \cdot \vec{cv}_{c_i} + \beta_{c_i} \cdot \vec{ekv}_{c_i} \right\| = 1,$$

then, solving these equations for α_{c_i} and β_{c_i}, we obtain:

$$\alpha_{c_i} = \frac{|C_{c_i}|}{|C_{c_i}| + |EK_{c_i}|} \quad \text{and} \quad \beta_{c_i} = \frac{|EK_{c_i}|}{|C_{c_i}| + |EK_{c_i}|}.$$

Thus, given concept, c_i, we can compute the corresponding extended concept vector as

$$\vec{ecv}_{c_i} = \frac{|C_{c_i}|}{|C_{c_i}| + |EK_{c_i}|} \cdot \vec{cv}_{c_i} + \frac{|EK_{c_i}|}{|C_{c_i}| + |EK_{c_i}|} \cdot \vec{ekv}_{c_i}.$$

Since, at this point, each concept in the original taxonomy has its own extended concept vector $\vec{ecv}$, the documents in the given corpus can be associated under these nodes, but using $\vec{ecv}$ vectors instead of $\vec{cv}$ vectors. Again, depending on the needs, we can select the most suitable association methods between the two approaches presented above. In this manner, using the extended vectors, we are able to associate to each concept not only the documents that contain that concept name, but also the documents containing some of the contextually relevant concepts and keywords.

3.4 Experimental Evaluation

In order to prove the efficacy of these pre-processing operations on meta-data hierarchies, in this Section we analyze the benefits of applying these techniques on taxonomical structures. In particular, as proposed in [21], we study the advantages of using a context-aware enrichment (from the considered corpus) of the structural information (dictated by the hierarchy itself) to better describe the meta-data knowledge. In fact, given a meta-data structure, it is important to describe its knowledge taking into account not only the hierarchical structures but also the *context* in which they will be used.

Moreover, we argue that the advantages of a pure structure-based meta-data knowledge definition (as the CP/CV algorithm, originally proposed

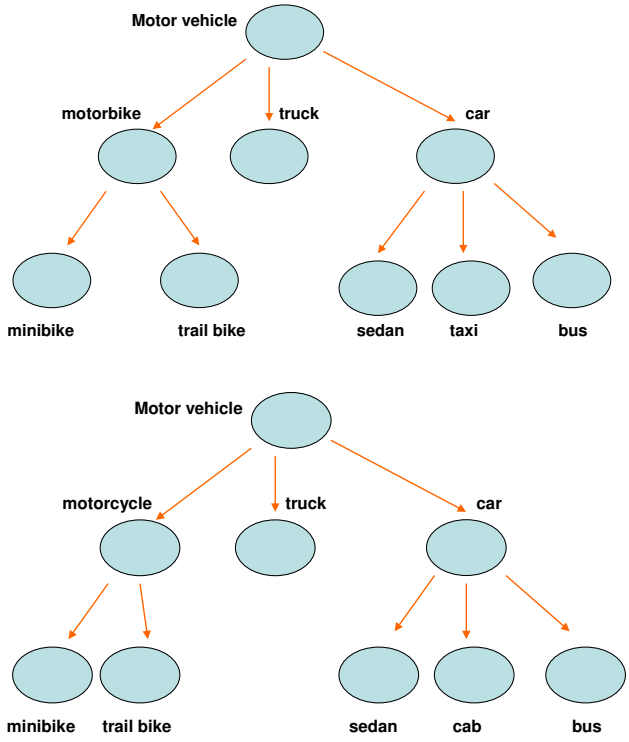


Figure 3.2: Two meta-data hierarchies about "*motor vehicles*".

in [76] and described in Section 3.1.1) can be positively combined with a context-informed knowledge definition. In fact, the method described in this chapter not only formalizes the structural information coded by the hierarchical structure (concept-vectors described in Section 3.1.1) but also leverages them to infer contextual extensions (as document associations, Section 3.2) of the concepts. This approach can help resolve conflicts and mismatches that might arise if only structural aspects are considered.

Thus, we prove our association-based meta-data definition by evaluating the capacity of our method to disambiguate the nodes contained in the given hierarchies. In other words, given two meta-data hierarchies, we believe that a good meta-data knowledge formalization approach should be able to define each node in such a way to recognize, if there are, similarities/dissimilarities among different meta-data structures, and recognize where they match and where they differ.

For example, let us consider the meta-data hierarchies presented in Fig-

ure 3.2; even if they represent the same domain knowledge (about "*motor vehicle*"), they report the same contents with different labels. Therefore, we believe that an effective definition of the meta-data information should permit to highlight where (and if) they represent the same content knowledge. For example, the concept nodes "*taxi*" and "*cab*" represent the same entity; thus, an effective meta-data knowledge representation should permit to identify these correspondences.

In order to prove these assumptions, we evaluate the benefits of using a context-informed meta-data definition for meta-data disambiguation purposes by studying the differences against pure structural taxonomical definition approaches. In particular, given a meta-data structure, we formalize its knowledge by using the approach described in Section 3.2; thus, after the taxonomical vectorization (Section 3.1.1), we leverage these concept-vectors to associate to each meta-data node a set of documents that best describe them. Then, we compare the proposed association-based meta-data knowledge definition (called in the experiment `Class`) against two pure structural meta-data node definition techniques:

- `CP/CV` which, as described in Section 3.1.1 formalizes the meta-data knowledge by associating to each taxonomical node a concept-vector that describes its structural relationships with all the other nodes in the meta-data hierarchy;

- common-ancestor distance (`Anc`), which defines each meta-data node by taking into account its distance from all its ancestors in the hierarchy (i.e., each node is defined by its counting the distance wrt all the ancestors in the hierarchy).

Then, we perfomed several evaluation experiments by considering a meta-data hierarchy extracted from DMOZ[2]. The considered hierarchy has 72 nodes, depth 4, and different branching factors in its internal nodes (the average value is 5.14). Then, we classify 17420 article abstracts describing NSF awards for basic research [3].

To evaluate the effectiveness, in terms of disambiguation capacity, of the proposed association-based meta-data definition strategy, in the presence of different conditions, we then created several (and similar) other hierarchies to be matched against. Thus, considering the original meta-data hierarchy extracted from DMOZ, we create alternative structures to

[2]accessible at the link *http://www.dmoz.org/Science/*

[3]http://kdd.ics.uci.edu/databases/nsfabs/nsfawards.html

be matched against, by introducing controlled distortions to the original hierarchy. These distortion permit to maintain the structure of the considered meta-data hierarchy but introduce small modifications of the concept nodes; in particular, we introduce the following distortions:

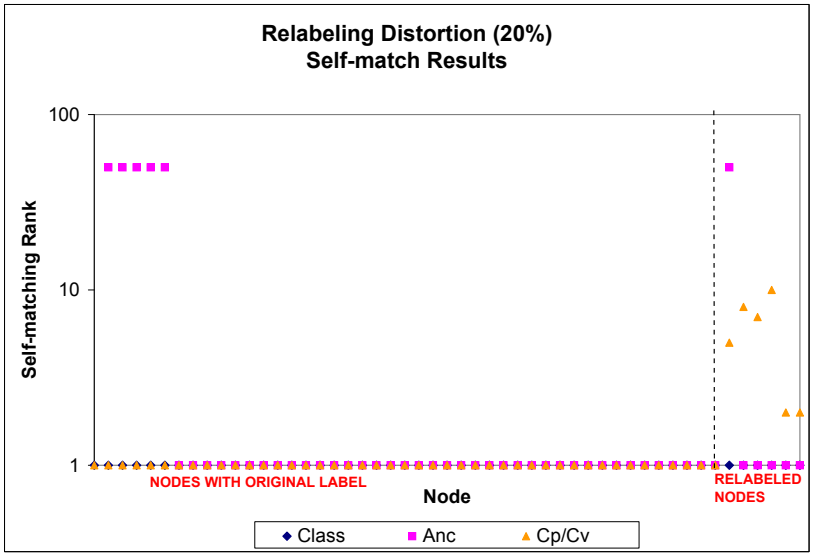
- *synonyms :* we randomly pick a percentage of nodes and relabel their concept names with other terms (without affecting the structure of the hierarchy). Note that the new labels are actually random (i.e., not real English synonyms) and do not actually occur at all in the data corpus. This constitutes a worst case situation for association-based algorithms (that also leverage the node labels for classification purposes).

- *Homonyms:* we randomly picked a percentage of nodes and, for each of them, we introduced a replica in randomly selected positions of the other meta-data hierarchy. The replica has the same concept name, but it is contextualized in a different position in the structure of the hierarchy (i.e., it is a homonym).

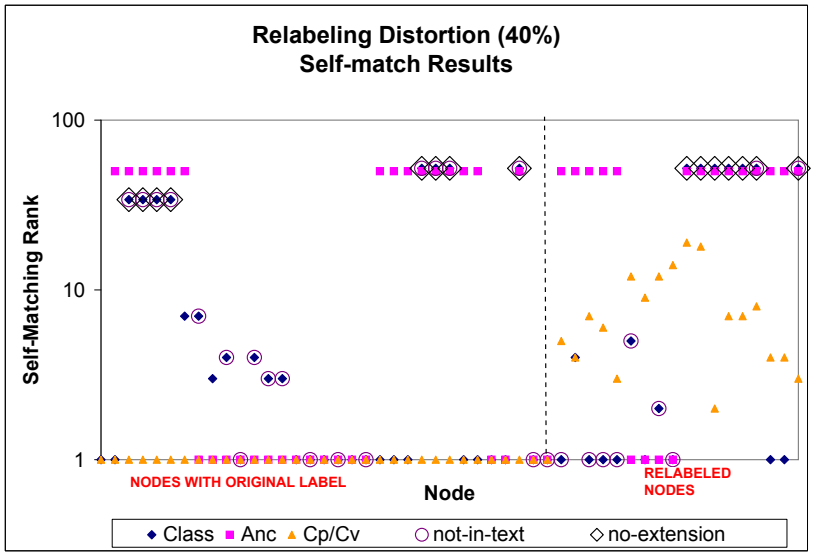
Considering these two conditions, we applied distortions of the order of 20% and 40% of the nodes. Thus, the main aim of the experiment is to prove that, leveraging the association-based definition of the nodes, it is possible to positively retrieve in the alterated hierarchy, the corresponding node.

Figures 3.3(a) and (b) show the synonym matching results. In this Figure, the X axis denotes the nodes and Y axis denotes the rank at which the corresponding node, in the distorted meta-data hierarchy, is found. Note that if the alignment algorithms works perfectly, then these distortion operations would not have any impact and all nodes will be found at rank 1.

- For the 20% distortion case (Figure 3.3(a), the portion relabeled is on the right), we observe that `Class` (which identifies the proposed association-based meta-data knowledge definition) is always able to retrieve the corresponding nodes in the compared structure (100% of exact matches). On the other hand, `Anc` (which identifies the meta-data knowledge definition based on the edge distance among the hierarchy nodes) makes some mistakes when also an ancestor node is relabeled (86.3% of exact matches). However, `CP/CV`-based formalization approach defines each node in such a way to be able to always retrieve, for non-relabeled nodes, the corresponding nodes in the alternative hierarchy but (since it relies on the concept labels to some



(a)



(b)

Figure 3.3: Matching results under concept re-labeling (rank=1 indicates an exact match).

degree) performs imperfectly for re-labeled nodes (88.3% of correct matches).

In short, while our meta-data formalization method is also based on the concept-vectors, `Class` improves the results by leveraging available document associations.

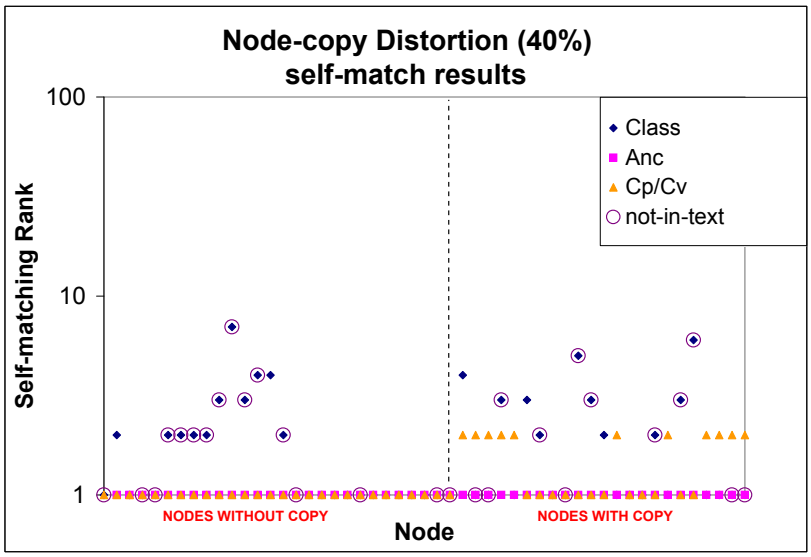
- On the other hand, when 40% of the nodes are arbitrarily relabeled (Figure 3.3(b), the portion relabeled is on the right), the impact on the performances are significant: in fact, `Anc` makes significant errors (only 42.2% of exact matches). However, `CP/CV` works very well for non-relabeled nodes and performs imperfectly for re-labeled nodes (totally 64.8% of exact matches). In contrast, `Class` performs very well even in this heavily distorted situation (83.4% of exact matches).

Figures 3.4(a) and (b) show the matching results in the presence of multiple concepts with identical labels. Figure 3.3(a) presents the matching ranks for the corresponding nodes in the altered hierarchies, while Figure 3.3(b) shows the matching rank for the new inserted homonyms. Note that, for the former case, the closer to 1 the rank, the better are the results; whereas, for the homonym case, the further from 1 the ranks, the more discriminating is the algorithm. In fact, in this second case, further from 1 the ranks, better the formalization method permits to distinguish among the homonyms.

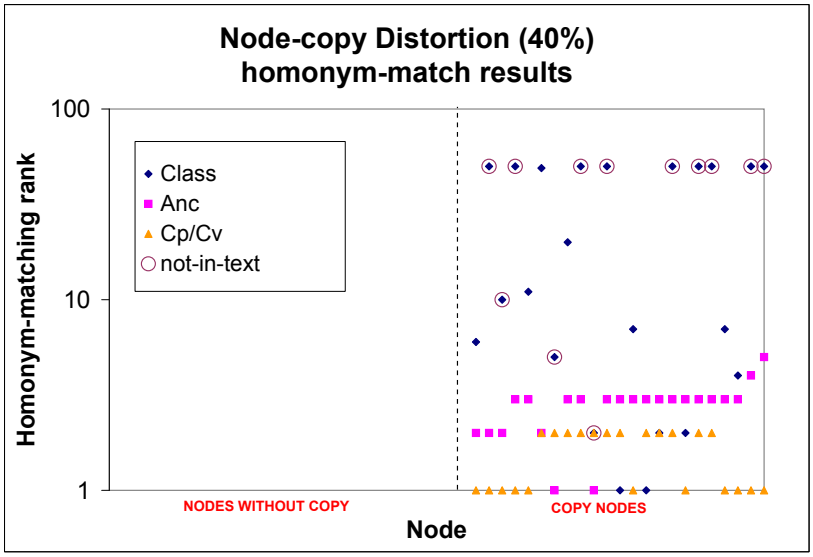
- For self-matching cases (Figure 3.4(a)), since the node-copy distortions do not affect any internal nodes, `Anc` works perfectly (100% of exact matches). On the other hand, `CP/CV`, which gets structural context also from descendants, introduces some errors (rank 2 instead of 1) but it is still able to retrieve the majority of corresponding nodes in the alternative structure (78.5% of exact matches). `Class`, however, works well (91.2% of exact matches) unless the concept does not occur in the corpus at all (these cases are marked with $\bigcirc$).

- For homonym-matching cases (Figure 3.4(b), we observe that `Class` works the best (puts the homonym furthest away from rank=1), whether the concept occurs in the corpus or not. In both cases, while the original concept is able to leverage the context provided by its neighborhood to classify documents, the arbitrarily picked context of the copy does not classify similar documents and homonyms are clearly identified.

Thus, the experimental evaluation showed that a properly context-informed meta-data definition can greatly help disambiguate the meta-data contents, enriching the structural information with corpus content knowledge. In fact,



(a)



(b)

Figure 3.4: Matching results under homonyms (for (a) rank=1 indicated perfect match while for (b) it is better to have match rank $\gg 1$.

a pure structural based knowledge definition can not guarantee these disambiguation performances, and poorly performs even when the similarities

among the considered structures are clearly evident.

Chapter 4

Adaptation of Hierarchical Meta-Data for Textual Documents' Management

Hierarchical meta-data structures can play a central role in the annotation and retrieval process of large data sets. From the viewpoint of knowledge engineers, meta-data embody formalized knowledge, which can be understood and reused by others. They are generally developed (or extracted) by human domain experts and represent a structured knowledge easily understandable by human users.

Generally, a better choice for an engineer is to reuse existing hierarchical structures rather than develop a new one from scratch, due to the fact that the domain knowledge definition could be a hard and time consuming task and it is often impossible to define a proper meta-data hierarchy for each considered data set. Moreover, even when the web communities are able to provide suitable hierarchies (in the last few years the interest in such field is increasing due to the emerging necessity of data organization), they are usually too detailed or they simply don't reflect the real distribution of the data. In fact when developers distribute a meta-data structure for a particular domain, this is usually very detailed because their aim is to provide a deep, as precise as possible, knowledge about the considered domain without any optimization for its final usages.

For example, depending on the context, an application can need a variable number of concepts (and related information) per meta-data or different organization about their relationships. Thus, this meta-data reuse generates a new question: *is there a feasible approach to extract a meta-data structure*

from existing ones?

Considering this problem, we proposed in [24] a new method for extracting knowledge from an existing hierarchical meta-data structure to produce an abridged version for a particular user (or users) and task (or tasks). In fact, given a general meta-data hierarchy, our work defines a new method for extracting the most relevant information from it by analyzing the original data and removing the redundant information from the considered structure. Moreover, the method re-defines the internal relationships among nodes in order to reflect as much as possible the real data distribution, avoiding (if present) redundancy and returning a structured knowledge that best represents the considered data set. Our method also permits to set the detail level requested by the user, thus allowing to select how much data can be preserved in the final structure.

In the next Sections we formalize these ideas based on the preliminary notation introducted in Chapter 3, and apply them on datasets consisting of collections of textual documents.

4.1 Preliminary Motivations

While there are many strategies for organizing text documents, hierarchical categorization –usually implemented through a pre-determined hierarchical meta-data structure– is often the preferred choice. In fact, hierarchical meta-data embody formalized knowledge easily understandable by human users and define relationships between concepts in a given domain.

In a hierarchy-based information organization, each category can index text documents that are relevant to it, facilitating the user in the navigation and access to the available contents. For example, an on-line educational site needs to present resources in a compact and understandable structure to help the user locate resources relevant to her interests and this task can be positively realized by using a hierarchical structure (Figure 4.1).

Unfortunately, given a set of text documents, it is not easy to find the appropriate categorization that best describes the contents. In fact the available hierarchical meta-data are usually designed for broad coverage of concepts[1] in a considered domain, failing to reflect important details (within the users' foci of interest) expressed by the considered data set. Especially in dynamically evolving domains, it might be the case that the available hierarchies do not necessarily reflect the content knowledge. For all these motivations, when developing a taxonomy, a better choice for an engineer

[1]In this thesis, we will use the terms "concept" and "category" interchangeably.

Figure 4.1: A scientific categorization example (used by NSF's National Science Digital Library web site, `http://nsdl.org`) to organize digital resources.

could be to investigate and reuse existing taxonomies rather than develop a new one from scratch (that would need a non-trivial analysis of the considered contents). In fact, depending on the context, the user can need a variable number of nodes (and related information) per taxonomy and different semantical relationships among the considered categories in order to facilitate particular search tasks.

Based on these considerations, in this thesis we introduce a new method [24] for distilling a meta-data hierarchical domain categorization from an existing one, based on a given set of text documents that have to be represented and indexed by the distilled taxonomy.

Note that any adaptation of a hierarchy– whether by adding new concept nodes or by collapsing and summarizing unnecessary details of it – implies a distortion of the original structure. But as long as this distortion is aligned

with the considered contents, it could not imply any loss of expressivity of the hierarchy or limit effective access to the underlying text content. In contrast, as long as the distortion is limited to where it matters, it will help improve effectiveness of search and navigation.

For all these purposes, we recognize that the primary role of a taxonomy is to describe or *narrate* the natural relationships between concepts in a given domain to its users. Therefore, a contextually relevant adaptation of a taxonomy should essentially distill and manipulate the structure of the existing taxonomy by appropriately segmenting and, if needed, summarizing this narrative relative to the documents in a given corpus. Based on this key observation, we propose *A Narrative-based Taxonomy Adaptation* method, `ANITA`, our hierarchical meta-data structure distillation approach for adapting existing taxonomies to varying application contexts. In particular, we introduce:

- *The narrative view of a taxonomy:* we view a taxonomy as a discourse introducing the general domain topics (the higher-levels of the taxonomy) and then going into further details (lower levels in the hierarchy). As described in Section 4.2.1, we transform each category in the original taxonomy into a *sentence* by associating to each concept a vector of weighted related terms extracted from the corpus. Then, we order these sentence-vectors (Section 4.2.1) in such a way to reflect both the semantical relationships among the categories and the structural constraints expressed by the hierarchy.

- *The segmentation of the narrative*: this narrative, which preserves the structure of the taxonomy (e.g., structural-relationships between the concepts), is then segmented based on a narrative-development analysis, highlighting where the narrative significantly drifts from one topic to another (Section 4.2.2).

- *The re-construction (or distillation) of an adapted taxonomy based on the segmentation results:* the resulting narrative segments (each describing a group of concepts/categories that collectively act as a single topic) are re-organized into a hierarchical structure, linking each concept-segment to others that are structurally related to it (Section 4.2.3).

The result of the above process is a contextually-relevant *adapted meta-data structure*, where details are highlighted where they matter and suppressed where they do not support the current context. In Section 4.4, we evaluate the proposed scheme using different text collections.

4.2 Narrative-Driven Meta-data Adaptation

Given an input hierarchical meta-data structure $H(C, E)$ where $C = \{c_1, \ldots, c_n\}$ is the set of n concept nodes (or categories) and E is the set of structural edges, our goal is to create an adapted taxonomy $H'(C', E')$, based on a given context defined by a corpus, D, of text documents. As mentioned before, `ANITA` relies on a "narrative" interpretation of the input taxonomy to achieve this goal; unlike the original taxonomy, which is hierarchical, the narrative is linear in structure. However, it is created in such a way that the structure of the narrative corresponds to the structure of the hierarchy. More specifically, the scope of each concept (represented as a sentence) is contextualized by those that precede and follow it, and this contextual scope corresponds to both the structural information (coming from the original structure) as well as the content of the considered corpus. Experiments reported in Section 4.4 show that `ANITA` is able to leverage this narrative to improve the effectiveness of the adaptation process with respect to more generic clustering-based approaches, which cannot represent the structural context.

As described before, our method consists of three steps.

1. In the first step, we analyze the input hierarchy to obtain a *narrative view* that reflects the structural relationships between the concept-nodes in the hierarchy.

2. In the second step, we analyze the narrative to identify boundaries of *coherent segments* where the narrative drifts from one topic to another. Intuitively, each of these topics are sets of concept-nodes that are focused around a central concept in the current context.

3. Finally, the last step re-constructs an adapted hierarchical meta-data structure based on partitions of nodes returned by the previous step. Each partition is represented by a unique label that represents the central concept and these are organized in a tree structure which preserves the original hierarchy *as much as possible*.

In the next Sections, we present the details of each of these steps.

4.2.1 Step I: Narrative View of a Taxonomy

In this Section, we first introduce the narrative interpretation and then describe the taxonomy adaptation process in detail.

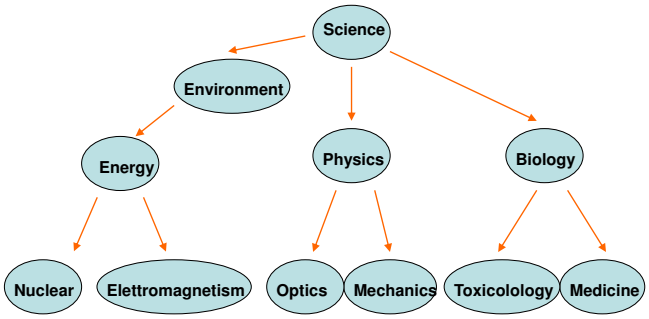


Figure 4.2: A portion of a hierarchical meta-data structure about *science*, extracted from DMOZ.

Step Ia: Concept-Sentences

Whereas a taxonomy is a hierarchy of concept-nodes, a *narrative* is a sequence of sentences. Therefore, in order to create a narrative corresponding to the taxonomy, we need to map concept-nodes of the input taxonomy into **concept-sentences**. What we refer to as concept-sentences are not natural language sentences, but vectors obtained by analyzing the structure of the given taxonomy and the related corpus of documents. Intuitively, these sentence-vectors can be thought of as being analogous to *keyword-vectors* commonly used in representing documents in IR systems.

Concept-sentences associate to each concept a coherent set of semantically related keywords, extracted from the associated text corpus. Thus, for each concept c_i in the considered hierarchy, we associate a sentence-vector $\vec{sv}_{c_i}$ as

$$\vec{sv}_{c_i} = \{w_{i,1}, w_{i,2}, w_{i,3} \cdots w_{i,v}\}$$

where v represents the total number of considered terms (the corpus vocabulary and labels in the taxonomy), and $w_{i,j}$ is a weight quantifying the degree of the semantical correlation between the j-th term and the i-th taxonomical concept. Table 4.1 reports the sentence-vectors, related to the taxonomy fragment shown in Figure 4.2, which include concepts from the taxonomy as well as keywords from the data set.

Concept-sentences can be obtained in many different ways; [25], [32], [147] propose various approaches that leverage semantic similarities between concepts in a given context for obtaining such vectors. In [32], the authors model the context of a certain term as a vector

$\vec{sv}_{science}$	{**science**, **student**, education, **physics**, teacher $\cdots$}
$\vec{sv}_{environ.}$	{**environment**, **science**, ecology, **energy**, earth $\cdots$}
$\vec{sv}_{physics}$	{**physics**, quantum, particle, **mechanics**, theory $\cdots$}
$\vec{sv}_{biology}$	{**biology**, **energy**, genetic, cell, ecology, student, biochemistry $\cdots$}
$\vec{sv}_{energy}$	{**energy**, **environment**, **electromagnetism**, thermodynamics, conservation $\cdots$}
$\vec{sv}_{optics}$	{**optics**, **physics**, light, **science**, radiation $\cdots$}
$\vec{sv}_{mechanics}$	{**mechanics**, **physics**, force, **science**, quantum$\cdots$}
$\vec{sv}_{toxicology}$	{**toxicology**, **biology**, department, student, **science** $\cdots$}
$\vec{sv}_{medicine}$	{**medicine**, safety, disease, **science**, policy $\cdots$}
$\vec{sv}_{nuclear}$	{**nuclear**, cell, power, **physics**, particle $\cdots$}
$\vec{sv}_{electromag.}$	{**electromagnetism**, interaction, **physics**, **science**$\cdots$}

Table 4.1: (a) The sentence-vectors ($\vec{sv}$), referred to the taxonomy fragment in Figure 4.2, obtained applying the method described in Section 3.3 using the NSF document set (described in Section 4.4). The sencence-vectors are ordered based on the corresponding weights which are omitted in the figure for clarity. Terms that are not in bold are picked from the NSF document corpus.

representing syntactic dependencies which are automatically acquired from the text corpus with a linguistic parser. On the other hand, in [147], the authors describe an unsupervised WordNet-based system that is able to determine the meaning of a term by analyzing semantic relatedness with respect to the most related terms in the considered context.

In this thesis, we use our approach presented in [25] and described in Section 3.3 to associate to each concept a keyword-vector (also called extended concept-vector), that tightly integrates terms extracted from text documents and labels of concepts obtained from the considered domain taxonomy. This approach leverages the structural domain knowledge (analyzing the structural relationships among the concepts nodes) to associate to each concept a set of relevant text documents from the considered corpus. Then, it is possible to leverage these associations to extract a relevant set of terms semantically related to the considered concepts. Thus, the resulting vectors reflect both the structural context (imposed by the meta-data hierarchy) and the documents content (imposed by the corpus).

Step Ib: Sentence Ordering

As we mentioned earlier, we recognize that the primary role of a meta-data structure is to describe or narrate to its users the natural relationships that exist among the considered concepts in a given domain. Thus, after the vector-based encoding of the *concept-sentences*, the next step is the creation of the narrative by ordering these sentences (therefore the nodes in the

original hierarchy) in an order representing the structure of the taxonomy.

Ancestor-Descendant Ordering. We consider three different narrative orders: the pre-order, parenthetical and post-order traversals of the taxonomy.

- *Pre-order Traversal of the Taxonomy:* a hierarchy (especially a concept hierarchy) is structured in a way that the most general concept is used as the root of the hierarchy and the most specific ones are the leaves. In a sense, each node provides more specialized knowledge within the context defined by all its ancestors. We leverage this aspect to define a narrative in which the sentences associated to the nodes of the taxonomy are read in pre-order; i.e., each concept-sentence is immediately followed by its detailed description in terms of its specializations.

- *Post-order Traversal of the Taxonomy:* this traversal generates a narrative in which the different concepts are presented bottom-up: after presenting the most specific concepts, their super-concept is narrated. Any super-concept presented after the narration of its children can be seen as summarizing the description of its sub-concepts.

- *Parenthetical Traversal of the Taxonomy:* intuitively, the parenthetical traversal is analogous to a narrative where each passage is presented with an *introduction* and goes in *details* until a general *conclusion*. In parenthetical traversal of the tree, each parent node is visited twice, representing both the general introduction and the conclusion to the argument that the children specialize.

Distance-Preserving Sibling Ordering While pre-order, post-order and parenthetical traversal of the tree help us decide in which order ancestors and descendants are to be considered, they do not help us choose the order in which the siblings in the hierarchy are to be concatenated in the narrative.

Let us consider a node c_0 with m children $\{c_1, c_2 \cdots c_m\}$. Our primary goal is to ensure that the narrative is ordered in a way that reflects the similarities – or dissimilarities – among these m siblings (as well as their parent c_0). In fact, in a narrative, each argument is introduced by smoothly contextualizing its topic (reporting earlier sentences that introduce it) and drifts to the other topics by introducing and defining the context of the next argument. Therefore, each node should be anticipated by the concept that best

	biology	medicine	toxicology
biology	0	0.4	0.2
medicine	0.4	0	0.2
toxicology	0.2	0.2	0

(a)

biology	1rst
toxicology	2rth
medicine	3nd

(b)

Table 4.2: (a) The dissimilarity matrix M obtained using the sentence-vectors for the concept-nodes "*biology*", "*medicine*" and "*toxicology*" and (b) the MDS-ordering of the children of "*biology*".

introduces it and followed by the node that can best specialize its knowledge. For example, in Figure 4.2 "*biology*" has two children, "*toxicology*" and "*medicine*"; if "*biology*" is more semantically related to "*medicine*" than "*toxicology*", we would like to order the narrative in such a way to preserve this information.

For this purpose, we first compute the dissimilarity matrix M based on the sentence-vectors corresponding to all $m+1$ concepts (the parent and the m children); in other words,

$$M[i][j] = 1 - cos(\vec{sv}_{c_i}, \vec{sv}_{c_j})$$

where the function *cos* measures the cosine similarity between the two vectors. We then use a distance-preserving embedding technique to map these concepts onto a one-dimensional ordering. In particular, without loss of generality, we use **multi-dimensional scaling** (MDS [154]), to embed the concepts onto a 1-dimensional order. MDS works as follows: given as inputs (1) a set of N objects, (2) a matrix of size $N \times N$ containing pairwise distance values and (3) the desired dimensionality k, MDS tries to map each object into a point in the k-dimensional space in such a way that a stress value, defined as

$$stress = \sqrt{\frac{\sum_{i,j}(d'_{i,j} - d_{i,j})^2}{\sum_{i,j} d^2_{i,j}}},$$

where $d_{i,j}$ is the actual distance between two objects o_i and o_j and $d'_{i,j}$ is the distance between the corresponding points in the resulting k-dimensional space, is minimized. Therefore, by providing as input $N = m+1$ concepts and $k = 1$ target dimension, the resulting order of concepts would preserve the semantic ordering between the concepts as best as possible. Notice that, due to the special nature of the node c_0 (it is the parent), we need to make a minor modification in the MDS algorithm: in particular, we constrain the stress minimization process in a way that forces the position of c_0 at the beginning of the list. This way, the resulting order of the children concepts

will reflect the concept similarities with respect to the position of the parent concept in the narrative.

As an example, let us re-consider the taxonomy fragment presented in Figure 4.2. In order to decide in which order the children of "*biology*", "*medicine*" and "*toxicology*", should be included in the narrative, we first calculate a dissimilarity matrix, $M_{biology}$, of these three nodes (Table 4.2(a)). Then, we apply the (slightly modified) MDS algorithm to obtain the ordering of the children with respect to the parent (Table 4.2(b)): first "*toxicology*" is included in the narrative and, then, "*medicine*".

Figures 4.3(a),(b) and (c) show the three distance preserving ordering approaches.

4.2.2 Step II: Segmentation of the Narrative

At this point the narrative is a sequence of sentences (or more precisely sentence-vectors), each including the information coming from the structural knowledge (hierarchy) and the context knowledge (documents), defining a global discourse that covers all the topics addressed by the taxonomy, according to the knowledge expressed by the contents. In this step, we analyze this narrative to identify segments (or partitions) that are highly correlated. The idea is that if, in the given corpus, two concepts are highly correlated, they may not need two separate nodes in the adapted meta-data hierarchy. In contrast, if there is a significant difference between two portions of the narrative, then these two portions (or segments) do necessitate different concepts in the resulting meta-data hierarchy.

In the literature, there are various techniques for segmenting a narrative into coherent units. Many authors proposed various techniques for segmenting texts into multi-paragraph units that represent passages or subtopics; the methods are based on lexical co-occurrence and distribution analysis. Some of these techniques, such as [119, 45] rely on the analysis of the topic evolution within the narrative to decide the positions of segment boundaries. Textile [59, 58] and Vectile [74] algorithms, for example, plot similarity scores (based on lexical co-occurrence and distribution analysis) of neighboring portions of the text. The dips (i.e., local minima) in the resulting similarity curve correspond to regions of the text where there is significant change in the content. Therefore, these dips are identified as text segment boundaries.

In order to partition the narrative $\vec{sv}_1, \vec{sv}_2, \ldots, \vec{sv}_n$ into coherent segments, we use a similar strategy. However, instead of searching for local minima of similarities, we seek partitions that correspond to similar **in-**

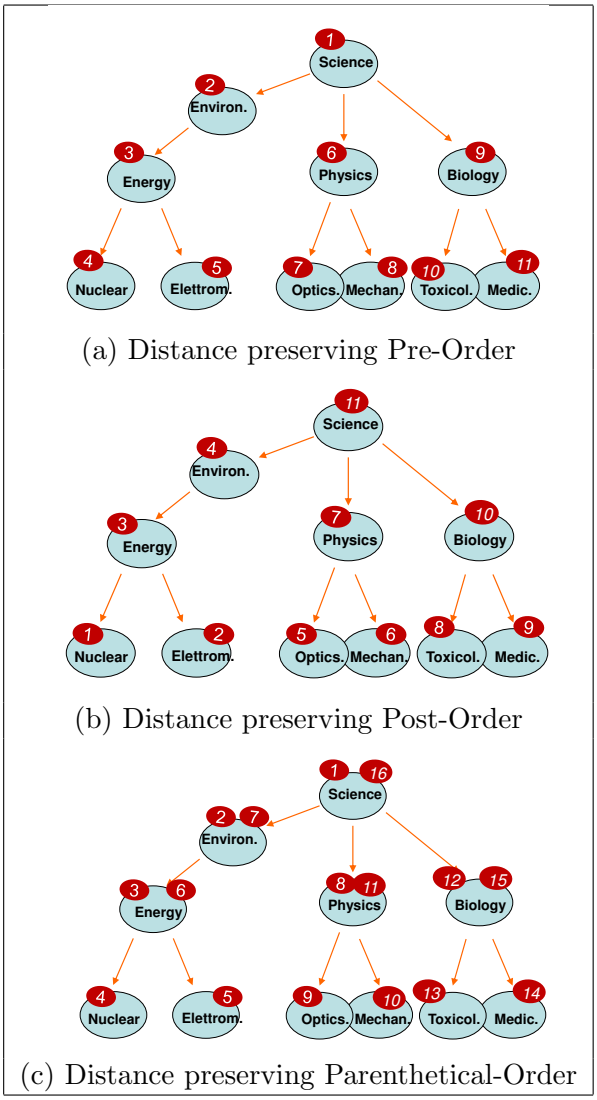


Figure 4.3: The narrative order (denoted by the circled number) for the hierarchy presented in Figure 4.2 based on a distance preserving pre-order, (b) distance preserving post-order and (c) distance preserving parenthetical ordering approach.

ternal coherence (defined in terms of the total amount of internal topic drift):

1. Given the narrative (i.e., ordered sequence of sentence-vectors), we first

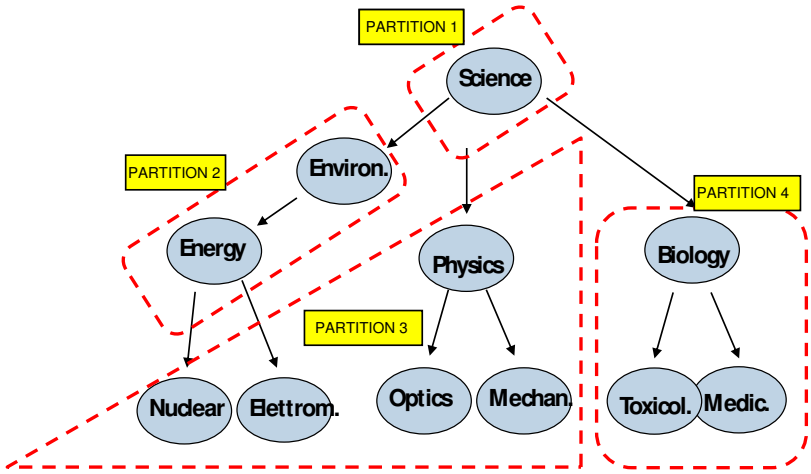


Figure 4.4: Narrative-based adaptation of the meta-data fragment presented in Figure 4.2: based on the structural constraints and the available contents, the meta-data nodes are grouped in 4 partitions.

compare each pair of neighboring vectors, $\vec{sv}_i$ and $\vec{sv}_{i+1}$ $(1 \leq i \leq n-1)$ by computing their *dissimilarities*:

$$\Delta_{i,i+1} = 1 - cos(\vec{sv}_i, \vec{sv}_{i+1})$$

2. The sequence of vectors is then analyzed for *topic drifting*. We say that a topic drift occurs for a given segment of the narrative when the degree of change between its starting and ending points is above a given threshold. If $Seg_{i,j}$ denotes a segment from the vector $\vec{sv}_i$ and $\vec{sv}_j$, the corresponding degree of drift is defined as $drift_{i,j} = \sum_{k=i}^{j-1} \Delta_{k,k+1}$.

   A segment $S_{i,j}$ is said to be *coherent* if it holds that $drift_{i,j} < \lambda_{max}$, where $\lambda_{max} = \frac{drift_{1,n}}{k}$ is the *coherence threshold*, and k is the target size of the summarized taxonomy[2].

At the end of the process, we obtain a set of segments, or partitions, $P = \{P_1, P_2, \cdots, P_k\}$ that represent sequences of coherent narrative compo-

[2]Note that the value of k can be set by the user/application depending on the visualization constraints (how much information can be shown in the display) and/or users' preferences

nents. Note that, each partition is a sequence of concepts from the original taxonomy and defines a single concept in the revised taxonomy.

Let us consider again the taxonomy presented in Figure 4.2; based on the NSF data corpus (described in Section 4.4), the meta-data hierarchy is partitioned in four groups of nodes (Figure 4.4). Note that the segmentation process can alter the structure of the hierarchy, since the relationships among concepts could change from one domain to another one. In fact, in popular/scientific magazine context, two concepts as "*nuclear*" and "*environment*" will result strongly related, while in the context of a scientific professional journal, the concept "*nuclear*" could be rigorously related to the concept of "*physics*" (in fact, as shown in Figure 4.5, considering the NSF awarded abstracts, "*nuclear*" has been connected to "*physics*"). Therefore, considering the knowledge expressed by the domain experts in the original taxonomy, `ANITA` tries to preserve the original relationships among concepts, but alters the structure when there is sufficient evidence in the corpus that a different structure would reflect the content better.

Notice from Figure 4.3(c) that the parenthetical traversal introduces each parent concept twice; in this case, if a parent node is associated to two different partitions, it is removed from the partition whose drift value (with respect to neighbor nodes in the sequence) is higher.

4.2.3 Step III: Hierarchy Distillation from the Partitions

In order to construct the adapted meta-data hierarchy from the partitions created in the previous step, we need to re-attach the partitions in the form of a tree structure. Furthermore, for each partition, we need to pick a *label* that will be presented to the user and will describe the concepts in the partition.

Step IIIa: Partition Linking

The adapted taxonomy, $H'(C', E')$ with $C' = \{c'_1, \ldots, c'_k\}$ (where each node c'_i represents the partition P_i) should preserve the original structure of $H(C, E)$ as much as possible. Thus,

- The root of H' is c_{root} $(1 \leq root \leq k)$ such that the corresponding partition P_{root} contains the root node of H.

- Let us consider a pair, P_i and P_j, of partitions in P. The decision on whether (and how) the corresponding concepts c'_i and c'_j should be connected is based on the following analysis. Let $E_{i,j}$ be the set of edges in E

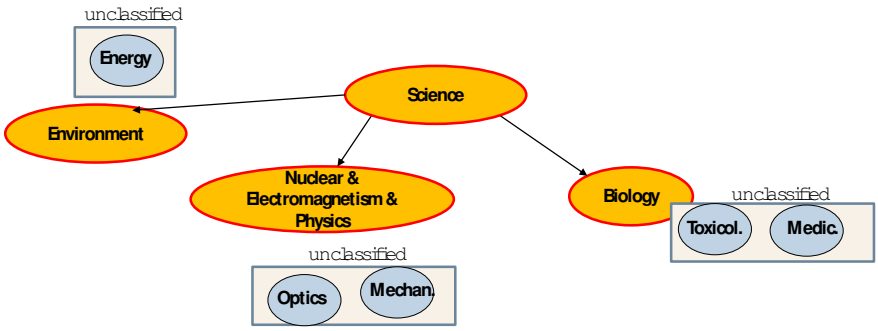


Figure 4.5: Meta-data hierarchy reconstruction process: based on the partitions shown in Figure 4.4 (a) the meta-data hierarchy is recostructed by linking the partitions to each other. Finally, each partition is labeled by selecting a representative label.

linking any concept in P_i to any concept in P_j. Similarly, let $E_{j,i}$ be the set of edges in E linking any concept in P_j to any concept in P_i. With the goal of preserving to the best the structure of H, we measure the strength of the structural constraints implied by E in H, and we propose as our solution the adapted taxonomy which maximally preserves such constraints.

Let $e = \langle c_a, c_b \rangle$ be an edge in H that connects two different partitions P_i and P_j (i.e. $c_a \in P_i$, $c_b \in P_j$). The strength of the structural constraint e, $strength(e)$, (i.e., the strength of the structural constraints induced by e) is $1 + d_b$, being d_b the number of descendants of c_b in H that also belong to P_j. Based on this, the decision of having the corresponding c'_i as the ancestor of c'_j is supported by the strength of the structural constraints associated to the edges in $E_{i,j}$.

Thus, the taxonomy H', is constructed by maximally preserving such constraints as follows:

1. create a complete weighted directed graph, $G_P(V_P, E_P, w_P)$, of partitions, where
   - $V_P = P$,
   - E_P is the set of edges between all pairs of partitions, and
   - $w_P(\langle P_i, P_j \rangle) = \sum_{e \in E_{i,j}} strength(e)$;
2. find a *maximum spanning tree* of G_P rooted at the partition P_{root}.

strength of the structural constraints among the partitions				
	partition 1	partition 2	partition 3	partition 4
partition 1	-	0	0	0
partition 2	2	-	0	0
partition 3	3	2	-	0
partition 4	3	0	0	-

Table 4.3: *strength* of the structural constraints among the partitions shown in Figure 4.4. These values reflect the number of edges that will be broken if two partitions will be not directly linked to each other.

For example, let us consider the taxonomy fragment and its partitions shown in Figure 4.4. In the adapted hierarchy (Figure 4.5), `ANITA` picks as root the partition containing the root node ("*science*"). Then, the remaining three partitions have been attached to it by analyzing the constraints given by the original edges.

In fact, considering the partitions shown in Figure 4.4 and the structural constraints imposed by the hierarchy (dictated by the edges of the taxonomy) the partition containing the concept "*physics*" (*partition 3*) could be attached to the *partition 1* (containing the node "*science*") or the *partition 2* (containing the concept "*envirnoment*"). But, as shown in Figure 4.5, our linking approach decides to attach the *partition 3* to the *partition 1* because the *strength* of this correlation is higher than the one with *partition 2* (3 structural constraints vs 2). In Table 4.3, the strength of all the structural constraints among the partitions retrieved in Figure 4.4 is shown.

Step IIIb: Partition Labeling

In order to select a representative label for each partition we need to analyze the obtained partitions in the context of the original structure. If there is a concept $c_i \in P_i$ that dominates all the other nodes in the partition (i.e., $\forall c_j \in P_i \quad c_j$ is a descendant of c_i), then the label of c_i is selected as the label of c'_i. If there is no such single node, then the minimal set D_i of nodes covering the partition P_i (based on H) is found, and the concatenation of the concept labels in D_i is used as the partition label. Intuitively, a concatenation implies that, in the given document context, these corresponding concepts are found to be not sufficiently distinguished from each other. On the other hand, any label that was in the original taxonomy, but is not included in the new taxonomy is found to be unnecessary in the new context.

Again, an example of the label strategy (referring to the partition shown in Figure 4.4) can be seen in Figure 4.5.

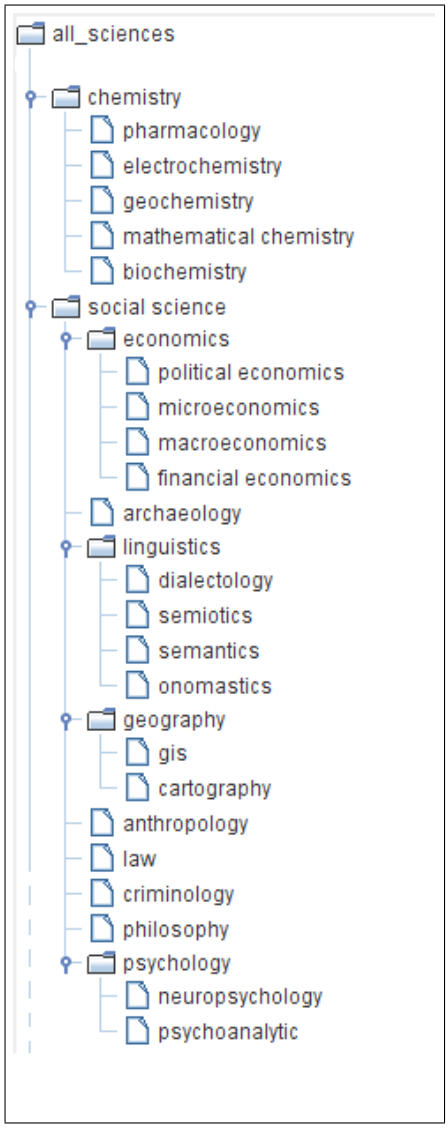


Figure 4.6: Scientific taxonomy fragment extracted from DMOZ.

4.3 Case Study

At the end of these steps, we are able to adapt a given hierarchical categorization in order to properly represent the knowledge expressed by a set of text contents. In this Section, in order to better explain the proposed method, we provide an example of adaptation of a given hierarchical meta-

data structure to a corpus of text document.

Let us consider the hierarchy fragment of a scientifical categorization in Figure 4.6 that can help a user navigate the abstract articles in the dataset consisting of the scientific abstracts from National Science Foundation. Applying the proposed narrative-based adaptation algorithm, we can now adapt the categorization as show in Figure 4.7 to the considered corpus, compacting redundant information and helping the user navigate the considered documents.

In this example, the granularity of the considered hierarchy is now reduced from 29 concepts to only 16 concepts (obtained by using $k = 16$, randomly choosen, and parenthetical ordering approach); a brief overview on the resulting structure highlights two important aspects:

- the most visible reduction has been obtained at the lowest level of the hierarchy (the leaves);
- the lenght of some node labels increased considerably (they have been merged to other labels).

These two aspects allow us to make some general considerations about our method and our assumptions. In fact, as explained in the introduction, we believe that any pre-determined hierarchical meta-data structure is generally developed in order to widely describe the considered domain knowledge. In order to do that, many un-necessary details (the degree of redundancy or overabundance varies depending on the application usage context) are introducted in the structure by adding concept nodes at the lowest levels of the hierarchy: in fact, if the highest internal nodes represent very general concepts that most of the domain experts probably share, the leaves represent details that can be interpreted or reported differently depending on the domain expert that defines the meta-data structure. Thus, our assumption is that, if these concepts can be superfluous or even redundant, it is possibile to re-define them (or even remove them) depending on the application usage context.

In fact, in the reported adaptation (Figure 4.7), we can easily notice that the internal nodes remain basically unchanged, while the most significant structural modifications have been performed on the leaves. For example, in the adapted taxonomy, the original categories "*pharmacology*", "*mathematical chemistry*", "*geochemistry*", "*electrochemistry*" have been collapsed and merged with their parent node "*chemistry*", representing them as a unique topic (the root of this sub-tree was selected as representative). Therefore, in this case, this decision has been supported by the corpus of documents

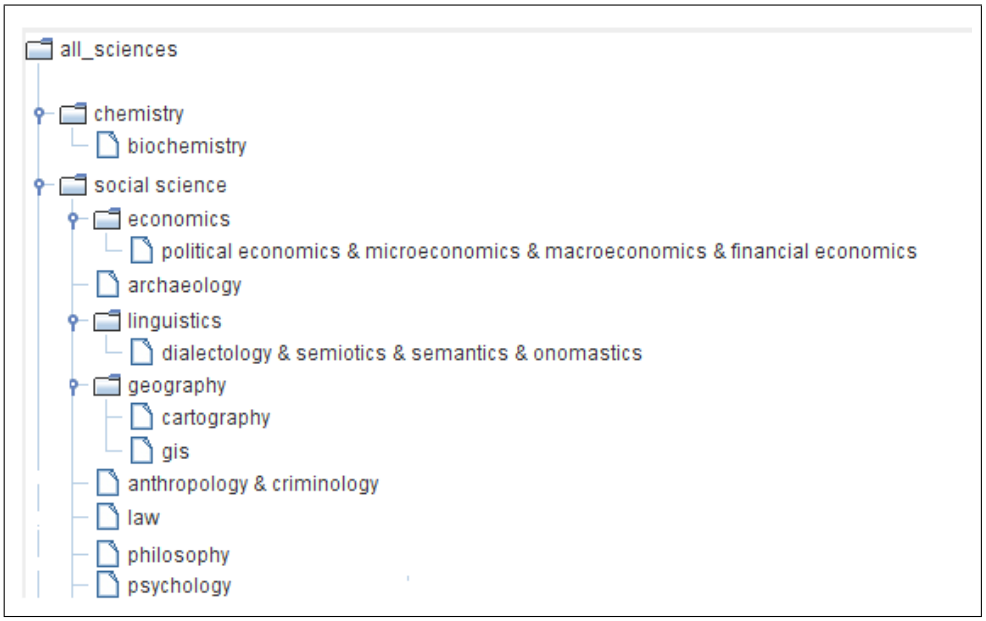


Figure 4.7: Adapted taxonomy based on the context defined by the NSF data set (described in Section 4.4).

where there was no significant difference among the documents reporting these categories initially represented in the hierarchy. However, the documents associated to "*biochemistry*" have been found as sufficiently different from the ones about "*chemistry*", supporting the decision to represent this concept as a separate topic in the resulting adapted meta-data structure. Obviously, the two new concepts have to be semantically related one to each other in the new meta-data structure; in fact, the method preserves as much as possibile the original relationships by connecting them in the new adapted hierarchical structure.

In contrast, the four children of "*economics*" have been collapsed into a single category node, highlighting the fact that no significant semantic difference was found comparing the documents associated to these concepts. In this case, it is possibile to notice that the representative label is obtained by merging all the single concept labels; in fact, in this case it is not possibile to find a unique node that dominates the others.

In conclusion, considering the reported example, the hierarchical meta-data structure has been considerably reduced in terms of its granularity and re-defined in terms of its concept relationships. Thus, it now reflects the

real distribution of the data, highlighting details where they matter and suppressing them where they are not sufficiently supported by the corpus of documents. Moreover, the granularity reduction makes it even suitable for devices with physical visualization constraints, compacting the relevant information in a smaller structure by reducing the redundancy as much as possibile. Notice that the new adapted taxonomy can be also considered as understandable by human users; however, in order to quantify the user-feedback while using adapted meta-data structures, we also provide user studies that show that the proposed algorithm is able to adapt the taxonomy in a new compact and understandable structure from a human point of view.

4.4 Evaluation

In this Section, we evaluate the performance of the proposed narrative-based hierarchical meta-data adaptation algorithm, `ANITA`, originally introduced in[24] and described in this chapter.

In our experiments, we used two different data sets:

- a corpus of news articles from New York Times (NY Times) data set[3] (∼64K text entries with over ∼100K unique keywords), and
- a set of scientific abstracts from National Science Foundation[4] (∼50K article abstracts describing NSF awards for basic research, with over ∼30K unique keywords).

Both data sets represent possible contents that need to be indexed and presented to the users through a navigational categorization hierarchy to allow an easy and fast navigation into the contents.

For each data set, we used a corresponding domain meta-data hierarchy extracted from the *DMOZ* categorization[5] by considering the most relevant terms, in the considered domains, extracted from the corpora. Specifically, we considered a hierarchy of science (with 72 nodes) which we used to index the NSF abstracts, and geographical hierarchy (181 nodes), against which we classified the articles from the NY Times. Note that, to avoid bias derived from the meta-data extraction process, we selected different subsets of these original hierarchies by randomly removing some of their nodes. Specifically, we created a total of 18 distinct hierarchies for each domain,

[3] http://archive.ics.uci.edu/ml/datasets/Bag+of+Words
[4] http://kdd.ics.uci.edu/databases/nsfabs/nsfawards.html
[5] accessible at the link *http://www.dmoz.org/*

obtained by removing anywhere between 10% to 60% (with 10+ increments, three different cases per percentage) of the concepts of the considered DMOZ hierarchy fragments. Moreover, for each of them, we considered different target meta-data sizes. The results in these sections are averages for all these hierarchies.

4.4.1 Meta-data hierarchy based Classification

As discussed in the introduction, meta-data hierarchies define aggregations between concepts in a given domain that can be easily used for indexing data in such a way to facilitate their organization. Thus, as in [164], we use the classification effectiveness as a measure of meta-data hierarchy quality (we formalize the measures in Section 4.4.2).

In our experiments, for each concept c_i in the considered hierarchy, we have a set of associated documents A_{c_i} that best match it. For taxonomy-based classification, without loss of generality, we rely on the vectorization method introduced in Section 3.2. These are vectors (associated to the meta-data hierarchy nodes) representing the structural relationships within the hierarchy, and thus not only can be used for measuring similarities of concepts to each other, but can also be used for computing the relationship of a document to a given concept by quantifying, as usual, the cosine similarity between the document keyword-vector (containing term frequencies) and the concept-vector.

4.4.2 Effectiveness Measures

In order to better understand the behavior of `ANITA` under different settings and to compare its performance to other algorithms on a concrete basis, we quantify the quality of the adapted meta-data hierarchies using three measures.

An important role of meta-data hierarchies in many applications is to help provide search and access to text documents. Thus, it is essential that they properly reflect the content of the corpus.

Definition 4.4.1 (Domain coverage) *Given a corpus of documents D and a meta-data hierarchy $H(C, E)$, the coverage of D by H is defined by the percentage of documents in D that can be associated to at least one concept in C using some classification process. Let $A_{c_i} \subseteq D$ be the set of documents associated to the concept $c_i \in C$, we define the domain coverage measure as*

$$cover(H, D) = \frac{\left|\bigcup_{c_i \in C} A_{c_i}\right|}{|D|}.$$

The main idea of the proposed method is to minimize the loss in terms of domain coverage while we potentially reduce the size of the given hierarchy. Thus, the higher the domain coverage, the more effective the hierarchy in covering the knowledge expressed by the considered corpus.

Note that it would be trivial to increase the domain coverage simply by concatenating more and more labels. This would not result in a desirable meta-data hierarchy. Therefore, it is important to quantify other properties, such as the degree of discrimination of the nodes of the hierarchy, along with domain coverage. Thus, we define the redundancy measure as

Definition 4.4.2 (Redundancy)

$$redundancy(H, D) = \frac{|overlap(D, H)|}{\left|\bigcup_{c_i \in C} A_{c_i}\right|},$$

where $overlap(D, H)$ *returns the set of documents in* D *associated to* at least *two concepts in* H.

Thus, this measure quantifies the discrimination power of the concepts in the resulting meta-data hierarchy, i.e, the degree of overlapping in the sets of documents associated to different concepts. The lower the redundancy, the higher the discrimination power, and thus the more effective the hierarchy in helping search and access text documents.

Finally, the *label term-length* (ltl) measure reports the average number of labels in the original meta-data hierarchy included in the labels of the adapted hierarchy.

Definition 4.4.3 (Label term-length) *Given an initial hierarchy* $H(C, E)$ *and its adapted version* $H'(C', E')$, *let* $length(label_{c'_i}, H, H') = l$ *iff* $label(c'_i) = label(c_1), \ldots, label(c_l)$, *with* $c_1, \ldots, c_l \in C$. *Then, label term-length is defined as*

$$ltl(H, H') = \frac{\sum_{c'_i \in H'} length(label_{c'_i}, H, H')}{|C'|}.$$

Intuitively, a concept with a concatenated list of labels corresponds to a composite concept. Since longer compositions will induce some confusion,

Context: NSF Corpus			
	cover.	redund.	Ltl
Pre-Order (sibling ord.)	0.123	0.551	1.724
Parenth. (sibling ord.)	**0.128**	**0.510**	**1.681**
Post-Order (sibling ord.)	**0.128**	0.530	1.702
Pre-Order (no sibling ord.)	0.125	0.729	1.423
Parenth. (no sibling ord.)	**0.128**	**0.725**	**1.402**
Post-Order (no sibling ord.)	**0.128**	0.736	1.463

Table 4.4: Impact of different narrative orders.

Context: NY Times Corpus			
	cover.	redund.	Ltl
Pre-Order (sibling ord.)	0.752	0.634	2.289
Parenth. (sibling ord.)	**0.759**	**0.573**	**2.204**
Post-Order (sibling ord.)	0.755	0.612	2.277
Pre-Order (no sibling ord.)	0.755	0.792	2.063
Parenth. (no sibling ord.)	**0.758**	**0.789**	1.966
Post-Order (no sibling ord.)	0.756	0.792	**1.809**

Table 4.5: Impact of different narrative orders.

arguably the lower the label length, the more informative is the label. If we consider for example the adapted meta-data hierarchy fragment in Figure 4.7(b), the composite concept "*political economics & microeconomics & macroeconomics & financial economics*", composed of 4 original labels, will be less precise than each individual concept in the list. Therefore, we roughly quantify this ambiguity by counting the labels that compose each concept name.

In the following experiments, we present results that rely on these three measures. In Section 4.4.9, we report the execution times. In Section 4.4.10, we then report user study results that quantify the impact of `ANITA` on the users' navigation experience.

4.4.3 Impact of the Narrative Orders

In Section 4.2.1 we introduced different sentence ordering approaches that help define, based on different interpretation of the meta-data hierarchy, the internal order of the nodes of the considered meta-data hierarchy. In this Section we analyze all the three proposed methods and we study their behaviour based on the measures introduced in Section 4.4.2.

Tables 4.4 and 4.5 present the values of the effectiveness measures for the three proposed narrative orderings (Section 4.2.1), with and without distance preserving sibling ordering. The values are averages of the performance results for five different target meta-data hierarchy sizes (from 10%

to 50% of the original number of concepts, with a 10+ increments).

From these two tables, we observe that sibling ordering results in slightly higher label term-length. This behavior is due to the fact that the ordering of siblings is likely to lead to longer sequences of similar siblings, which will be concatenated if the sequence does not contain the parent. It is important to note that this lengthening of the labels does not result in any increase in the redundancy of the resulting hierarchies. In all cases, the versions with sibling ordering have significantly smaller redundancies than the corresponding versions with the random ordering of siblings. The differences in terms of their domain coverages are negligible.

Considering the different traversal strategies, we observe that, for both data sets, parenthetical traversal provides lower redundancies and lower label term-lengths. Parenthetical traversal also provides the highest coverages, especially when distance preserving sibling ordering is used. Thus, in the rest of the evaluation Section, we only consider the parenthetical traversal with distance preserving sibling ordering.

4.4.4 Impact of the Corpus Context

The proposed hierarchical meta-data adaptation method relies on a statistical evaluation of the considered contents to properly adapt on the given meta-data structure. Thus, in this Section we evaluate the impact of using the information coming from the corpus context (represented by the sentence-vectors introduced in Section 4.2.1) on the proposed adaptation method.

In particular, we try to quantify the benefits of performing the data corpus information integration to support the pure structural information contained in the hierarchy itself. In other words, we compare the application of `ANITA` on the sentence-vectors (which are based on the extended-vectors described in Section 3.3) with a version of `ANITA` where the narrative structure is created based only on the original concept-vectors (Section 3.1.1) which reflect only the structure of the meta-data hierarchy and do not take into account the corpus in any way. Thus, in this second case, the sentence-vectors represent the original concept-vectors, without any integration of terms extracted from the considered data corpus.

The three charts in Figure 4.8 plot the performance ratios, $\frac{\texttt{ANITAwithcontext}}{\texttt{ANITAwithoutcontext}}$ for both NSF and NY Times data sets and for different target hierarchy sizes. For both data sets, the use of corpus context improves domain coverage and lowers the redundancy.

In terms of the lengths of the term labels, especially for very low target

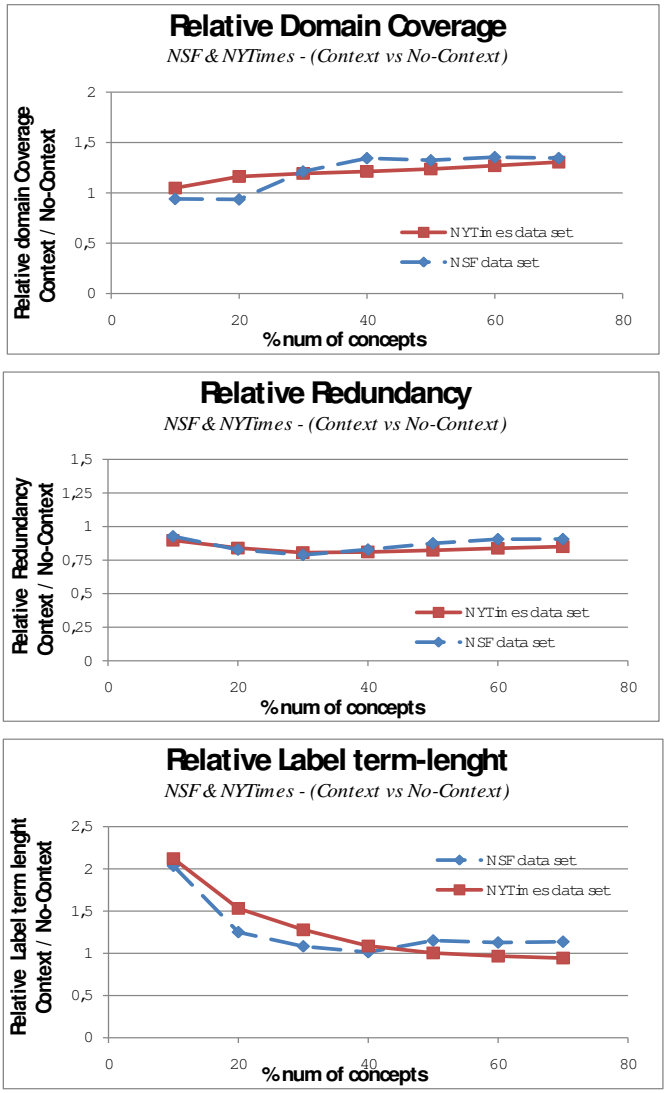


Figure 4.8: Domain coverage, redundancy, and label term-length ratio ($\frac{\texttt{ANITAwithContext}}{\texttt{ANITAwithoutContext}}$) curves. The two curves on each of the charts correspond to the NSF and NY Times data sets.

meta-data hierarchy sizes, the ratio is close to 2.0 for both data sets, indicating that the use of context results in longer descriptors. However, the ratio decreases significantly when the targeted size of the meta-data hierarchy increases (basically when the number of concepts is higher than 20%).

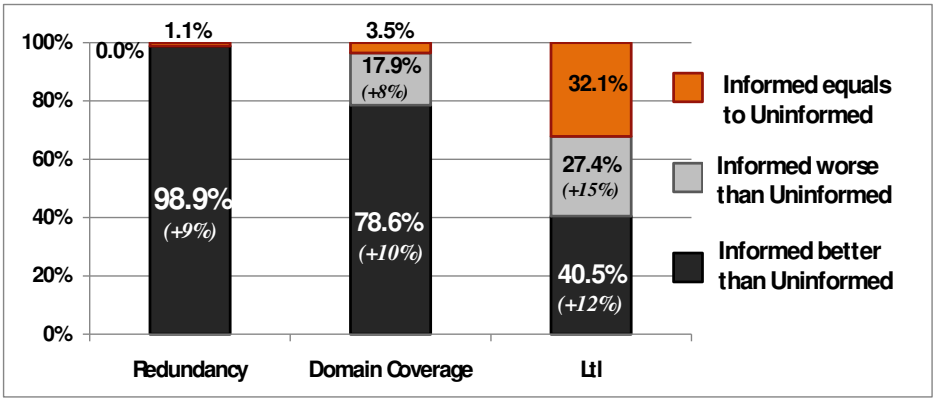


Figure 4.9: Comparison in terms of Domain coverage, redundancy, label term-length between Informed and Uniformed meta-data hierarchies (NSF data sets). The values in parentheses are the average gains by the winning scheme.

4.4.5 Impact of Document Context in Meta-Data Hierarchy Adaptation

One of the key motivations of the proposed meta-data adaptation approach is that a hierarchy that properly reflects the corpus knowledge can more precisely guide the user into its exploration instead of another structure not properly informed about it. Therefore, in this Section, we verify this hypothesis by evaluating the performance of `ANITA` adapted hierarchies, obtained without considering a subset of documents in the considered corpus, and we compare them against those informed about the entire data set. In order to do that, considering the 18 original taxonomies (and for each of them, a target meta-data hierarchy size between 10% and 70%) and their informed adaptations H' (using the entire set of documents represented by the NSF data set D), we evaluate two other different cases:

- considering the NSF data set without documents concerning "*biology*" ($D_{\neg bio} = D$ - D_{bio} where D_{bio} represents the set of documents containing the term "*biology*") we obtain the adapted hierarchies $H'_{\neg bio}$ which are *uninformed* about the documents concerning "*biology*";

- considering the NSF data set without documents concerning "*astronomy*" ($D_{\neg astr} = D$ - D_{astr}) we obtain the adapted hierarchies $H'_{\neg astr}$ which are *uninformed* about the document concerning "*astronomy*".

Therefore, in order to quantify the importance of properly informed adapted hierarchies, we compare the performance of each H' against the "*biology*" and "*astronomy*" uninformed adapted hierarchies $H'_{\neg bio}$ and $H'_{\neg astr}$, using the original corpus D of NSF text documents.

The comparison in terms of domain coverage clearly demonstrates the benefit of using the informed meta-data hierarchies instead of the uninformed ones: in 78.6% of the cases the informed structure permits to obtain higher coverages (with an average gain of $\sim 10\%$). Moreover, the benefits in terms of redundancy is also more evindent, providing in 98.9% of the cases a lower redundancy in terms of associated documents (with an average gain of $\sim 9\%$). Finally, in terms of label term-length, the informed clustering tends to provide lower label-length (with an average gain of $\sim 12\%$).

It is important to notice that, even in the small portion of cases in which the uninformed approach reports better performances, its relative gains are similar to those obtained when the informed approach reports better results.

4.4.6 Comparison wrt. other Segmentation Methods

While there are many segmentation approaches in the literature, we introduced a novel method (Section 4.2.2) to determine, given a sequence of sentences, the position of the segments' boundaries.

Thus, in order to evaluate the introduced method, we compare the obtained results against an alternative approach; in particular, we evaluate the performance of the `CUTS` algorithm, originally proposed in [119].

The authors of [119] provide a segmentation method that maps text entries into a curve in a way that makes apparent a variety of topic development patterns; then they analyze the curve for automatic segmentation of topics. In particular, `CUTS` algorithm works as follows; first, the sequence of entries (represented by their corresponding keyword-vectors) is mapped onto a curve, which highlights the development patterns in terms of the similarity between adjacent entries. This pattern development curve is then analyzed, and the topic segments, as reflected by the changes in the slopes of the curves, are identified (Figure 4.10 illustrates the main phases of `CUTS` algorithm).

In this Section, we compare the segmentation approach reported in Section 4.2.2 against `CUTS` algorithm [119], analyzing the benefits and the disadvatanges of using the proposed segmentation method.

Considering the NSF data set, in Figure 4.11 we plot the results in terms of domain coverage, redundancy, and label term-length for `ANITA` and `CUTS` algorithm; it is possible to observe that, for very small hierarchies (that

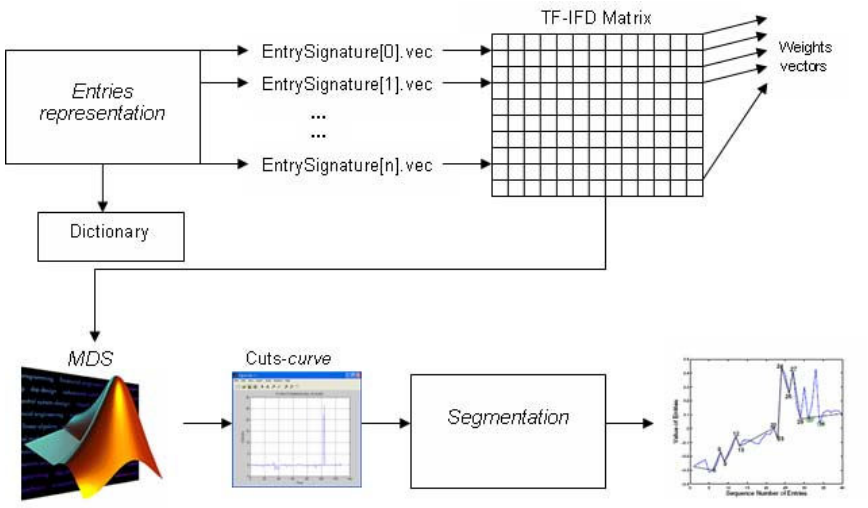


Figure 4.10: Overview of the CUTS segmentation algorithm [119].

consist in very low domain cover values) CUTS provides better results: but when the granularity of both increases, they converge to similar values.

Moreover, Figure 4.11 shows a significant difference in terms of label term-lenght: in fact, ANITA provides longer labels that, however, do not provide any loss in terms of redundancy, where the advantage of using ANITA segmentation method appears evident: in fact, even when the domain coverage increases, the redundancy provided by ANITA stays lower.

4.4.7 ANITA vs. Concept Clustering Methods

The proposed segmentation method, when applied on concepts within a hierarchical meta-data structure, can be easily seen as a clustering strategy that aims to group those nodes that are strongly related to each other (based on the considered context). Therefore, in this Section, we compare the narrative-based partitioning approach (Section 4.2.2) against other alternative clustering methods. In particular, we considered the k-Means clustering strategy, with k also being equal to the target meta-data hierarchy size requested from ANITA. In both cases, sentence-vector representation of the meta-data nodes are used to support partitioning. Also, in both cases, once the partitions are obtained, the same meta-data hierarchy re-construction and labeling strategies (described in Section 6.5.3) are used to stitch the hierarchy back.

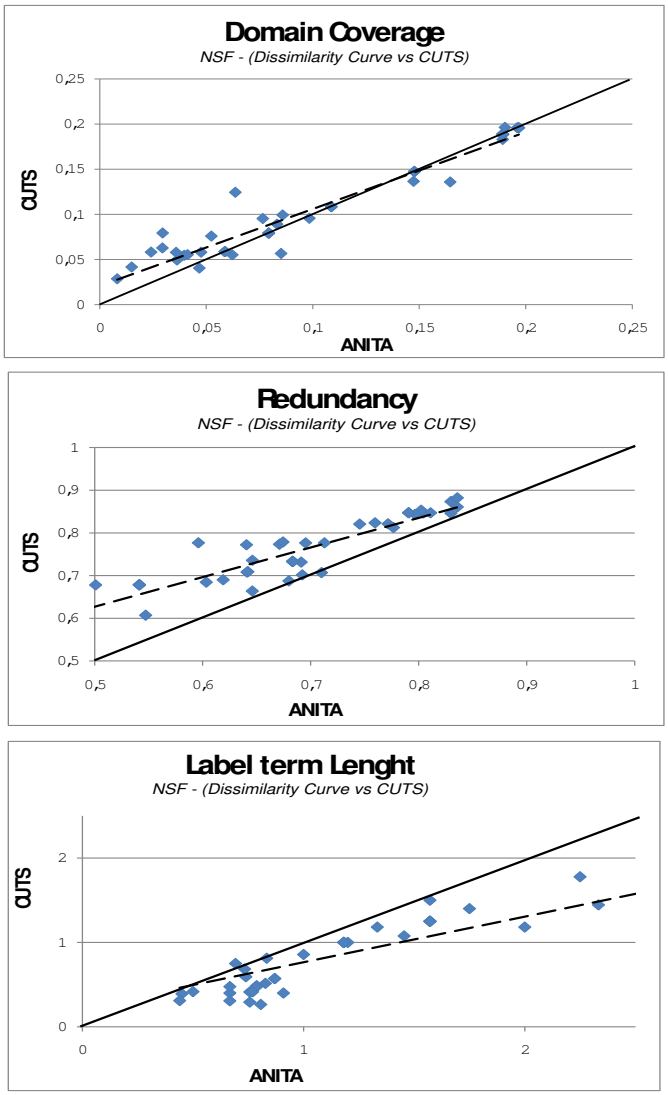


Figure 4.11: (a) Domain coverage, redundancy, and label term-length for `ANITA` and `CUTS` algorithm with NSF data set.

In these experiments, we considered target taxonomy size between 10% and 70% (with 10+ increments). The results in Figure 4.12 simply reports the percentages of cases in which one approach provides better performances than the other; as it is possible to notice, `ANITA` provides a clear gain in terms of lowering the amount of redundancy in the taxonomy (in 95.2% of the cases

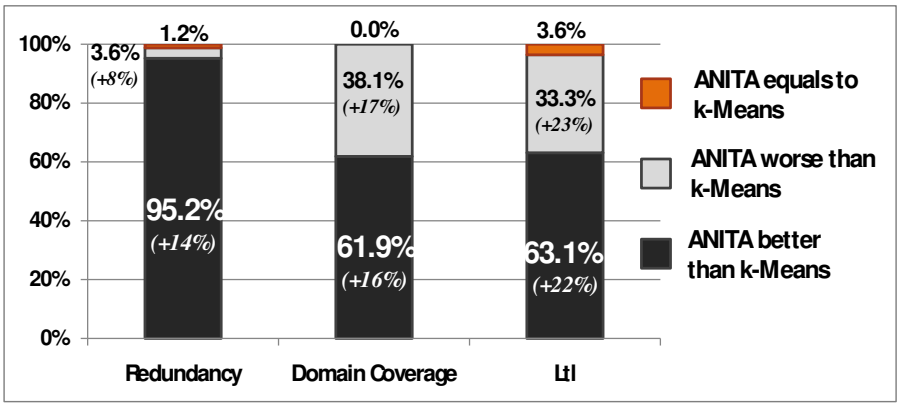


Figure 4.12: Comparison in terms Domain coverage, redundancy, label term-length between ANITA and k-Means (both data sets). The values in parentheses are the average gains by the winning scheme.

ANITA provides lower redundancy, with an average gain of $\sim 14\%$).

Moreover, ANITA also provides a significant gain in terms of domain coverage (in 61.9% of the cases, with an average gain of $\sim$ 16%) and lower values in terms of label term-length (in 63.1% of the cases, with an average gain of $\sim 22\%$), highlighting the global benefits of using the proposed adaptation approach. Again, it is important to notice that, even when k-Means reports better performances, its relative gains wrt. ANITA are similar to the relative gains obtained when ANITA reports better results.

This is consistent with the key design goals of ANITA; i.e., creating compact meta-data hierarchies that provide high category differentiation (to support effective navigation), even when the domain coverage increases. In fact, if both clustering approaches define a set of partitions of concepts/categories that are similar one to each other, ANITA leverages the statistical context information to define an order between concepts/categories that allow the system to find, as k-Means, the most similar concepts, but with relation to the structural distance given from the knowledge expressed by the domain expert through the original meta-data hierarchy. The main idea is that, given a concept node, we explicitly search for the *local* redundancies in neighbourhood; in fact we believe that, when a domain expert defines a concept c_i, he generally introduces (by inserting children nodes) many details that can be redundant or even irrelevant in a context. Thus, by ordering the concept-sentences, ANITA reduces the total redundancy of the categorization by search first for local redundancy and, proportionally to

Context: NSF+NYTimes Corpora			
	cover. ratio	redund. ratio	Ltl ratio
ANITA/H-EM	1.140	0.866	0.939
ANITA/EM	1.072	0.688	0.966
ANITA/X-Means	1.089	0.675	0.959

Table 4.6: ANITA vs. Hierarchical-EM ($\frac{\texttt{ANITA}}{H-EM}$), EM ($\frac{\texttt{ANITA}}{EM}$) and X-Means ($\frac{\texttt{ANITA}}{X-Means}$).

the structural distance, extend the search to more distant nodes.

We also compared the proposed `ANITA` narrative-based clustering approach against other clustering algorithms, such as EM, X-Means, and Hierarchical-EM (Hierarchical-EM method applies EM clustering strategy to each sibling group).

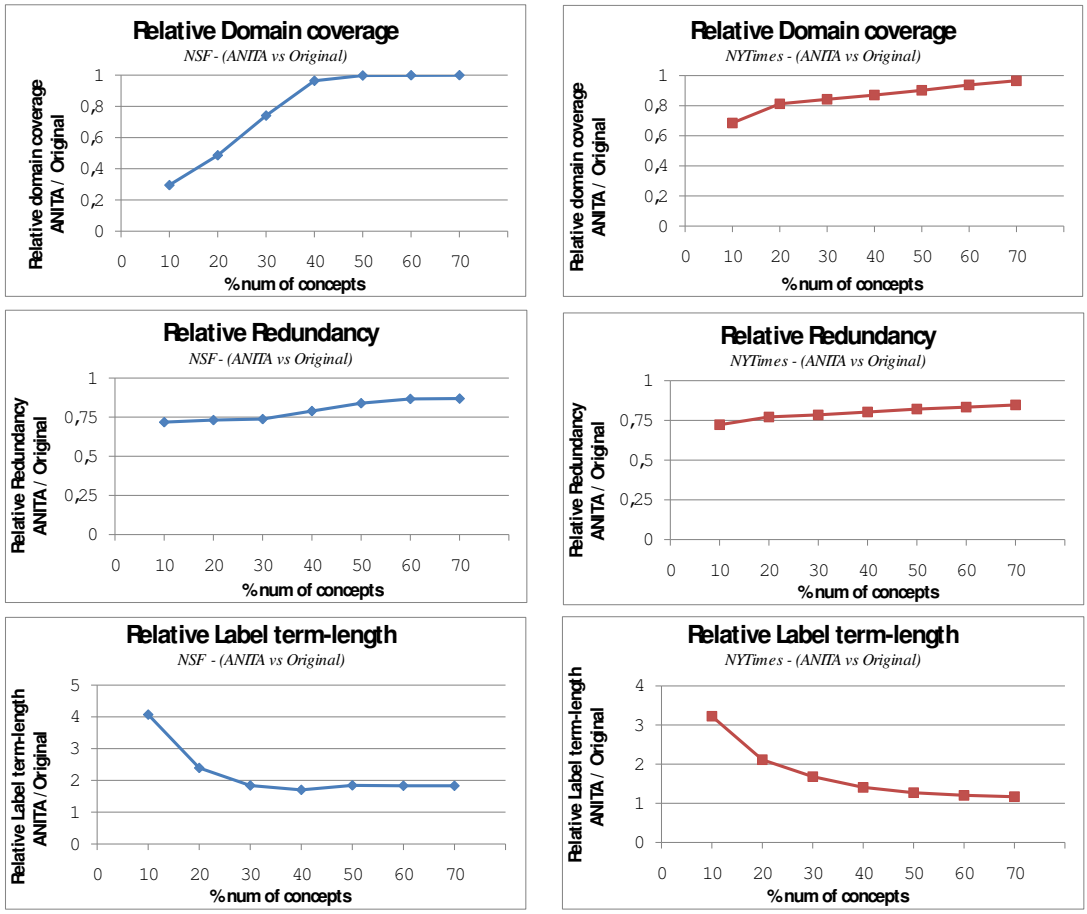
Since these algorithms do not take target number of clusters as input, we first apply these algorithms and then use `ANITA` with the number of clusters returned by them. As these results show (Table 4.6), `ANITA` provides better results in terms of all measures against these alternative clustering strategies; `ANITA` provides a clear gain in terms of lowering the amount of redundancy in the hierarchy in comparison to all the considered alternative approaches (up to 32% drop) as in terms of domain coverage (up to 14% increase) and provides lower values in terms of label term-length (a reduction up to 6%).

4.4.8 Comparison wrt. the Original Meta-Data Hierarchy

One of the key motivation of our meta-data adaptation method is that a properly adapted hierarchy can reduce the overall redundancy of the original meta-data and leads the user into a more effective exploration of the contents. Thus, in order to verify this hypothesis, in this Section we quantify how much difference in domain coverage and redundancy with respect to the original meta-data hierarchy occurs for varying target meta-data hierarchy sizes.

Figure 4.13 shows the ratios between the considered effectiveness measures on the adapted and the original hierarchies, refering to the NSF (blue lines in the Figures) and NY Times (red lines in the Figures) data sets.

Figure 4.13 shows that, for both data sets, the relative domain coverage is very close to 1.0 for adaptations with $\geq 30\%$ of the nodes; this means that the adapted hierarchies can index the same amount of contents as the original hierarchies. As expected, the coverage drops when the size of the adapted meta-data hierarchy is pushed further down, even though the label length increases to compensate for this drop. Note that, despite this



NSF Data Set NY Times Data Set

Figure 4.13: Domain coverage, redundancy, and label term-length ratio ($\frac{\texttt{ANITA}}{Original}$) curves using NSF data set and NY Times data set.

increase in the label lengths, ANITA is still able to lower the redundancy in the hierarchy, even when the compression rates are lowered down to 10% range. Finally, note that the similarities between the NSF and NY Times redundancy and label term-length curves on these charts highlight that the performance of ANITA in redundancy and label term-length is largely independent of the data set.

One major difference among the two data sets is the coverage behavior: in the case of the NSF data set, the original meta-data hierarchy appears to have many unnecessary nodes (i.e., many nodes have very few documents

associated in the considered corpus); thus, the relative domain coverage stays unaffected even when the target meta-data hierarchy has only 40% of the original nodes; after this point, there is a sharp drop implying that most documents are represented by only few nodes in the original hierarchy. In contrast, in the NY Times data set, the drop in coverage is slight, but more or less constant indicating that (a) most of the hierarchy nodes are significantly represented in the data set, but (b) the documents have more geographical meta-data nodes under which they can be classified.

4.4.9 Execution Time

For all the experiments we used an Intel Core 2CPU @2,16GHz with 1GHz Ram. The execution time is dominated by the narrative interpretation of the concept categories, obtained through an initial text processing and concept analysis (Section 4.2.1), which for these experiments was around 60 seconds for the scientifical input meta-data hierarchy of 72 nodes and 50K NSF articles (and around 140 seconds for the geographical meta-data hierarchy of 181 nodes and 64K NSF articles). The adaptation process itself takes less than 0.1 seconds (for both `ANITA` and k-Means).

Note that since the text processing is an off-line and one time process, the impact of the adapted meta-data hierarchy on the users' navigation times is a more critical factor than the execution time itself. We study this next through user studies.

4.4.10 User Study

In order to analyze the benefits of using the `ANITA` adapted categorization for text data indexing purposes, we also conducted a user study (similarly to [29]) and evaluate the feedback of 16 users when exploring NSF text articles using different meta-data hierarchies. The users represent various range of ages, backgrounds, jobs and education level and they have intermediate web ability (they are not computer scientists or domain experts).

We presented to the users, three different meta-data hierarchies that indexed NSF documents: the original portion of DMOZ-extracted hierarchy, with 72 concepts, its `ANITA`-based adaptation with 13 concepts (with k randomly set to 13) and the k-Means based adaptation (with same value of k). In order to avoid bias in the evaluation of the presented hierarchies, we presented the 3 meta-data hierarchies to the user in a random order.

Context: NSF Corpus		
	avg time (sec)	avg num. of interactions
Original (72 concepts)	23.5	5.1
ANITA (13 concepts)	9.7	2.3
k-Means (13 concepts)	11.0	2.9

Table 4.7: User Study: Average time and average number of interactions (clicks on the structure for expanding or collapsing nodes) per meta-data hierarchy, when the users explore the structure to retrieve documents related to a randomly selected concept.

Search Time and Interaction Counts

Given a randomly selected concept label extracted from the original hierarchy (different for each partecipating user), we asked the users, for each presented meta-data hierarchy, to retrieve related documents by exploring the presented categorizations. Therefore, we analyze the time and the number of interactions (in terms of expansions/collapses of the presented nodes in the hierarchy) the user needs to reach satisfactory documents. As reported in Table 4.7, `ANITA` adapted meta-data hierarchy reports significant gains in terms of time (from an average of 23.5 seconds to an average of 9.7) and number of interactions (from 5.1 to 2.3) by reducing the number of nodes the user has to navigate through. On the other hand it is important to note that, even if k-Means adapted meta-data hierarchy presents the same number of nodes as `ANITA`, it is not able to guide the user in an accurate exploration of the documents as well as `ANITA` adapted hierarchies do; in fact, with respect to the `ANITA` adapted hierarchy, the user needs more time to find relevant documents (an average of 11.0 seconds) and also more interactions to retrieve the appropriate contents (an average of 2.9 operations). Therefore, we can state that `ANITA` is not only able to reduce the cardinality of the selected meta-data hierarchy, but also organizes the concepts in such a way to facilitate the retrieval operations.

Classification Accuracy

Given a randomly selected article (different for each considered user), extracted from the considered NSF corpus of documents, we asked to the users, for each presented meta-data hierarchy, to select those nodes (if any) that would best represent the selected content. Then we compared these user associations with the ones automatically provided by the system (Section 4.4.1), calculating the percentage of shared concepts associated. All the considered users provided, for each document, between two and three

associated concepts per hierarchy. The results indicate that, for the original meta-data hierarchy, 67.7% of the user selected concepts were shared by the system. Similarly, the ANITA-based adapted hierarchy provides a 68.7% of shared concepts, indicating that the quality of the hierarchy is as good as original one despite containing much smaller number of concepts. On the other hand, the k-Means based adapted hierarchy does not perform well: only 37.4% of the user selected concepts had been effectively associated by the system to such nodes, highlighting the fact that a naive re-structuring process (such as k-Means) can cause, from a user point of view, a significant increase in terms of confusion and disorganization.

Subjective Questionnaire Measures

After the study, each user also completed a brief questionnaire which included two questions ("Is the meta-data hierarchy easy to use?" and "Is the meta-data hierarchy sufficiently detailed?"); the users could quantify the responses using a 5-point scale ratings.

As shown in Table 4.8, the users reported that the ANITA adapted hierarchy was as "easy to use" as the original one (both 4.1) while the k-Means adapted hierarchy was significantly harder to use (3.3). Moreover, even if the number of presented nodes was dropped almost 80%, the users commented that, in terms of providing "sufficient details" (i.e., the number of alternatives), ANITA adapted hierarchy provides a good range of details, close to the original one (3.6 vs 3.8). Therefore, we can infer that the user does not care about the pure number of presented alternatives, but only cares about those that she really needs. We can summarize these results as follows: as initially supposed, the original meta-data hierarchies, developed by domain experts for broad coverage of documents, provide unnecessary details that can be removed without causing significant loss in terms of contextual knowledge. On the other hand, a general adaptation method such as k-Means, could introduce confusion and disorientation; in fact, from a user point of view, the k-Means adapted meta-data hierarchy significantly reduces the "sufficiency" (only 2.6) and results in hierarchies that the users find harder to use (3.3 in terms of "easy to use").

Thus, in conclusion, the case study and the experiments, presented in this dissertation, show how this approach enables contextually-informed strengthening and weakening of semantic links between different concepts. The unique aspect of our approach is that it mines emerging topic correlations within the data, exploiting both statistical information coming from the document corpus and the structured knowledge represented by the input

Context: NSF Corpus		
	easy to use	sufficiently detailed
Original (72 concepts)	4.1	3.8
ANITA (13 concepts)	4.1	3.6
k-Means (13 concepts)	3.3	2.6

Table 4.8: Subjective questions in the user study: for each question, each user has quantified her opinion by a 5-point scale rating.

meta-data.

Chapter 5

Exploration of Text Documents using Adapted Meta-Data Hierarchies

In the previous chapter, we defined our narrative-based method to reduce a given meta-data structure in order to reflect the real distribution of a large set of text documents. In this chapter, as proposed in [25] and [26], we provide a new exploration mechanism that leverages the obtained adapted meta-data in order to improve the efficiency of the exploration process, using the natural relationships expressed by the given contents in addition to those formalized by the associated adapted meta-data structures.

5.1 Preliminary Motivation

Even if many popular approaches to text exploration are based on available feature statistics [136], many recent systems begin to leverage available semantics to guide the retrieval process towards an equilibrium between relatedness and wisdom [50].

We recognize that the assumption that users know what they want precisely is not always valid. Also, the conventional way of presenting the user a list of candidate documents may fail to help the user observe the contextual relationships, among general categories and documents, hidden in the data contents. Therefore, traditional feedback processes, which can be degraded significantly if the user's feedback is uninformed or inconsistent, may fail to be effective.

This problem can be addressed to a limited extent by relying on domain

meta-data structures that can inform the user about the current domain specific relationships among concepts/categories and, thus, support relatively more informed navigation/exploration within the document space [135]. However, the meta-data structures describe the given domain with categories and relationships which are valid at the time at which the meta-data was created. Thus, it might be convenient to first perfom the adaptation process (described by the previous Chapter) on the given hierarchical structures, in order to adapt them to the semantics expressed by the considered data.

In this chapter, as proposed in [25] and [26], we present our exploration system to help the users navigate within text collections, relying on a novel *keywords-by-concepts* (KbC) graph that leverages the optimized knowledge expressed by the adapted hierarchy. The KbC of a text collection, in the context of a given domain knowledge, is a weighted graph constructed by integrating a domain knowledge (formalized in terms of the adapted hierarchical meta-data structures, i.e., the semantic context) with the given corpus of text documents (i.e., the contents). Consequently, unlike related works, where the feature weights either reflect the keyword statistics in the corpus of text contents or the structural relationships between the concepts in the meta-data (see Section 2.2 for a discussion on literature related to this problem), the weights in the KbC graph reflect both the semantic context (imposed by the meta-data structure) and the documents' content (imposed by the available document corpus[1]).

Figure 5.1 shows a fragment of a sample KbC graph. This example leverages a previously adapted geographical domain knowledge (which organizes geographic entities of the World - cities, provinces, regions, states, continents) and the keywords extracted from a collection of newspapers articles. In this example, the newspaper articles from which the keywords are extracted are about the "*9/11 World Trade Center terrorist attack*" and the "*American invasion of Afghanistan*":

- Each node in the graph is either a concept, from the adapted hierarchical meta-data, or a keyword extracted from the content of the document base.

- The graph is bipartite: each edge connects a domain concept to a content keyword (hence the name *keywords-by-concepts* graph). The edges are weighted and they weight the strength of the relationship

[1]In the news application, that motivates this research, this corpus is defined by the temporal frame of interest and/or the keywords appearing in the news articles.

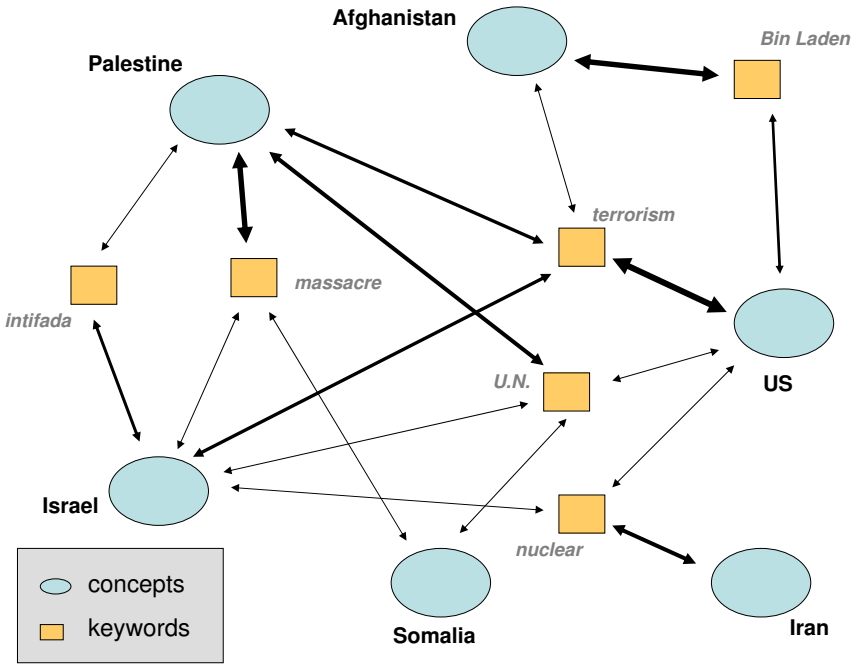


Figure 5.1: An example KbC graph constructed using concepts from a geographical hierarchical meta-data structure and keywords extracted from a corpus of news documents.

between the connected nodes in the given context. In Figure 5.1, the weights of the edges are visually represented through the thickness of the edges.

Consider the geographical concepts "*US*" and "*Afghanistan*". In the graph fragment, "*US*" is linked to the content keywords *"terrorism", "Bin Laden", "U.N."*, and *"nuclear"* (in decreasing order of weights), while "*Afghanistan*" is connected to *"terrorism"* and *"Bin Laden"*. Thus, these last two keywords create a content-based association between the two geographical concepts "*US*" and "*Afghanistan*". In fact, before the 9/11 events, very few people would immediately associate "*Afghanistan*" and "*US*". After the 9/11 events, however, keywords, such as "*terrorism*" and "*Bin Laden*" would strongly link "*US*" and "*Afghanistan*". Thus, domain-specific meta-data structures, when used alone, cannot be effective in capturing and leveraging the evolving semantics associated to the concepts. In particular, keywords associated to the same concept would strongly differ at different times

because the background contexts about the places, people, and the facts are different. Meta-data alone cannot capture this.

Thus, we propose to address these deficiencies of traditional purely feedback-based and purely meta-data based solutions, by developing an innovative exploration and navigation approach which discovers and highlights hidden, contextually-relevant relationships between concepts as well as keywords characterizing documents in the corpus. More specifically,

- we define the *keywords-by-concepts* (KbC) graph, which is a weighted graph constructed by a tight integration of the semantic context (e.g., the previously adapted meta-data) with the content (i.e., the keywords extracted from the documents search space) (Figure 5.1);
- we assign the weights of the edges in the KbC graph to reflect both the keyword statistics in the corpus of documents as well as the semantics and structural relationships between the concepts in the meta-data. Thus, we leverage these weights to associate a ranked list of documents to each node of the graph;
- we finally rely on the KbC graph in the CoSeNa (*Context-based Search and Navigation*) system for context-aware navigation and document retrieval.

Using CoSeNa the user can navigate within the document space by starting from any concept or keyword. In Figure 5.2 an example is shown; the user started the New York Times articles exploration experience by typing the keyword query "*peace*". CoSeNa presented the user many navigational alternatives as well as documents that are relevant, in the selected context, to the user request (listed at the right of the interface).

Navigational possibilities are represented relying on the tag cloud metaphor: the font sizes express the strength of the relationships among concepts and keywords. In the example, many navigational possibilities have been proposed; "*israel*" and "*middle east*" represent geographical concepts that are strongly related to the user query, while terms as "*barak*" or "*yasser arafat*" indentify corpus terms that are relevant to the user request.

Documents associated to the user query are enumerated in a ranked list. When the user clicks on a document, the system shows it and highlights the contextually important concepts and keywords in the document (by showing the most relevant terms, in the selected document, in different colors).

The user can navigate into the KbC space by clicking on the concepts and keywords highlighted in the tag clouds as well as in the documents. Considering the example proposed in Figure 5.2, clicking on the term "*yasser*

Figure 5.2: A sample screenshot of the Navigation interface of CoSeNa engine.

arafat", the user can enable a query refinment about the clicked term on the context defined by her query (the keyword "*peace*" in this case). The proposed system also provides integration with three on-line popular media sources: Google Maps, Flickr, and YouTube (at the left-bottom of the interface). To achieve context-based integration, CoSeNa queries the content sources leveraging the concepts and keywords in the clouds and presents the results to the user in a unified interface. By clicking on the proposed video/images, the user can visualize more in details the selected content (in a separate window).

5.2 Construction of the Keywords-by-Concepts Graph

In this Section, given a meta-data structure H and a corpus of documents (contents) D, we describe how to create the **keywords-by-concepts** (KbC) navigational graph to support the exploration of a large text collection, by highlighting the keyword and concept relationships. The construction algorithm combines information coming from a structural analysis of the relationships formalized in H with the analysis of the most frequent keywords appearing in the corpus D of documents. In the resulting graph, the weighted edges connecting keywords and concepts provide context-based navigation opportunities. In Section 5.4, we will show the use of the graph in assisting navigation and exploration within CoSeNa system.

The construction of the graph is preceded by a 4-step analysis process, which extracts, from the given meta-data and document corpus, the information needed to identify the concept-keyword mappings relevant to the given context:

1. The first step takes as input a meta-data structure and the set of considered text documents; then, it preliminarily adapts this structure to the considered corpus of documents. Therefore, it maps the new adapted concepts onto a concept-vector space in a way that encodes the *structural* relationships among nodes in the adapted hierarchy. The embedding from the concept hierarchy to the concept-vector space is achieved through the previously described concept propagation scheme which relies on the semantical relationships between concepts implied by the structure of the meta-data to annotate each concept node in the hierarchy with a concept-vector (Section 5.2.1).

2. For each concept in the adapted meta-data structure, the second step enriches the related concept-vectors by leveraging the knowledge expressed by the corpus of documents. This helps identify highly correlated concepts and keywords, providing the basis for the *keywords-by-concepts* (KbC) navigational graph construction.

3. Using these semantic correlations, we finally create the Keywords-by-Concepts navigational graph that tightly integrates keywords and concepts to provide a unique structure for an efficient exploration of the corpus.

4. Finally, we also extend the semantics of each considered keyword belonging to the KbC graph, in order to improve the efficacy of the graph

in guiding the user in the exploration process.

Next, we discuss these steps in detail.

5.2.1 Meta-Data Analysis: Adapting the Structure and Embedding Concepts into a Concept-Vector Space

Given a hierarchical meta-data structure $H(C, E)$, and a corpus of related text documents D, we perform the adaptation process described in Chapter 4 in order to obtain a new hierarchical meta-data $H'(C', E')$ that best describes the considered text documents. Then, in order to support discovery of mappings between concepts and documents, we need to map concepts in the adapted meta-data onto a concept-vector space.

Again, for this analysis step, we rely on the mapping process presented in Section 3.1.1. Given a meta-data structure, we assign a *concept-vector* $\vec{cv}$ to each concept node in the hierarchy, such that the vector encodes the structural relationships between this node and all the other nodes in the hierarchy.

5.2.2 Discovery of Concept-Keyword Mappings

The next step towards the KbC construction process is to discover the concept-keyword mappings using the concept-vectors identified in the previous step. In other words, in this phase, we find those keywords that strongly relate to the concepts in the taxonomy. Let $\vec{cv}_{c_i}$ denote the concept-vector corresponding to concept c_i. At this step, given a concept c_i and the related association, $A_{c \to d}(\vec{cv})$ containing the most related documents (calculated by considering the adaptive cut-off Algorithm described Section 3.2), we search for the most contextually informative keywords corresponding to this concept. More specifically, we compute the degree of matching between the given concept and a keyword which occurs in the associated documents by using the process described in Section 3.3. This process can be read as treating the concept-vector corresponding to the concept c_i as a query and the set of associated documents $A_{c \to d}(\vec{cv})$ as positive relevance feedback on the results of such query. For each concept, we consider all keywords contained in at least one document. We apply an adaptive cutoff (see Section 3.2 for the adaptive cut-off algorithm) to this set in order to select those keywords with the highest weights. At the end of this step, each concept c_i in the adapted meta-data hierarchy is associated to a so-called **extended-vector**, $\vec{ev}_{c_i}$, that tightly integrates keywords from the corpus and concepts from the hierarchy. Thus, these vectors not only quantify the structural relationships

among the hierarchy concepts, but also formalize the relationships with the chosen context documents (by including those keywords better describing the concepts); in fact, for each concept, these keywords are extracted from the related document association (Section 3.2) by using the the approach proposed in 3.3.

5.2.3 Constructing the KbC Graph to Support Document Retrieval

At this point, for each concept c_i, we have obtained an extended-vector,

$$\vec{ev}_{c_i} = \langle u_{i,1}, u_{i,2}, \ldots, \rangle,$$

where $u_{i,j}$ represents the degree of matching between the concept c_i and the j-th keyword which occurs in the associated documents. This extended-vector encodes the related keywords in the corpus and their weights. In order to construct the KbC graph, we link together the concepts and keywords using these relationships.

Let $C = \{c_1, \ldots, c_n\}$ be the set of concepts in the input taxonomy, H, and $K = \{k_1, \ldots, k_m\}$ be the set of all keywords appearing it at least one extended-vector. We construct KbC in the form of an undirected, node-labeled, edge-weighted graph, $G(V_C \cup V_K, E, l, \rho)$, as follows:

- Let V_C be a set of vertices, $V_C = \{v_{c_1}, \ldots, v_{c_n}\}$, where vertex $v_{c_i} \in V_C$ is labelled as "c_i"; i.e., $l(v_{c_i})$ ="c_i";
- Let V_K be a set of vertices, $V_K = \{v_{k_1}, \ldots, v_{k_m}\}$, where vertex $v_{k_j} \in V_K$ is labeled as "k_j"; i.e., $l(v_{k_j})$ ="k_j"; and
- For all $v_{c_i} \in V_C$ and $v_{k_j} \in V_K$ such that $\vec{ev}_i[j] \neq 0$, there exists an edge $\langle v_{c_i}, v_{k_j} \rangle \in E$ such that

$$\rho(\langle v_{c_i}, v_{k_j} \rangle) = \rho_{i,j} = \frac{\vec{ev}_{c_i}[j]}{\|\vec{ev}_{c_i}\|}$$

Therefore $\rho_{i,j}$ represents the relative weight of the keyword k_j in the corresponding vector $\vec{ev}_{c_i}$, i.e. the role of the keyword k_j in the context defined by the concept c_i.

5.2.4 Associating extended-vectors to the Keywords in the given Corpus

It is important to notice that the proposed graph leverages not only the concepts expressed by the meta-data structure, but also the keywords contained in the considered text documents to help the user explore the corpus. Thus, we also need to associate an extended-vector to each considered keyword belonging to the constructed graph. In order to perform this operation, we consider their concept neighbors in the corresponding KbC graph. By construction, each keyword node $v_{k_j} \in V_k$ in the KbC graph is connected to at least one concept node, $v_{c_i} \in V_C$. Thus, the extended-vector for $\vec{ev}_{k_j}$ is computed as

$$\vec{ev}_{k_j} = \sum_{c_i \in neighbor(v_{k_j})} \left(\frac{\rho_{i,j}}{\|\vec{ev}_{c_i}\|} \cdot \vec{ev}_{c_i} \right),$$

where $\rho_{i,j}$ is the strength of the relationship between concept c_i and keyword k_j obtained through adapted meta-data and corpus analysis in Section 3.3. As it is the case for the $\vec{ev}_{c_i}$ vectors, $\vec{ev}_{k_j}$ are also normalized to 1.

5.3 Unifying Concept And Keyword-Vector Spaces to Support Document Retrieval

In order to support exploration of the documents in the corpus, we need to associate, for each node of the KbC graph, a corresponding (ranked) list of documents. In order to do that, for each node of the KbC graph, we leverage the related extended-vectors previously calculated, and we simply calculate the cosine similarities against the document-vectors. The text documents that best match with it are collected and associated to the node (ranked based on the similarity value). Notice that, using the extended-vectors, we are able to associate to each node of the graph not only the documents that contain the label of the node, but also the documents containing all contextually relevant terms (i.e., concepts or keywords).

5.3.1 Associating Documents to KbC Nodes in the given Context

Since, at this point, each concept and keyword node in the KbC graph has its own extended-vector $\vec{ev}$, the documents in the given corpus can be associated under these nodes as in Section 5.2.2, but using $\vec{ev}$ vectors instead of $\vec{cv}$ vectors. In this manner, using the extended-vectors, our system is able

to associate to each concept and keyword, not only the documents that contain that concept or the keyword, but also the documents containing all contextually relevant terms.

5.3.2 Measuring Concept-Concept and Keyword-Keyword Similarities in the given Context

At this point, each concept and keyword node in the KbC graph has an associated extended-vector $\vec{ev}$, capturing both the taxonomical relationships between concepts and the context defined by the documents in the given corpus. Therefore, in addition to associating documents to KbC nodes, the similarities between concept and keywords in the given context (defined by the adapted meta-data and the document corpus) can be measured using the cosine similarities between these vectors. In the next Section, we will describe the use of these similarities in CoSeNa to support document exploration.

5.4 CoSeNa System and Use Case

In this Section, we present the COntext-based SEarch and NAvigation (**CoSeNa**) system [25, 26], which leverages the KbC model introduced before. With CoSeNa, the user can navigate through the nodes in the KbC graph (computed in a preliminary pre-processing phase), starting from any concept or keyword. At each step, CoSeNa presents the user navigational alternatives as well as documents that are relevant in the given context. Navigational alternatives are represented relying on the tag cloud metaphor: given a concept or keyword,

- the system identifies most related concepts and keywords (using the KbC graph and concept-concept/keyword-keyword similarities), and
- forms a concept cloud (consisting of related concepts) and a keyword cloud (consisting of related keywords).

Concept and keyword font sizes express the strength of the relationships among concepts and keywords. Documents associated to the concepts and keywords are enumerated in a list ordered with respect to the weights calculated as in Section 5.3.1. When the user clicks on a document, the system shows the corresponding document text and highlights the contextually important concepts and keywords in the document. The user can navigate into

Figure 5.3: CoSeNa search with geographical concept "*Iraq*".

the KbC space by clicking on the concepts and keywords highlighted in the tag clouds as well as in the documents.

5.4.1 Navigational Interface

Figures 5.3 and 5.4 show the use of the CoSeNa system in a scenario where a corpus of news documents (the New York Times articles collection, which contains 300,000 text entries with over 100,000 unique keywords[2]) is explored with the help of a geographical concept meta-data structure [3].
Figure 5.3 depicts the visual interface of the CoSeNa system after the user provides the concept "*Iraq*" to start exploration. Coherently to the KbC model, CoSeNa first identifies related content keywords (including "*Saddam*

[2]http://archive.ics.uci.edu/ml/datasets/Bag+of+Words. This data set has no class labels, and for copyright reasons no filenames or other document-level metadata.

[3]The hierarchical meta-data defines the context that drives the user in searching and navigating the documents. In this case we highlight geographical relationships. The use of a historical meta-data would instead make evident historical relationship among documents

Hussein", "*missile*", "*weapon*", "*Kuwait*", and "*persian gulf*") and presents these to the user in the form of a *keyword cloud*.

In addition, using the concept-to-concept similarities (described in Section 5.3.2), CoSeNa also creates and presents a related *concept cloud* consisting of geographical concepts "*Iran*", "*United States*", "*North Korea*", and "*Russia*". These geographical concepts in the concept cloud are also shown on a world map, with markers representing visual links. Note that the CoSeNa interface also shows related videos and images (searched on Youtube and Flickr by using the concept and term clouds) as well as documents that are associated to the concept "*Iraq*" as described in Section 5.3.1.

When the user clicks on the term, "*weapon*", in the keyword cloud, CoSeNa updates the tag clouds as well as media (text, images, and video) presented to the user accordingly. The result is shown in Figure 5.4. In this case, the concept cloud ("*Russia*", "*Iraq*", "*North Korea*", and "*United States*") represents geographical concepts neighboring the keyword "weapon" in the KbC graph (coherently with the previous case, geographical concepts are shown on the world map). The keyword cloud ("*missile*", "*security*", "*arsenal*", "*warhead*", etc.) is created using the keyword-to-keyword similarities, as described in Section 5.3.2. When the user clicks on a document, as also shown in Figure 5.4, CoSeNa displays the corresponding article and highlights relevant content and keyword cloud elements in the document.

5.4.2 Contextual Impact

As described above, CoSeNa relies on the extended-vectors ($\vec{ev}$) of the concepts and keywords to associate documents to the nodes of the KbC graph. The extended-vectors are also used in determining the strengths of the connections among concepts and among keywords.

As opposed to the concept-vectors ($\vec{cv}$), which capture only the hierarchical relationships between concepts, these extended-vectors capture, in addition to the semantic relationships between concepts in the adapted metadata, also the context defined by the documents in the given corpus. In order to observe the impact of this corpus context on the strength of the relationship between a given pair of concepts, c_i and c_j, we define the impact of the corpus context as the ratio

$$impact(c_i, c_j) = \frac{cos(\vec{ev}(c_i), \vec{ev}(c_j))}{cos(\vec{cv}(c_i), \vec{cv}(c_j))}.$$

Note that if $impact(c_i, c_j) \sim 1$, then it means that the corpus context

Figure 5.4: CoSeNa interface after the selection of keyword "*weapon*"; in the figure the document visualization interface of CoSeNa which highlights occurrences of the tag cloud terms in the document.

has no impact on the strength of the relationship between concepts, c_i and c_j. On the other hand, if $impact(c_i, c_j) \gg 1$, then the context defined by the corpus impacts one or both of the concepts in such a way that their relationship strengthens. In contrast, if $impact(c_i, c_j) \sim 0$, then the impact of the corpus on the concepts, c_i and c_j, is such that their relationship is weakened by the nature of the given set of documents (i.e., the concepts are strongly related to disjoint news events and, thus, the relationship between the concepts is weaker than it is in the given taxonomy).

Table 5.1 shows sample pairs of concepts with most positive, neutral, and most negative impact when using the entire news article corpus. As can be seen here, the content of the news articles significantly strengthen the relationships between concepts, "*Iraq*" and "*United States*", and concepts, "*Europe*" and "*Iran*". In contrast, the relationship between concept pairs, "*Tucson*" and "*London*", has been weakened to almost null. In fact, the keyword clouds corresponding to these two concepts show that, while the

Concept 1	Concept 2	Impact
Cuba	Florida	71.60 (Strengthened)
Europe	Iran	55.61 (Strengthened)
Iraq	United States	48.51 (Strengthened)
Afghanistan	United Stated	29.27 (Strengthened)
...	...	...
North America	United Stated	1.01 (No impact)
Las Vegas	Nevada	0.99 (No impact)
...	...	...
Madrid	Houston	~ 0 (Weakened)
London	Tucson	~ 0 (Weakened)

(a) Using all the available news articles

Table 5.1: The impact of the corpus context: relationships that are strengthened and weakened using the context defined by the entire corpus of news articles.

Concept 1	Concept 2	Impact
United States	China	68.28 (Strengthened)
United States	Japan	43.12 (Strengthened)
United States	Taiwan	41.28 (Strengthened)
Europe	Russia	21.24 (Strengthened)
...	...	...
South America	Brazil	1.01 (No impact)
North America	Canada	0.99 (No impact)
...	...	...
New York	Harare	~ 0 (Weakened)
Paris	Sydney	~ 0 (Weakened)

(b) Using the "*economy*" articles

Table 5.2: The impact of the corpus context: relationships that are strengthened and weakened using the context defined by the news articles containing the term "economy".

former is related to immigration news (with keywords such as "*border patrol*" and "*u.s. border*"), the latter is highly related to sports and arts news (with keywords, such as "*Hamilton*" –the name of a British Formula1 driver–, "*spectator*", "*art*", and "*theater*").

Table 5.2, on the other hand, shows sample pairs of concepts with most positive, neutral, and most negative impact when the set of documents used for extended-vector computation are limited to those containing the keyword "*economy*". As can be seen here, the content of the economy related news articles significantly strengthen the relationships between geographic concepts pairs, "*United States*"-"*China*", "U*nited States*"-"*Japan*","*United States*"-"*Taiwan*" and "*Europe*"-"*Russia*". It is important to note that, as expected, the sets of concept pairs that are most positively and most negatively impacted (i.e., strengthened and weakened) are different when the

user's focus is different.

5.4.3 Explaining the Relationships between Concepts in a given Context

Given a document corpus, CoSeNa can *explain* the relationships among the concepts in the associated meta-data in terms of keywords extracted from the corpus or explain the relationships among the keywords in terms of the concepts. In order to analyze these semantic relationships between a pair of concepts, we study the extended-vectors, $\vec{ev}$, associated to the nodes corresponding to these concepts in the KbC graph and search for those keyword dimensions (representing the keywords in the KbC graph) which enabled the relationship between these concepts.

More specifically given two taxonomy concepts c_i and c_j, we rank those keywords that occur in both extended-vectors, $\vec{ev}_{c_i}$ and $\vec{ev}_{c_j}$, in terms of their *contribution* to the relationship between the corresponding concepts. Since the similarity between two extended-vectors are computed based on cosine similarity, we measure the contribution of the keyword k to the concepts c_i and c_j as follows:

$$contribution(k) = \vec{ev}_{c_i}[k] \cdot \vec{ev}_{c_j}[k].$$

After we order the keywords based on their contribution to the relationship of the considered concepts, we select those keywords with the highest contribution.

Note that the process of selecting the concepts that explain the relationship between two keywords is similar: in this case, we analyze the extended-vectors of the keywords nodes in the KbC graph and compute and rank the *contribution*s of the concepts in the extended-vectors.

Note that since the extended-vectors reflect both taxonomical as well as corpus contexts selected by the user, this approach permits understanding the context-specific relationships between two taxonomical concepts or keywords.

Table 5.3 reports the most relevant terms that caused the strengthening of the relationships between the geographic concepts pairs reported in Table 5.1; as explained before, the listed keywords are shared by the extended-vectors of the pair of concepts, and contributed strongly to the creation of the semantic links in the KbC graph (Section 5.2.4). For example, considering the geographical concepts "*Florida*" and "*Cuba*", the strength of this semantic relationship is based on terms as "*Elian Gonzalez*" – a young

Concept 1	Concept 2	Keywords
Cuba	Florida	Elian Gonzalez, immigration, Naturalization Service, Miami
Europe	Iran	security, Middle East, intelligence, petroleum
Iraq	United States	weapon, defense, gulf war, inspection
Afghanistan	United Stated	taliban, terrorism, Bin Laden, Pakistan

Table 5.3: The most relevant terms, in the extended-vectors, that guide the strengthening of the relationships between concepts using the context defined by the entire corpus of news articles.

Concept 1	Concept 2	Keywords
United States	China	World Trade Organization, Clinton, globalization, cooperation
United States	Japan	technology, consumer, investor, sony
United States	Taiwan	independence, threat, negotiation, tension
Europe	Russia	agreement, energy, Ukraine, crisis

Table 5.4: The most relevant terms, in the extended-vectors, that guide the strengthening of the relationships between concepts using the context defined by the news articles containing the term "*economy*".

Cuban boy who was at the center of a controversy involving the governments of Cuba and the United States –, "*immigration*", "*Naturalization Service*" and "*Miami*" that help define the nature of the relationship.

On the other hand, when we focus on the KbC graph created based on the subset of documents related to "*economy*" (Table 5.4), it is possible to note that the terms that caused the strengthening of the relationships among geographical concepts are strictly related to the economic domain. For example, the relationships between "*United States*" and "*China*" has been strengthened (Table 5.2) based on terms as "*World Trade Organization*" or "*globalization*" that represent highly focussed keywords in the considered domain. Therefore, the KbC graph reflects the context in which it is created and can be used for explaining the relationships between the concepts of interest within the given context.

5.4.4 Identifying Dominant Concepts and Keywords in a Given Context

As we mentioned earlier, CoSeNa can leverage the KbC graph for identifying dominant concepts and keywords in a given context. For this, CoSeNa relies on a random-walk based technique that mimics the behavior of a sentient being that navigates over the KbC graph in a way that reflects the strengths of the links. The key observation, also used in web link analysis [110] and social network analysis [117], is that if this navigation process continues

indefinitely, the sentient being will spend most of its time on (concept and keyword) nodes that are strongly linked to the rest of the graph. Therefore, if one can measure the portion of the time the sentient being spends during its infinite random walk on the KbC on a given node, then this can be used to measure the dominance score of the corresponding concept or the keyword.

In CoSeNa, we are relying on a PageRank [110] like algorithm to compute the dominance scores. More specifically, the authority of a term (a keyword or concept in our KbC graph) depends on the number and the authority of its connected nodes. Hence, given a term $t_i \in (V_C \cup V_K)$, its *dominance* is defined based on the authorities of its neighbors as follows:

$$dom(t_i) = d \times \sum_{t_j \in in(t_i)} \frac{dom(t_j)}{|out(t_j)|} + (1 - d)$$

where $d \in (0, 1)$ is a dumping factor which represents the probability with which the sentient being will simply navigate from one node in the KbC graph to another and $(1 - d)$ represents the probability with which it will jump on an arbitrary node in the graph (d is often set to 0.85 [110]), $out(t)$ is a function that returns the set of terms that have an incoming edge from t and $in(t)$ returns the set of terms in the KbC graph that have an outoing edge pointing to t. Since the definition is recursive, in practice, the dominance values are calculated using an iterative algorithm, where, at the initial instant, each dominance value is initialized to:

$$dom^0(t_i) = \frac{1}{|V_C \cup V_K|}$$

Then, at each step, s, the algorithm recomputes the dominance values based on the dominance scores of the previous step::

$$dom^s(t_i) = d \times \sum_{t_j \in in(t_i)} \frac{dom^{s-1}(t_j)}{|out(t_j)|} + (1 - d)$$

In Tables 5.5(a) and (b) the most dominant keywords and concepts based on the full document set are reported. When the full data set is considered, most of dominant terms coming from the considered corpus (Table 5.5(a)) are political (names of american politicians or general terms semantically related to politics) and technology related. Similarly, the most dominant geographical concepts (Table 5.5(b)) represent the countries (such as "*Israel*", "*Lebanon*", "*China*", "*Russia*", "*Cuba*", "*Colombia*", and "*Iraq*")

Ranking	Keyword Node	Dominance value
1st	Al Gore	0.0032
2nd	George Bush	0.0028
3th	John McCain	0.0015
4th	government	0.0013
5th	computer	0.0012
6th	internet	0.0011
7th	voter	0.0010
8th	democrat	9.7403E-4
9th	woman	9.1351E-4
10th	technology	7.6794E-4

(a) Node of KbC graph from the corpus (using all the available articles)

Ranking	Concept Node	Dominance value
1st	Israel	5.6029E-4
2nd	Miami	4.4354E-4
3th	Lebanon	3.6508E-4
4th	United States	3.4079E-4
5th	China	3.1240E-4
6th	Russia	2.5890E-4
7th	Cuba	2.2968E-4
8th	New York	1.4158E-4
9th	Colombia	9.7460E-5
10th	Iraq	9.2171E-5

(b) Node of KbC graph from the taxonomy (using all the available articles)

Table 5.5: (a) The most dominant keyword nodes on the KbC graph generated using the entire New York Times corpus and (b) the most dominant concept nodes on the KbC graph generated using the same document corpus.

Ranking	Keyword Node	Dominance value
1st	George Bush	0.0027
2nd	Al Gore	0.0026
3th	percent	0.0020
4th	government	0.0016
5th	White House	0.0013
6th	statesman	0.0012
7th	United States	0.0012
8th	business	9.3666E-4
9th	.com	7.7473E-4
10th	economy	7.7328E-4

(a) Node of KbC graph from the corpus (using all the "*economy*" articles)

Ranking	Concept Node	Dominance value
1st	United States	0.0012
2nd	China	6.6503E-4
3th	Israel	4.9638E-4
4th	Russia	3.4715E-4
5th	Mexico	2.6899E-4
6th	Japan	2.6759E-4
7th	New York	2.6169E-4
8th	Argentina	2.1334E-4
9th	Zimbabwe	1.6380E-4
10th	Taiwan	1.5477E-5

(b) Node of KbC graph from the taxonomy (using all the "*economy*" articles)

Table 5.6: (a) The most dominant keyword nodes (on the KbC graph generated using the "*economy*" related subset of the New York Times corpus) and (b) the most dominant geographical concept nodes on the KbC graph generated using the same document subset.

and cities (such as "*Miami*" –which is related to "*Cuba*"– and "*New York*") related to foreign political and economic relationships of the United States in the period covered by the selected corpus.

In contrast, in Tables 5.6(a) and (b) the most dominant keywords and concepts based on the subset of corpus containing the term "*economy*" are reported. As can be seen in these tables, when only the economy related subset is considered, the most dominant keywords and the most dominant concepts reflect the considered corpus context and highlight the economic

focus of this subset of news articles.

In conclusion, we proposed a novel *keywords-by-concepts* (KbC) graph, which is a weighted graph constructed by a tight integration of adapted domain meta-data (considered as the semantic context) with the keywords extracted from the available corpus of documents. KbC graph is then leveraged for developing a novel *a Context-based Search and Navigation* (CoSeNa) system for context-aware navigation and document retrieval.

The unique aspect of our approach is that it mines emerging topic correlations within the data, exploiting both statistical information coming from the document corpus and the structured knowledge represented by the input taxonomy. The case study shows how this approach enables contextually-informed strengthening and weakening of semantic links between different concepts.

Chapter 6

Adaptation of Hierarchical Meta-Data for Data Table Management

In Chapter 4, we described ANITA [24], our method to adapt a hierarchical meta-data structure to a large set of text documents; in Chapter 5, we presented our Cosena system [25, 26] that leverages this optimized structure to guide the user for an efficient exploration of the corpus. But considering the many different possibilities of the data storage, it can be very limiting to examinate only text corpora cases. We argue that the efficiency or the goodness of the adaptation approach is not affected by the format of the data corpus used during the process. To support our claim, in this Chapter, we will focus on a different, widely used data format, and we will consider relational tables. We will analyze different techniques to extract and analyze the data (as done for text entries associated to categorization nodes) in order to perform adaptations of the related hierarachical meta-data structures [20]. In fact, considering data table formats with millions of entries and dozens of different attributes, we need to optimize our approach in order to handle those large amount of data and provide, as done for the text corpora, novel mechanisms to explore them (by significantly reducing the number of entries).

6.1 Preliminary Motivations

Considering the many scenarios where it is hard to display complete data sets, formed in many cases by millions of tuples and dozens of different

attributes, there is an emerging need for novel methods to explore these large amount of data.

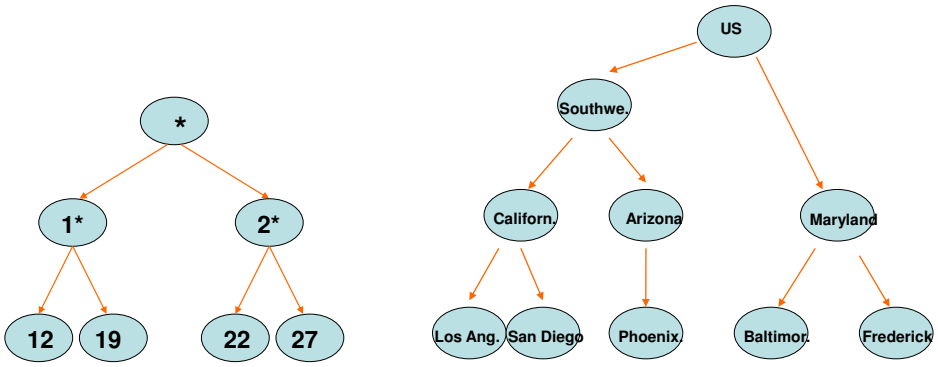
Consider, for example, a scientist exploring *the Digital Archaeologial Record (tDAR/FICSR)* [18, 120, 143], a digital library which archives and provides access to a large number of diverse data sets, collected by different researchers within the context of different projects and deposited to *tDAR* for sharing. When this scientist poses a search query through the system, her query might match many potentially relevant databases and data tables. For this scientist to be able to explore the multitude of candidate data resources as quickly and effectively as possible, data reduction techniques are needed.

Based on these considerations, one of the possible exploration approach relies on the idea of summarization: in fact, a summarization process takes as input a data table and returns a reduced version of it, permitting the user to analyze only few entries that represent the general trends. This abridged version of the data table needs to minimize the information loss (we will formalize in the following Sections this concept) due to the reduction in details; in particular each tuple in the original table needs to be represented, as in the summary, with a sufficiently similar tuple. Moreover, each tuple must be sufficiently different from other tuples to ensure that this summarized real-estate is not wasted.

Obviously, the result provides tuples with less precision than the original, but still informative of the content of the database; in fact, this reduced form can then be presented to the user for exploration or be used as input for advanced data mining processes.

Meta-data hierarchies have been commonly used for these purposes [28]. In fact, in [143], the authors have shown that meta-data hierarchies associated to the attributes of the tables can also be used to support table summarization. The table summarization process can benefit significantly from any prior knowledge about acceptable value clustering alternatives. When available, meta-data, such as value hierarchies (like the ones shown in Figure 6.1) associated to the attributes of the tables, can help greatly reduce the resulting information loss.

Consider, for example, Table 6.1 (a) which shows a data table consisting of 6 rows. If the user is interested in understanding the general trends expressed by this table (based on the attribute pair, $\langle Age, Location \rangle$), she may consider an abridged version where aggregation values are visualized in only 2 tuples; thus, using these aggregation values (extracted from the meta-data presented in Figure 6.1), it is possible to obtain a summary as in Table 6.1(b) that gives a general idea about the arguments expressed by the original table. In this case, the summary presents to the user the idea that



(a) Hierarchy for *Age* (b) Hierarchy for *Location*

Figure 6.1: Value hierarchy for attribute *Age* (a) and *Location* (b); directed edges denote the clustering/summarization direction (taken from [143]).

Name	Age	Location
John	12	Phoenix
Sharon	19	Los Angeles
Mary	19	San Diego
Peter	22	Baltimore
James	22	Frederick
Alice	27	Baltimore

(a) Data table

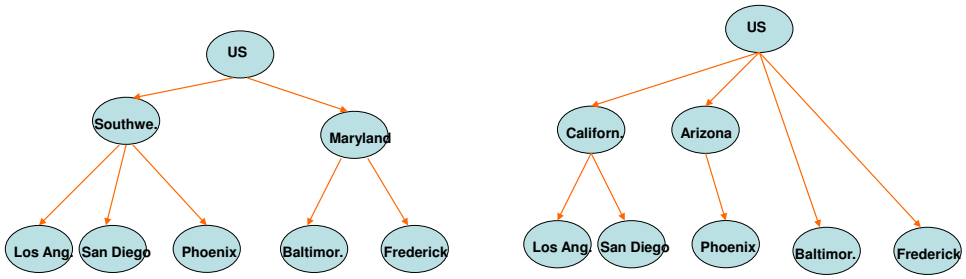
Name	**Age**	**Location**
-	1*	Southwest
-	2*	Maryland

(b) Summarized table

Table 6.1: (a) A database and (b) a summary on the $\langle Age, Location \rangle$ pair using hierarchies in Figure 6.1 (also taken from [143]).

two main entities are expressed in the original table; the first one represents people living in *Soutwest* (of United States) and the second one reports an entity about a community living in *Maryland*. Thus, considering this informative summary, the user can explore the data reported by the original table by analyzing human-understandable generalization values, obtained by the considered meta-data.

Based on these observations, we note that table summarization, whether carried out through data analysis performed on the table from scratch or supported through already available meta-data, is an expensive operation. For example, the computational cost of the meta-data supported table sum-



(a) Alternative #1 (b) Alternative #2

Figure 6.2: Two possible reductions of the location hierarchy in Figure 6.1(b).

marization process is exponential in the depth of the hierarchy (i.e., the number of alternative value clustering strategies) [143, 137].

Thus, considering this very important problem, the key observation driving this part of the thesis is that the speed of the summarization process can be significantly improved when the meta-data structures, used for supporting summarization, are pre-processed to reduce their unproductive details. The pre-processing of the meta-data to eliminate details not relevant for obtaining a table summary, however, needs to be performed carefully to ensure that it does not add significant amounts of additional loss to the table summarization process.

For example, while the reduced hierarchy in Figure 6.2(a) would still give the table summary in Table 6.1(b), the alternative in Figure 6.2(b) would cause further loss in the table summary. Thus, intuitively, in this context a *good* reduction of the given metadata is the one that leads to high-quality table summaries.

Then, based on these assumptions, we propose our *tRedux* algorithm for meta-data hierarchy pre-processing and reduction based on data table format [20]. Our hierarchy reduction process eliminates the details in the meta-data that are irrelevant for general exploration purposes.

More in details, our approach to meta-data hierarchy preprocessing and reduction consists of three steps:

Step I: create a graph representing the structural distances between the nodes in the given meta-data structure as well as the distribution of node labels in the database;

Step II: partition the resulting graph into disjoint sub-graphs based on connec-

tivity analysis; and

Step III: select a representative label for each partition and reconstruct a meta-data tree.

Before analyzing our meta-data hierarchy adaptation method, in Section 6.2 we introduce the concept of *value clustering meta-data* related to data table. Moreover, in Section 6.4 we formalize the table summarization problem and we introduce the *quality* measures for the table summaries. In addition, we will also explain how to apply hierarchical meta-data structures to the summarization problem.

6.2 Value Clustering Meta-Data

A value clustering meta-data is a tree $H(V, E)$ where V encodes values and clustering-identifiers (e.g., high-level concepts or cluster labels, such as "1*" in Figure 6.1(a), and E contains acceptable value clustering relationships. A value clustering hierarchy, H, is a tree $H(V, E)$:

- $v = (id : value) \in V$ where $v.id$ is the node id in the tree and $v.value$ is either a value in the database or a value clustering encoded by the node.
- $e = v_i \rightarrow v_j \in E$ is a directed edge denoting that the value encoded by the node v_j can be clustered under the value encoded by the node v_i.

Those nodes in V which correspond to the attribute values that are originally in the database do not have any outgoing edges in E; i.e., they form the leaves of the meta-data hierarchy.

Given an attribute value in the data table T and a value hierarchy corresponding to that attribute, thus, we can define alternative clusterings as paths on the corresponding hierarchy. In fact, given a value clustering hierarchy H, a meta-data node v_i is a clustering of a meta-data node v_j, denoted by

$$v_j \preceq v_i, if \exists path\ p = v_i \rightsquigarrow v_j \in H.$$

We also say that v_i covers v_j.

6.3 Tuple-Clustering and Table Summary

Let us consider a data table, T, and a set, SA, of attributes. Roughly speaking, our purpose is to find another relation T' which clusters the values in T such that T' summarizes T with respect to the considered attributes. Based on the above, in the following, we formalize the concept of tuple summarization.

Let t be a tuple on attributes $SA = \{Q_1, \cdots, Q_q\}$; t' is said to be a clustering of the tuple, t, (on attributes SA) iff $\forall i \in [1, q]$

- $t'[Q_i] = t[Q_i]$, or
- $\exists path\ p_i = t'[Q_i] \rightsquigarrow t[Q_i]$ in the corresponding value hierarchy H_i.

In this work, we use $t \preceq t'$ as shorthand.

Given this definition of tuple-clustering, we can define the summary of a table as a one-to-one and onto mapping which clusters the tuples of the original table.

Given two data tables T and T', and the summarization-attribute set SA, T' is said to be a summary of T on attributes in SA ($T[SA] \preceq T'[SA]$ for short) iff there is a one-to-one and onto mapping, μ, from the tuples in $T[SA]$ to $T'[SA]$, such that

$$\forall t \in T[SA], t \preceq \mu(t)$$

Here, $T[SA]$ and $T'[SA]$ are projections of the data tables T and T' on summarization-attributes.

6.4 Table Summarization Process

The general idea of a summarization algorithm is to leverage the underlying redundancy (such as approximate functional dependencies and other patterns) in the data to identify value and tuple clustering strategies that represent the (almost) same information with a smaller number of data representatives.

As introduced before, there are various meta-data supported table summarization algorithms [143, 137, 83].

K-anonymization algorithms [137, 83], for example, obtain summaries of the input tables based on available value hierarchies and given summarization parameter, k. Alphasum [143] and various others extend summarization support with more general metadata structures.

Without loss of generality, in this work, we use Samarati's k-anonymization algorithm [137] as the back-end table summarizer. In [137], each unique tuple gets clustered with at least $k-1$ other similar tuples to ensure that no single tuple can be uniquely identified. Given a table T, the authors consider a subset,

$$A_i, A_{i+1}, ..., A_{i+n}$$

of the attributes as the *quasi-identifiers* (i.e., those attributes that can identify the tuples in the database). Thus, the summarization process ensures that each tuple is clustered with $k-1$ others on their quasi identifier attributes.

The algorithm uses attribute value hierarchies to ensure that the amount of loss (i.e., value generalizations using the value hierarchies) is minimized. For each attribute, the algorithm takes a *value clustering hierarchy* which describes the generalization/specialization relationship between the possible values.

For example, consider a table with a "*location*" attribute. The meta-data hierarchy represents all the relevant values in the corresponding domain as the leaves of a tree (Figure 6.1(b)). The internal nodes in the value hierarchy will correspond to appropriate (geographic or political) clusterings of countries.

Thus, in a summary, an internal node of the hierarchy can be used to cluster all the leaves below it using a more general label. If in the summary a leaf value is used, this gives zero generalization ($g = 0$); if, on the other hand, a leaf at depth d is replaced with an internal node at depth d', this causes $g = d - d'$ steps of generalization; of course, by picking clusters closer to the root, the algorithm will be able to summarize more easily. On the other hand, more general cluster labels also cause higher degree of knowledge relaxation. In the next Sections we will refer to the knowledge relaxation due to the use of generalizing clusters as *dilution*.

Among all possible clusterings that put each tuple with $k-1$ other similar ones, [137] aims to find those that require *minimal generalizations*; i.e., the amount of distortion in the data needed to achieve the clustering is as small as possible. Intuitively, if there is a generalization at depth d that puts all tuples into clusters of size k, then there will be generalizations of level $d' \leq d$ that also cluster all tuples into clusters of size at least k, but will have more loss; conversely, if one can establish that there is no generalization at level d that is a k-clustering, then it follows that there are no other clustering of level $d' > d$ that can cluster all tuples into clusters of size at least k. Relying

on the fact that for a given attribute, applicable domain generalizations are in total order and that each generalization step in this total order has the same cost, [137] develops a binary search scheme to achieve savings in time[1]. It starts evaluating generalization levels from the middle-level to see if there is a corresponding k-clustering solution:

- if there is, then the algorithm tries to find another solution with less generalization by jumping to the central point of the half path with lower generalization;
- if there is none, on the other hand, the algorithm tries to find a solution by jumping to the central point of the half path with higher generalization.

The process continues in this binary search until a generalization level such that no solution with a lower generalization exists is found.

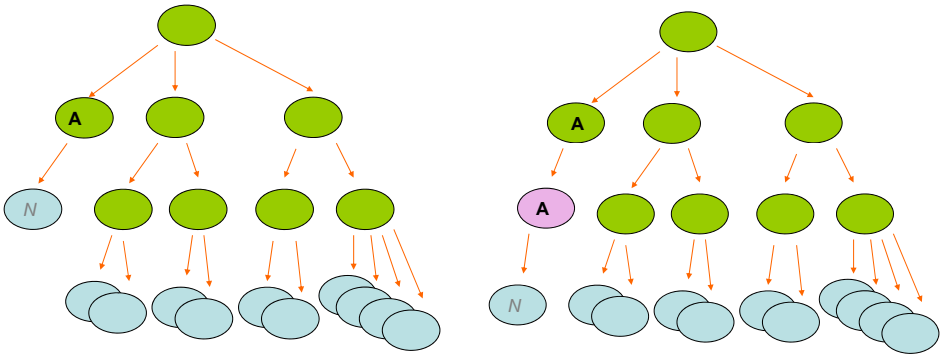
Unfortunately, this and other meta-data generalization based algorithms, including [83] and [8], are all exponential in the number of attributes that need summarization[2]. When there is a single attribute to summarize, for a hierarchy, H, of depth, $depth(H)$, the algorithm considers $log(depth(H))$ alternative clustering strategies. When there are m attributes to consider, however, the maximum degree of generalization is $\sum_{1\leq i\leq m} depth(H_i)$, where all attributes are generalized to the max, causing the greatest amount of loss. In this case, the algorithm considers $log(\sum_{1\leq i\leq m} depth(H_i))$ alternative generalization levels on the average; moreover, for each generalization level, g, the algorithm has to consider all combinations of attribute generalizations such that $(\sum_{1\leq i\leq m} g_i) = g$. Since in general, there can be exponentially many such combinations, the worst case time cost of the algorithm is exponential in the number of attributes.

One problem with using Samarati's algorithm [137] as the back-end table summarizer is that it assumes balanced meta-data hierarchies as input. In order to perform table summarization using unbalanced input meta-data hierarchies, we first balance the input hierarchies by introducing *ghost* nodes that fill the empty spots in the hierarchy.

As shown in Figure 6.3, these ghost nodes act as surrogates of the closest ancestors of the leaf nodes that are out of balance: in this example, in order

[1] [83] relies on the same observation to develop an algorithm which achieves attribute-based k-anonymization one attribute at a time, while pruning unproductive attribute generalization strategies. [8] further assumes an attribute order and attribute-value order to develop a top-down framework with significant pruning opportunities.

[2] In fact the problem is NP-hard [143]



(a) (b)

Figure 6.3: *Ghost* nodes in unbalanced hierarchy.

to put N at the same level of the other leaves, we introduce a *ghost* node between N and its parent A (Figure 6.3(b)). The ghost node is also labeled A as the parent. Note that, whether N is generalized to its original parent or the new (ghost) node, it will be replaced in the summarized table with A, therefore, this transformation does not result in additional loss.

6.4.1 Quality of a Table Summary

Information-based measures of quality leverage statistical knowledge, for example the knowledge about data frequencies, to measure information loss. One advantage of the use of meta-data hierarchies for table reduction is that the degree of loss resulting from the summarization process, can be quantified and explicitly minimized using the available value hierarchies.

Unlike purely numeric information loss measures, such as mean squared error, and statistical measures, such as entropy, classification, discernibility, and certainty [8, 68, 89, 51], knowledge about value hierarchies provides a mechanism to judge the significance of the distortion within the given application domain [137, 17, 158].

For example, a commonly used technique for measuring the amount of loss during the summarization process is to count the number of generalization steps required to obtain the summary [137, 8, 83]: given a generalization hierarchy, each step followed to achieve the value clustering is considered one unit of loss.

The weights of the value generalization alternatives may also encode different *utilities* including statistical information loss measures [129, 8, 17] and structural information. In structure-based method [123], the distance

between two nodes in a hierarchy is measured as the sum of the *distance* weights of edges between them: the weight of an edge between a parent concept and a child is measured based on the structural clues available in the hierarchy, such as depth (the deeper the edge in the hierarchy, the less information loss associated to the edge; e.g., in Figure 6.1(a), *San Diego* is more related to *Los Angeles* than *California* is to *Arizona*) and local density of the concepts (the denser the hierarchy, the smaller the semantic distance between the concepts in the neighborhood).

Let t and t' be two tuples on attributes $SA = \{Q_1, \cdots, Q_q\}$, such that $t \preceq t'$. Then the cost of the corresponding clustering strategy is defined through a monotonic combination function, $\sum$, of the **penalty** of the clustering along each individual summarization-attribute:

$$\Delta(t \preceq t') = \sum_{1 \leq i \leq q} \Delta_i,$$

where

- $\Delta_i = 0$, if $t'[Q_i] = t[Q_i]$
- $\Delta_i = \Delta(t[Q_i], t'[Q_i])$ (i.e., the minimal number of edges that separates $t[Q_i]$ to $t'[Q_i]$ in the corresponding original hierarchy) otherwise.

Let us consider a data table T, and a set SA of summarization attributes. Let T' be a summary of T on attributes in SA (i.e., $T[SA] \preceq T'[SA]$). We use two quality measures to evaluate table summaries: dilution and diversity.

Definition 6.4.1 (Dilution (dl))

$$dl(T, T', SA) = \frac{1}{|T|} \sum_{t \in T} \Delta(t[SA], \mu(t[SA])).$$

The smaller the degree of dilution, the smaller is the amount of loss and the higher is the quality of the summary.

Definition 6.4.2 (Diversity (div))

$$div(T', SA) = \frac{2}{|T'|(|T'|-1)} \sum_{t_1, t_2 \in T' (t_1 \neq t_2)} \Delta(t_1[SA], t_2[SA]).$$

The greater the diversity, the higher is the quality of the summary.

Therefore, given a table T and the set of summarization attributes, SA, the goal of the summarization algorithm is to find a summary such that the degree of dilution is minimized, yet the diversity is maximized.

6.5 Meta-data hierarchy adaptation

As we explained in the introduction, the table summarization process can be prohibitively costly, especially when the number of relevant attributes is large. In this Section, we propose our *tRedux* algorithm for value hierarchy reduction[20]. As already explained, our meta-data hierarchy adaptation approach firstly creates a graph representing the structural distances between the nodes in the meta-data and then divides the resulting graph into disjoint sub-graphs based on connectivity analysis. Then, it selects a representative label for each partition and reconstructs a meta-data hierarchy. In the following Sections we explain all these steps in details.

6.5.1 Step I: Constructing the Node Structural-Similarity/Occurrence Graph

Naturally, the most effective way to ensure the quality of the hierarchical meta-data structure is to cluster those nodes whose labels would be judged to be similar by human users of the system. Simultaneously, the cluster should also represent the joint-distribution of the node labels in the database. Therefore, the first step of the process is to create a graph that represents both the structural similarities of the nodes and the label co-occurrence in the database.

More formally, let us consider a data table T and a set SA of attributes. Let $H_i(V_i, E_i)$ be a value hierarchy corresponding to the attribute $Q_i \in SA$. In Step I of the adaptation method, the algorithm constructs a complete **weighted directed graph**, $G_i(V_i, E_i', w)$, where the set of vertices, $V_i = \{v_1, \ldots, v_n\}$, corresponds to the concepts in the input hierarchy. The weights $(w : E \rightarrow R^+)$ associated to the edges in E represent both

- similarities between pairs of concepts in the taxonomy, and
- occurrences of data values in the database.

In structure-based methods, the similarity between two nodes in a taxonomy is generally measured by the distance between them [123] or the sum of the edge weights along the shortest path connecting the concepts [130]. Information-based methods leverage the available data corpus to extract additional information, such as frequency, for corpora-sensitive similarity evaluation. Again, as described in Section 3.1.1, we first associate a node vector to each node in the meta-data (the vector represents the relationship of this node with the rest of the nodes in the hierarchy) and, then, we compare the

vectors to quantify how structurally similar any pairs of nodes are. We use the cosine vector similarity measure to quantify the structural similarities among the taxonomy nodes; comparisons against other approaches on available human-generated benchmark data [49, 129] showed that this concept vectorization improves the correlation of the resulting similarity judgments to human common sense [76]. More specifically, given the data table T, for each value hierarchy, $H_i(V_i, E_i)$, we construct a complete directed graph, $G_i(V_i, E_i', w)$: for each pair v_a to v_b of nodes in the taxonomy H_i, the edge between the corresponding nodes in G_i has the following weight:

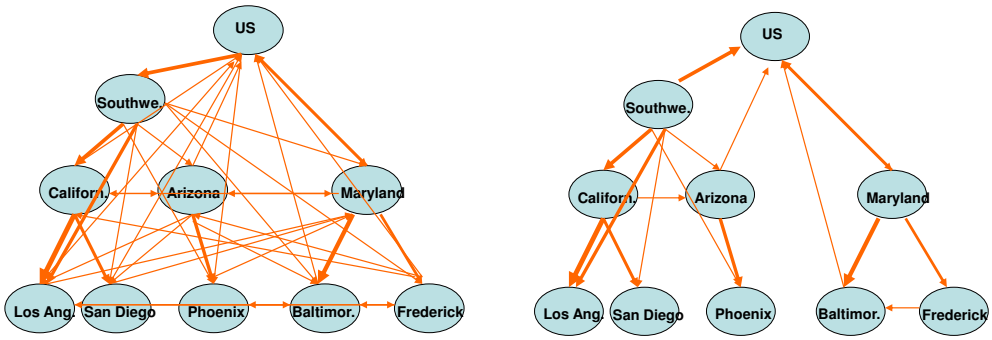
$$w(\langle v_a, v_b \rangle) = \sum_{t \in T} \vec{cv}_{v_a}[t.Q_i] \times \vec{cv}_{v_b}[t.Q_i],$$

where $t.Q_i$ is the value of tuple t for attribute Q_i and $\vec{cv}_x[y]$ gives the CP/CV value for node x along the vector dimension corresponding to the node y. Intuitively, the weight $w(\langle v_a, v_b \rangle)$ measures the aggregate similarity between the meta-data nodes v_a and v_b in the value hierarchy for all the values in the corresponding attribute in the database. Thus, the resulting graph $G_i(V_i, E_i', w)$ represents the structural relationships in $H_i(V_i, E_i)$ as well as the distribution of the data in the corresponding summarization attribute, Q_i, in the database: the weight of an edge is high if the concepts are structurally related in the value hierarchy, H_i, and the number of tuples in the corresponding attribute that are highly related to these concepts is also high.

Lastly, this graph $G_i(V_i, E_i', w)$ is thinned by applying a locally adaptive **edge thinning algorithm** [5, 114]. For each v_a in V, we consider the set, $out(v_a)$, of all outgoing edges:

1. we first sort the edges in $out(v_a)$ in decreasing order of weights;
2. next, we compute the *maximum drop* in consecutive weights; and identify the corresponding *max-drop* point in the sorted list of edges;
3. we, then, compute the *average drop* (between consecutive entities) for all those edges that are ranked before the identified *max-drop* drop point.
4. the first weight drop which is higher than the computed average drop is referred to as the *critical-drop*. All the edges in $out(v_a)$ beyond this *critical-drop* point are eliminated from E_i'.

This final thinning process ensures that only those edges that represent strongest relationships are maintained (note that, since the graph is directed



(a) (b)

Figure 6.4: Constructing the node graph based on the taxonomical structure of the hierarchy in Figure 6.1(a). The thickness of the edges represent their weights (omitted for clarity).

and the thinning process is asymmetrical, it is possible that the E_i' will contain a link from v_a to v_b, but not vice versa).

In Figure 6.4 an example, referring to the meta-data hierarchy previously presented in Figure 6.1(a), is shown; in Figure 6.4(a), the complete graph is created by connecting the nodes each other based on the distribution of the data (the weights of the edges are visually represented through the thickness of the edges). In Figure 6.4(b), the graph is finally thinned by only maintaining the edges with highest weights (that represent the strongest relationships).

6.5.2 Step II: Balanced Hierarchy Partitioning

In the next step, the resulting weighted graph $G_i(V_i, E_i', w)$ is partitioned based on its connectivity and the weights. In theory, any existing graph partitioning algorithm (e.g. [90, 106, 43, 56, 47, 73]) can be used in this stage. Many of these (including METIS [73], which we evaluate in the experiments Section), however, require advance knowledge about the number of clusters.

Thus, in practice, since the user will not be likely to have a target meta-data size, an adaptation algorithm which can partition the input graph based on its inherent structure, without requiring an input number of clusters, may be more suitable. Consequently, without loss of generality, we rely on a random walk-based graph partitioning algorithm [57] that does not require an advance knowledge of the number of resulting clusters.

A **random walk** on a graph, $G(V, E)$, is simply a Markov chain whose state at any time is described by a vertex of G and the transition probability is distributed equally among all outgoing edges. The transition probability distribution for a Markov model is often represented as an adjacency matrix. Here, we provide a brief overview of the clustering strategy we adopted. A stochastic process is said to be Markovian if the conditional probability distributions of the future states depend only on the present. A Markov chain is a discrete-time stochastic process which is conditionally independent of the past states. The components of the first singular vector [9] of the adjacency matrix of a random walk on a graph will give the portion of the time spent at each node after an infinite run. The singular vector corresponding to the second eigenvalue, on the other hand, is known to serve as a proximity measure for how long it takes for the walk to reach each vertex.

Considering this clustering approach, given the graph $G_i(V_i, E_i', w)$ constructed in the previous step we derive a random walk graph by associating the following transition probability to edges: let e be an edge from vertex v_a to vertex v_b; then, the corresponding probability of transition is

$$p_{a,b} = \frac{w(\langle v_a, v_b \rangle)}{\sum_{e_k \in out(v_a)} w(e_k)}.$$

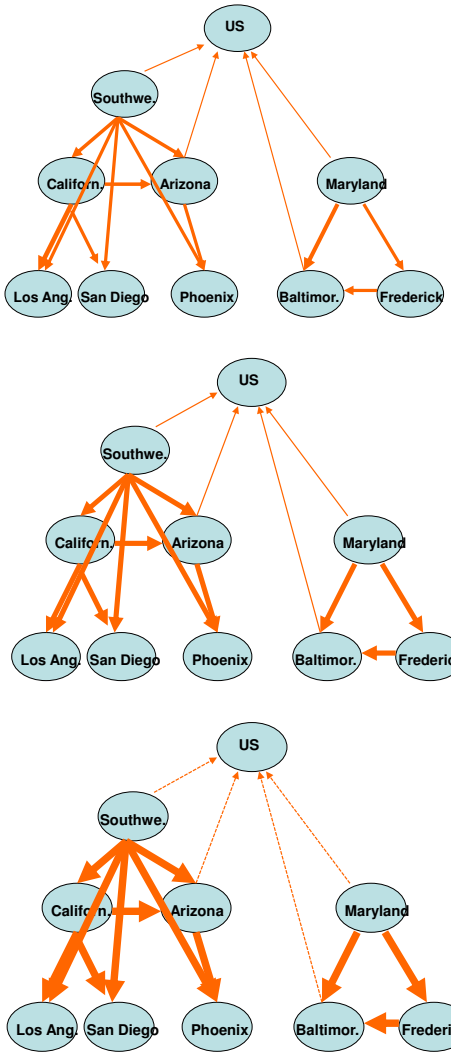
These probabilities are represented in the form of an adjacency matrix M_i.

Intuitively, two vertices in the same cluster should be quickly reachable from each other through a random walk. It is also possible to argue that if two nodes are in the same cluster, then the corresponding values in the second eigenvector must be close to each other.

Consequently, the random walks clustering approach proposes to use the values in the second eigenvector of the graph (which measure the proximity on the random walk) as the measure of being in the same cluster. In particular, the graph is partitioned by looking for significant jumps in the values of the second eigenvector of M_i. Thus, at each iteration of the algorithm, [57] lowers the transition probabilities of the edges that connect two verteces of different clusters (called *separetors*) and increases those connecting two verteces of the same cluster.

An example is shown in Figure 6.5; based on the thinned graph presented in Figure 6.4(b), the random walks clustering algorithm retrieved the edges that separate the clusters (called *separators* and identified in the Figure with dotted lines).

Existing random-walk based clustering algorithms, such as [57], consider



(a) 1rst iteration

(b) 2nd iteration

(c) 3rd iteration

Figure 6.5: Random walks clustering algorithm applied on the graph presented in Figure 6.4(b); the transition probabilities are visually represented by the thickness of the edges (the values are omitted in the Figure for clarity). At each run, the algorithm lowers such transition probabilities that identify cluster separators while it increases those that connect entities within the same cluster.

only the connectivity and the weights and do not seek to return partitions balanced in terms of the number of vertices. In other words, the number

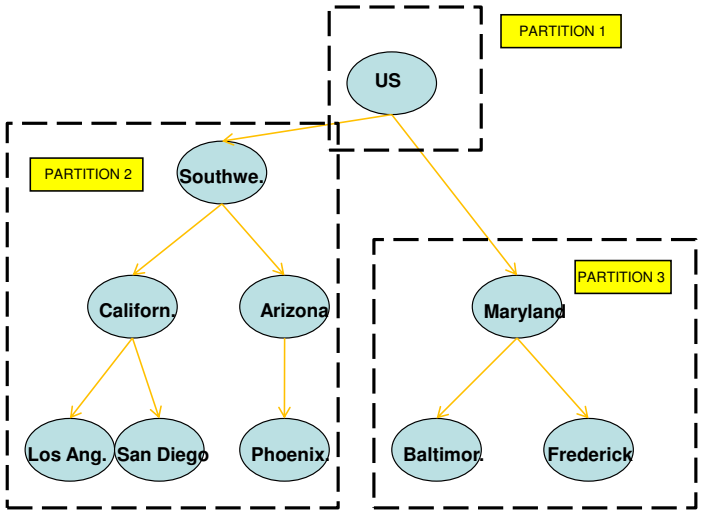


Figure 6.6: Node graph partition example: based on the separators retrieved by applying the random walks clustering approach [57] (Figure 6.5), 3 partitions have been found.

and the sizes of the clusters is strictly dependent on the connectivity of the considered graph. However, we may not want an adapted meta-data, where some summary nodes are precise (and represent only a few nodes in the original value hierarchy), whereas others are vague (and represent large numbers of nodes). As we have mentioned earlier, having concept clusters of widely varying sizes may be disadvantageous, as large concept clusters will be more vague than smaller concept clusters. An equally distributed set of clusters, on the other hand, would permit to generate a more informative and representative adaptation of the initial taxonomy, as each new entry in the reduced taxonomy represents (approximately) the same number of original nodes.

Therefore, we follow the initial partitioning approach with a **rebalancing step**. Let $H_i(V_i, E_i)$ be a hierarchical meta-data and $\mathcal{P}_i = \{P_{i,1}, \ldots, P_{i,m}\}$ be the set of partitions obtained through the random walk process.

In Figure 6.6 an example is shown; the meta-data hierarchy example proposed in Figure 6.1(a) is partitioned in 3 groups of nodes based on the random walk partition approach applied on the graph shown in Figure 6.4(b).

At this point, considering the m obtained partitions, in order to promote

balance in partitions, we introduce a *tolerance value*,

$$\tau = \theta \frac{|V_i|}{m}$$

that sets the maximum number of concepts that could be represented by any partition. If a cluster, $P_{i,j}$, contains too large a number of concepts, then a set, $\mathcal{X}_i$, of extra vertices are picked and moved to other partitions. This set of vertices are selected in such a way that the cost, $cost(\mathcal{X}_i)$, of displacement of the set of extra vertices among partitions is minimized.

The term $cost(\mathcal{X}_i)$ is

$$\sum_{v_a \in \mathcal{X}_i} \left(\sum_{e_j \in (edges(v_a) \cap \mathcal{P}_i)} w(e_j) - \sum_{e_j \in (edges(v_a) \cap dest(v_a))} w(e_j) \right),$$

where $edges(v_a)$ is the set of all incoming and outgoing edges to v_a and $dest(v_a)$ is the partition, other than P_i, with the highest weighted connectivity to v_a. The vertices in $\mathcal{X}_i$ and their destinations are selected through a K-means like iterative improvement process. Obviously, this method is strictly affected by the value of θ, that constrains the balancing effect of this partitiong approach. In the experiment Section we will deeply study the use of θ.

6.5.3 Step III: Meta-Data Hierarchy Re-construction

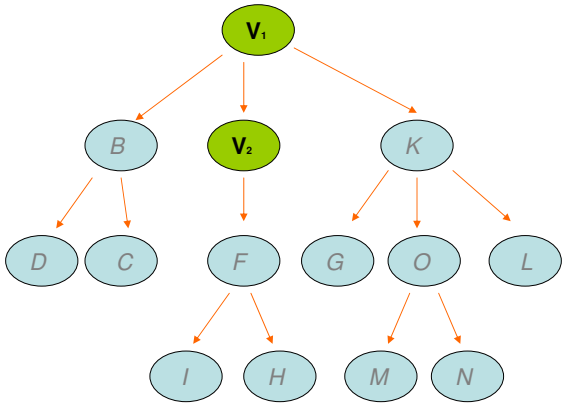
In order to construct the adapted meta-data hierarchy, we need to link the partitions, obtained in the previous step, in the form of a tree structure. Furthermore, for each partition, we need to pick a *label* describing the concepts in the partition.

Partition Labeling

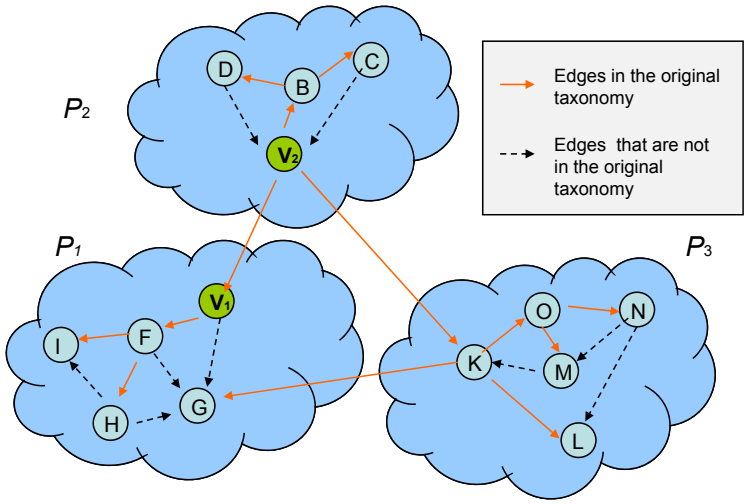
Considering a meta-data node, its label is important because it is what will be presented to the user in the exploration process. Thus, considering our partitions, we have to carefully select the appropriate labels in order to be sufficiently representative of the cluster.

To this purpose, we adopt a labelling method similar to the one we used in Section 4.2.3, in the context of the adaptation of domain meta-data hierarchies to enhance the exploration of textual databases.

Let $P_{i,j}$ be a partition in $\mathcal{P}_i$. In order to pick a label for $P_{i,j}$, we consider the relationships of the vertices in $P_{i,j}$ in the original hierarchy H_i. If there



(a) a sample taxonomy



(b) partitioning example

Figure 6.7: A sample taxonomy and its partitioning.

is a vertex, $v_a \in P_{i,j}$ that dominates all the other vertices in the partition (i.e., $\forall v_b \in P_{i,j} \;\; v_b \preceq v_a$), then v_a is selected as the label. If there is no such single vertex, then the minimal set, D_j, of vertices covering the partition $P_{i,j}$ (based on H_i) is found and the set, D_j, is used as the partition label.

Partition Linking

The reduced hierarchical meta-data H'_i should preserve the original structure of H_i as much as possible. Partitions linking follows the same ideas at the basis of the corresponding step (see Section 4.2.3) in the context of meta-data hierarchy adaptation for navigation within textual corpora:

- The root of H'_i is the partition $P_{i,j}$ which contains the root vertex of H_i.

- Let us consider a pair, $P_{i,j}$ and $P_{i,k}$, of partitions in $\mathcal{P}_i$. Let $E_{j,k}$ be the set of edges in H_i that go from the vertices in $P_{i,j}$ to vertices in $P_{i,k}$. Similarly, let $E_{k,j}$ be the set of edges in H_i that go from the vertices in $P_{i,k}$ to vertices in $P_{i,j}$.

  If in H'_i, $P_{i,j}$ is an ancestor of $P_{i,k}$, then the broken set of edges in $E_{k,j}$ will result in structural constraints that are violated. If $P_{i,k}$ is an ancestor of $P_{i,j}$, then broken edges in $E_{j,k}$ will result in structural constraints that are violated. If neither is an ancestor of the other, on the other hand, the edges in $E_{k,j} \cup E_{k,j}$ will determine the constraints that are violated.

  Let $e = \langle v_a, v_b \rangle$ be an edge from partition $P_{i,j}$ to $P_{i,k}$. If e is broken, then its cost ($cost(e)$) is the number of descendants of v_b in the original hierarchy H also contained in $P_{i,k}$ plus one (for v_b). For example, in Figure 6.7, if the edge between V_1 and K is broken, then the cost of this edge is equal to $1 + |\{O, N, L, M\}| = 5$.

  Thus, the taxonomy H'_i, minimizing the errors due to structural constraint violations can be constructed by

  1. creating a complete weighted directed graph, $G_P(V_P, E_P, w_P)$, of partitions, where
     - $V_P = \mathcal{P}_i$,
     - E_P is the set of edges between all pairs of partitions, and
     - $w_P(\langle P_{i,j}, P_{i,k} \rangle) = \sum_{e \in E_{k,j}} cost(e)$; and
  2. finding a *maximum spanning tree* of G_P rooted at the partition $P_{i,j}$ which contains the root of H_i.

At this point, the original meta-data has been partitioned and a reduced hierarchy has been reconstructed.

6.6 Case Study

In this Section, in order to further clarify the proposed meta-data hierarchy adaptation method, we analyze a real case study. We considered the real *Census Income* data set (also known as *Adult* data set), extracted from the 1994 Census database [4] (containing ∼30K tuples and including 16 attributes), and we analyze and adapt the associated geographical value hierarchy representing geographical aggregation values related to the *Native Country* attribute. As reported before, the considered meta-data hierarchy represents, as internal nodes, the possible aggregation values, and contains, as leaves, all the different data values. The original hierarchy, containing 67 nodes, is shown in Figure 6.8.

As we already discussed, we believe that any pre-determined value hierarchy is generally designed in order to widely describe the considered domain knowledge for general purpose applications; thus, many un-necessary details, corresponding to possible aggregation values, might be introducted in the hierarchy. Thus, our assumption is that, if there are superfluous data aggregation values, it is possibile to re-define and/or remove these nodes from the hierarchy, depending on the distribution of the values in the considered data table.

Therefore, considering our chosen case study domain, we apply the proposed value hierarchy adaptation algorithm to adapt the structure to the considered data (as show in Figure 6.9) and reduce the overall redundancy while maintaining the most relevant data.

In this example, the granularity of the considered hierarchy is now reduced from 67 concepts to 50 concetps. We observe that, as opposed to the text corpus case, the most visible reduction has been obtained at the *highest* levels of the hierarchy. In fact, in this case the leaves have to be preserved in the adapted hierarchy in order to be able to retrieve the original values represented in the data table. Thus, the algorithm performs its operations on the internal nodes, reducing, where it is possible, the redundant information and preserving those values that better represent the tuples. In fact, the algorithm analyzes the distribution of the data over the considered meta-data hierarchy (Section 6.5.1), and adapts the structure in order to represent with a higher number of nodes branches that are very dense (and therefore, need to be discriminated with a higher number of possibilities). In contrast, the less representative branches will be significantly reduced, due to the fact that they do not need a high number of aggregation values. For example, the node "*lands-middle-america*" has been removed in the adapted hierarchy, simply because the real data distribution on the original table does not justify its

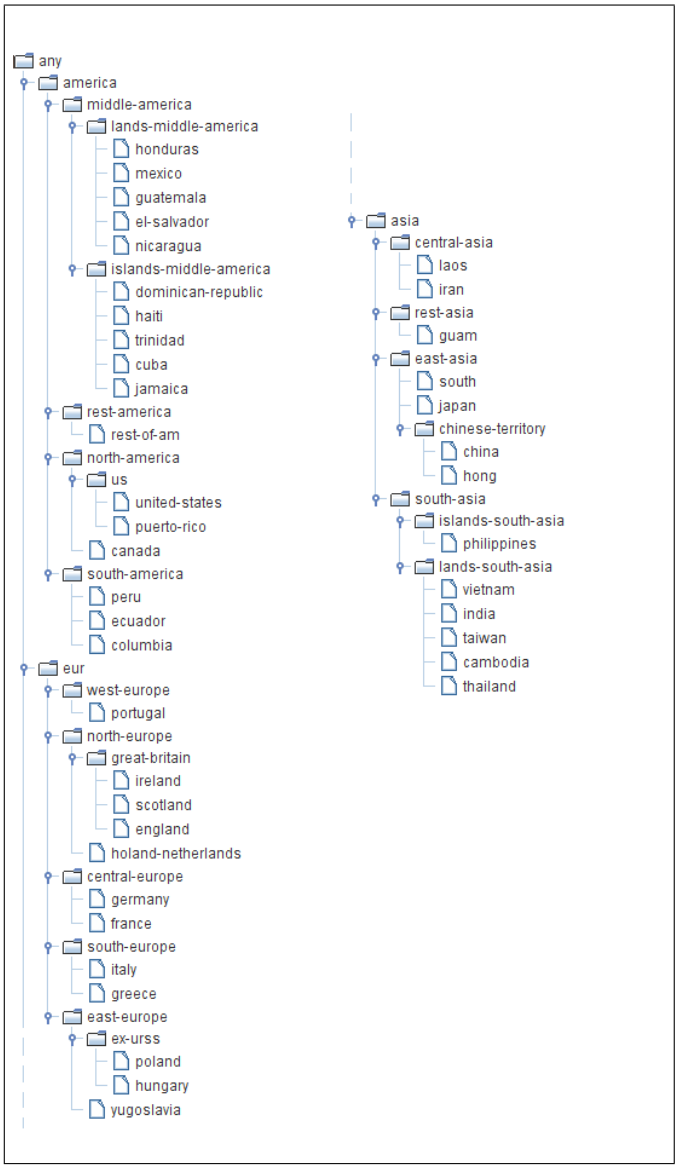


Figure 6.8: Geographical Value hierarchy containing the values (leaves in the structure) related to the attribute *Native Country* of the Adult Data Set [4].

presence; in fact, in case of data reduction processes, the algorithm reports that its children nodes ("*Honduras*", "*Mexico*", "*Guatemala*", etc.) could

be summarized with the ancestor "*middle-america*", without any significant loss of information with respect to summarizing them with "*lands-middle-america*". Indeed, these data values are very rare on the original table, and due to this characteristics, they do not need a specific aggregation value (or, at least, it could be sacrified for reduction purposes).

It is also important to notice that, in contrast with the text corpora case, the length of the node labels did not increase (i.e., labels have not been merged to other labels). The interpretation of this behaviour is correlated again to the data distribution. In fact, considering that the internal nodes of the hierarchy are not represented at all in the original data table, the proposed clustering strategy leverages at the maximum the meta-data structure to group the data value (leaves in the taxonomy). Therefore, each coherent group can be simply represented by its parent node (or a higher ancestor, if necessary) reducing the cases in which it is necessary to merge the labels to find a unique representative.

In conclusion, in this example, the value hierarchy has been reduced in terms of number of internal nodes and re-defined in terms of its concept relationships, in such a way to reflect the real data distribution. It is also very important to notice that the new adapted meta-data structure is still undestandable by human users and therefore usable for exploration purposes. In the experimental part (Section 6.7) we will deeply analyze this characteristic.

6.7 Experimental Evaluation

In this Section we consider the table input format, for evaluating our hierarchical meta-data adaptation method.

Meta-data supported table summarization needs three inputs:

1. a table T to summarize with q attributes;
2. a set of domain hierarchies D_i (for each attribute which we want to summarize), and
3. a parameter k that determines the minimum size of tuple clusters in the summary.

Thus, we experimented with different data sets, meta-data hierarchies, and k values. We evaluated our technique considering different types of data sets (and therefore different types of domain generalization hierarchies). In order to evaluate our method, we considered two different datasets:

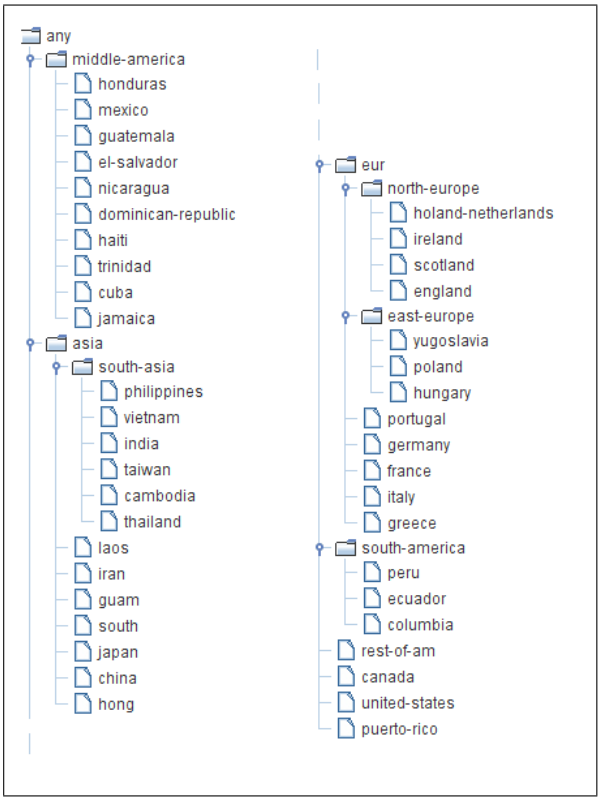


Figure 6.9: Adapted value hierarchy of the one proposed in Figure 6.8.

- Real dataset: we used the *Census Income* dataset (also known as *Adult* dataset), extracted from the 1994 Census database [4]. This data set contains ∼50K tuples and includes 14 attributes.
- Synthetic dataset: we constructed sub-sets of tuples with different properties in order to evaluate *tRedux* under different conditions (see Section 6.7.2).

For both data sets, we varied the number of tuples in the data set.

For all experiments described in Section 6.7.1, we considered 7 different subsets of tuples (from ~ 100 to ~ 800) for the Adult dataset and 6 different subsets (from ~ 100 to ~ 1000) for the synthetic dataset.

We also varied the *tuple count variance* (t-var), which is defined as the variance in the number of occurrences (in the input table) of the leaf values of the hierarchy; this value was varied between 0 (i.e., uniform distribution) and ~ 11 for Adult dataset and between 0 and ~ 15 for Synthetic dataset.

We also experimented with different numbers (1,2 and 3) of attributes in the summary. For each case, we considered different summarization requirements, varying k in the set $\{5, 10, 20, 30\}$.

In addition to the real and synthetic data, we also experimented with real and synthetic domain hierarchies. The synthetic domain hierarchies we used for the experiments also varied in structure (size and height). We provide more details about the variations in the domain hierarchy structures in Section 6.7.2.

Finally, we have also experimented with different partition balance tolerance values when creating the reduced taxonomies (see Section 6.5.3). We varied the tolerance value, θ in the set $\{1, 1.5, 2, 3, 4\}$ ($\theta < 1$ is not meaningful, $\theta = 1$ means balance, and $\theta > 1$ is increasingly lax in terms of balance requirement – as we will see in Section 6.7.2, diversity and dilution is more or less constant for $\theta \geq 2$, therefore this range is sufficient for observing the impact of θ). Unless explicitly stated, the default tolerance value, $\theta = 2$, is used.

For all the experiments we used an Intel Core 2CPU @2,16GHz with 1GHz Ram.

6.7.1 Loss in Diversity and Dilution due to Reduced Meta-data

Before we analyze the behavior of the *tRedux*-based table summarization under different system parameters, we first compare dilution and diversity behaviors of various alternative meta-data driven table summarization approaches. In particular, we compare the following alternative schemes:

- table summarization using the *original* hierarchies; in this scheme the input hierarchies are not reduced.
- table summarization using hierarchies reduced by applying *tRedux*.
- table summarization using hierarchies reduced by applying (instead of *tRedux*) k-METIS clustering [73] over the concept similarity graph described in Section 6.5.1: the k-METIS algorithm guarantees that all partitions will be approximately equally distributed[3]. In these experiments, we vary the number of partitions as 20%, 30%, 40% and 50%

[3]In the literature exists also h-METIS, that has an unbalance parameter which allows the algorithm to deviate from an equal distribution. The h-METIS algorithm did not provide significantly better results than the k-METIS algorithm. So, the k-METIS algorithm is preferable because of its shorter execution time [34]

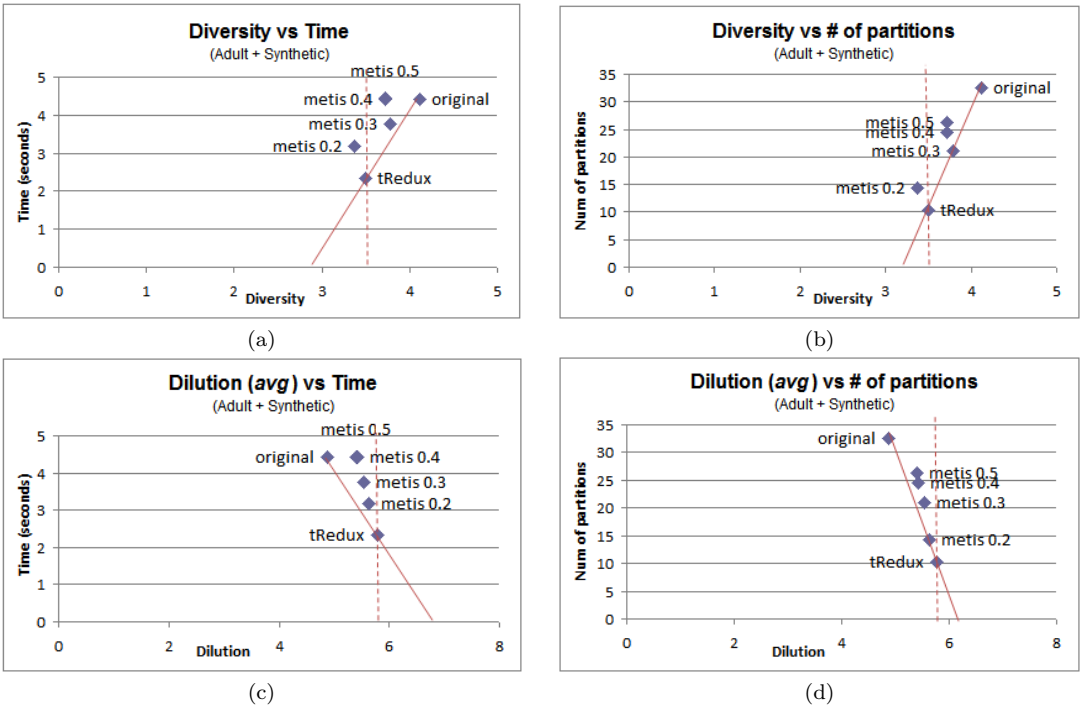


Figure 6.10: Comparisons among *tRedux*, original, and k-METIS approaches: (a) diversity-vs-time, (b) diversity-vs.-number of partitions, dilution-vs-time, (c) dilution-vs-number of partitions.

of the number of nodes in the input hierarchy (METIS-0.2, -0.3, -0.4 and -0.5 in the charts).

The experiments reported in this Section are high-level averages of all experiments carried out with varying system parameters. As we mentioned above, we varied values of k, the number of tuples, the hierarchy size. Then, for each alternative algorithm, we computed average diversity, average dilution, and average execution time and plotted them against each other to observe the general, high-level trends without focusing on the impacts of the specific system parameters. In Figure 6.10, the first scheme, "*original*", does not use hierarchy reduction, while the other schemes, "*tRedux*" and "k-METIS", are both instances of meta-data hierarchy reduction based table summarization approach.

Diversity vs. Time

Figure 6.10(a) shows the amount of diversity maintained by alternative schemes against the amount of time required by the table summarization algorithm. As can be seen in this figure, table summarization using the original hierarchies provides the highest diversity; but also takes the greatest amount of time. METIS algorithms with 40% and 50% hierarchy nodes cause drops in the diversity, without any significant temporal gain. METIS with 20% and 30% nodes result in some gains in time; but the highest gain in time occurs when using *tRedux* for summaries. Most importantly though, the diversity-vs-time behavior (highlighted by the slopes of the line segments that connect the point corresponding to *original* summaries with the points corresponding to the algorithms), is the best for *tRedux*. Overall, *tRedux* provides a $\sim$ 50% gain in execution time, with only a $\sim$ 15% reduction in diversity.

Diversity vs. # of Nodes in the Reduced Hierarchy

Figure 6.10(b) shows the diversity maintained by alternative schemes against the number of nodes (partitions) in the reduced hierarchy. As expected, there is a correlation with the number of nodes in the hierarchy with the overall diversity. However, as can be seen comparing METIS with 20% of nodes and *tRedux* results, *tRedux* is able to maintain a similar amount of diversity with smaller number of nodes in the hierarchy.

Dilution vs. Time

Figure 6.10(c) shows dilution[4] against table summarization time: the highest absolute and relative (to dilution) time gains are achieved by the *tRedux*.

Dilution vs. # of Nodes in the Reduced Hierarchy

Figure 6.10(d) shows the dilution[5] caused by alternative schemes against the number of nodes in the reduced hierarchy. As can be seen here, as expected, the smaller the number of nodes in the hierarchy, the higher the resulting dilution is. On the other hand, among the different metadata reduction

[4] In this setting, we are using the agnostic *avg* combination function to compute dilution. In these experiments, the effect of the dilution definition on the result were extremely minute; thus charts considering other functions (min, max, sum) are omitted for the sake of space.

[5] Again, using agnostic *avg* combination function.

schemes, *tRedux* has the best relative dilution behavior: a 66% drop in the number of nodes in the hierarchy results in only a less than 20% increase in dilution.

Summary

The results in this Section shows that *tRedux* is able to reduce the taxonomy (based on its inherent structure, without requiring the size of the output taxonomy as an input) in a way that provides the best diversity-time and dilution-time trade-off. Algorithms, like k-METIS can be used as the base graph partitioner if the user would like to reduce the sizes of the input taxonomies beyond what is structurally recommendable (albeit at the cost of further information loss).

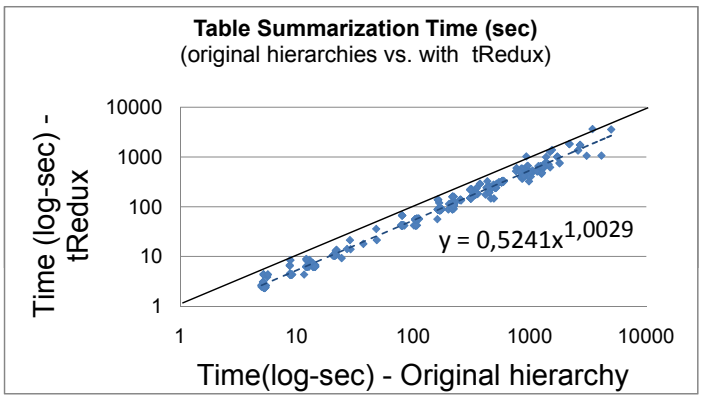
6.7.2 Dissecting tRedux

We have looked at the *high-level* behavior of the various algorithms and seen that metadata reduction based table summarization can provide significant time gains, while resulting in relatively small increase in dilution and drop in diversity. We have also seen that among alternative ways to taxonomy reduction, *tRedux* has the best dilution-time and diversity-time behaviors.

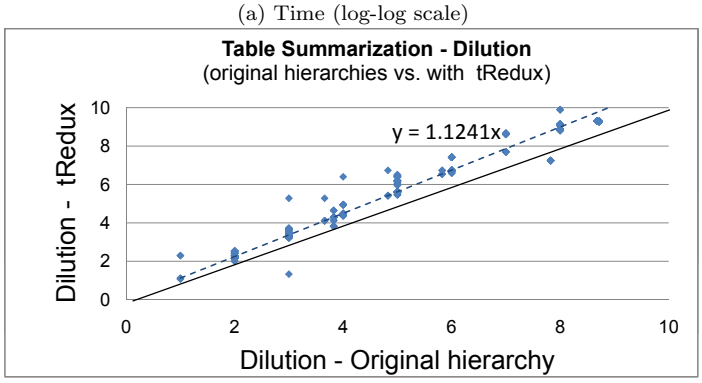
In this Section, we look at the *tRedux* algorithm in greater detail and study how different problem parameters affect dilution, diversity and time behaviors of *tRedux*. In particular, we vary (a) the imbalance tolerance value, θ, (b) the number of tuples in the input table, (c) the value distributions in the data, (d) the sizes of the hierarchies, and (e) the heights of the hierarchies, and compare the *tRedux*-supported summaries with summaries using original hierarchies. We mostly experiment with synthetic data where we can freely change various parameters and observe the behaviour of tRedux, but we also include results with the Adults data set.

Overview

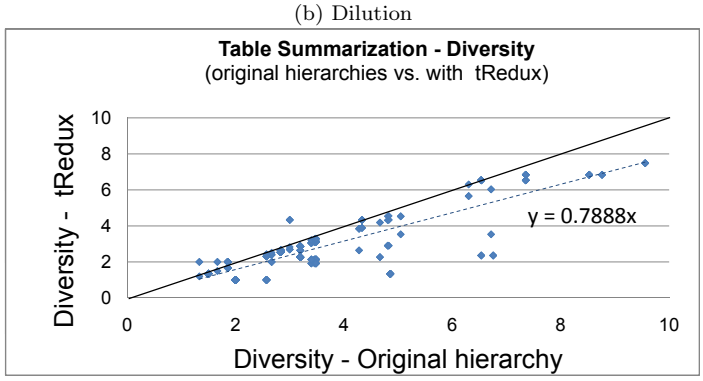
First, Figures 6.11(a),(b) and (c) bring together all experiment instances (independent of their parameters) into three tables which plot performance measures (time, dilution and diversity) for table summarization with the original hierarchy against table summarization with *tRedux*. As the trend line in Figure 6.11(a) shows, on the average the summarization times with *tRedux* is just $\sim 50\%$ of the summarization times needed with the original summary (i.e., summarization is $2\times$ as fast when using *tRedux*) and this behavior is highly consistent. Moreover, the average loss in terms of dilution



(a) Time (log-log scale)



(b) Dilution



(c) Diversity

Figure 6.11: Table summarization total results with and without *tRedux*.

(Figure 6.11(b)) is only $\sim$ 12% higher when using a summarized taxonomy, while the average loss in terms of diversity is $\sim$ 21% (Figure 6.11(c)).

Next we consider the impact of the individual parameters on these three

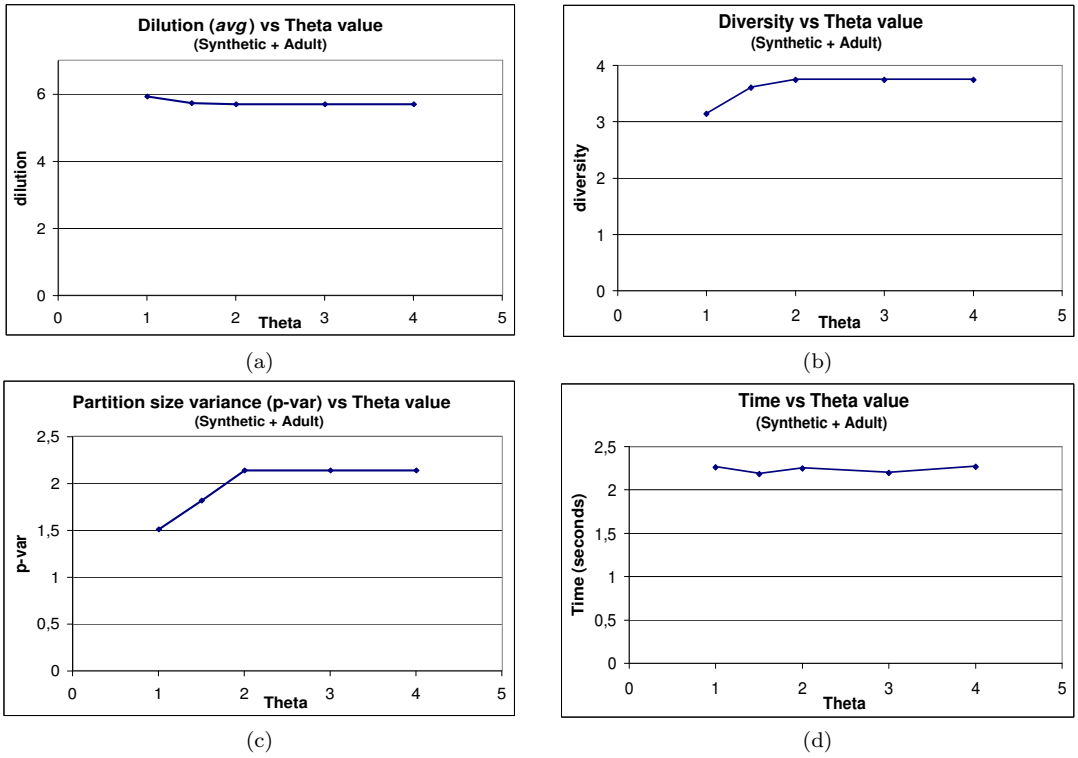


Figure 6.12: Dilution, diversity, partition size variance, and time for different imbalance tolerance values.

measures.

Impact of the Partition Imbalance (Θ Parameter)

As introduced in Section 6.5.2, depending on the need, the user can balance the resulting partitions by using the parameter θ. In fact, when reducing input hierarchies, creating partitions with widely varying sizes might be undesirable: some partitions in the reduced hierarchy will be more precise (corresponding to only a few entries in the original hierarchy), while some others will be very vague (corresponding to a large number of values). On the other hand, requiring perfectly balanced partitions might also be counter-productive since this may result in nodes in the re-constructed hierarchy that are consisting of poorly related (non-homogeneous) concepts in the original hierarchy.

Indeed, as shown in Figures 6.12(a) and (b), requiring strictly balanced

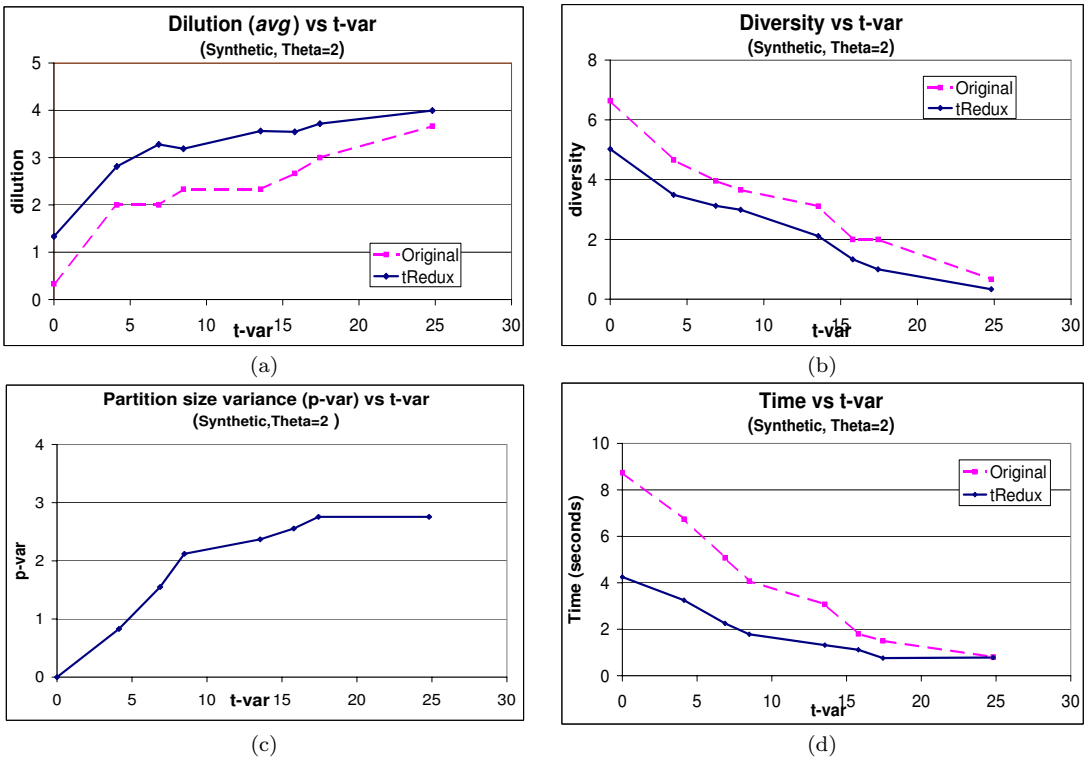


Figure 6.13: (a) Dilution, (b) diversity, (c) partition size variance, and (d) time as a function of tuple value count variance.

partitions ($\theta = 1$) results in a slightly higher dilution and lower diversity. Any $\theta \geq 2$, however, provides the same performance; this is because for such large θ values there is no need to re-balance the partitions.

Thus, while we foresee that in most cases *tRedux* will be used with $\theta = 2$, we also recognize that some applications may require balanced partitions and (as shown in Figure 6.12(c)), in these cases, θ can be used to control the balance of the partitions. Note that, since the number of partitions stays the same, θ does not affect the table summarization time (Figure 6.12(d)). Notice that, while diversity and dilution stay constant for $\theta >= 2$, there is no need to consider very high values of θ.

Impact of the Value Distribution in the Data Table

We also experimented with different value distributions in the data table. Results for this setup are for a single attribute (with a balanced hierarchy

with 127 nodes and 64 leaves) and 256 tuples in the table.

In this experiment, we varied the *tuple count variance* (*t-var*) of the table between 0 and ~ 25 (in the case with *t-var* = 24.80, we have almost all tuples distributed on only one leaf of the domain hierarchy and the other leaves are only represented by one tuple each). For each t-var, we analyzed 3 different randomly generated sets of tuples. Each presented result is the average of these three cases.

As can be seen in Figures 6.13(a) and (b), large variances in the tuple distributions negatively impact the dilution and diversity for summarization. As expected, the original hierarchy provides better diversity and dilution than *t-Redux*, but is much slower Figure 6.13(d). As *t-var* increases, the dilution, diversity, and execution time behaviors of *t-Redux* and the original scheme approach each other. This is because an increase in the count variation also causes an increase in the partition size variation (Figure 6.13(c)) and when the partition variances are higher, Samarati's algorithm tends to pick nodes closer to the root instead of analyzing combinations of the internal nodes.

Impact of the Table Size

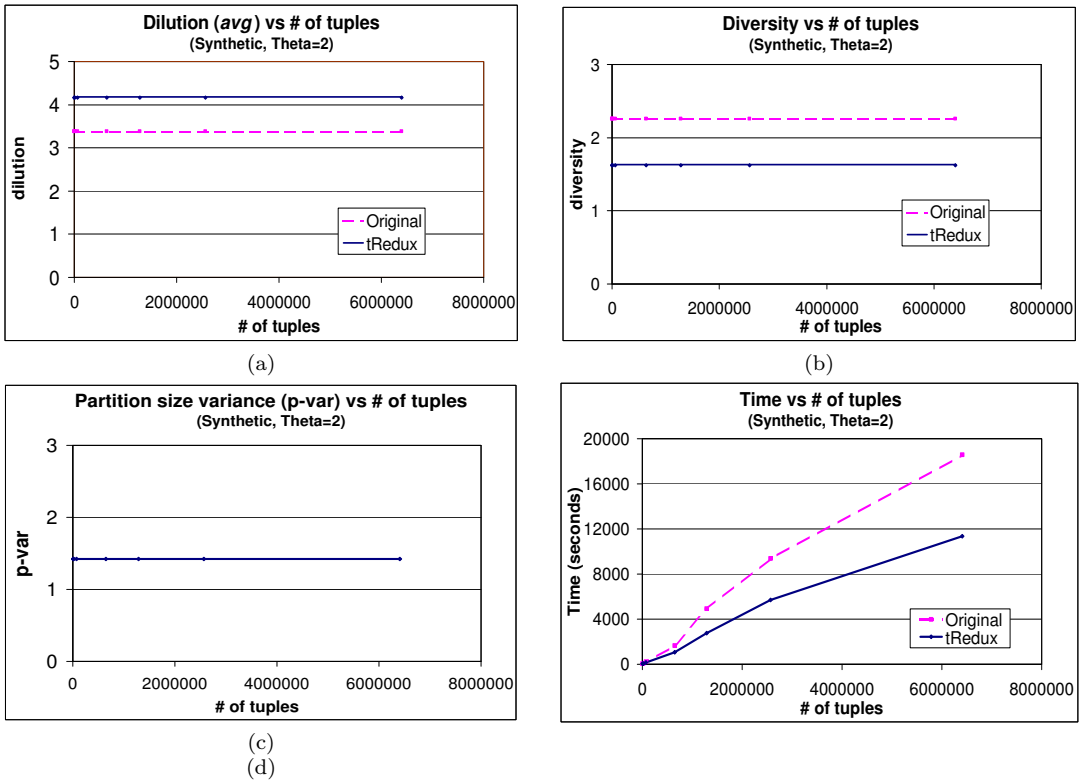
To observe the effect of the table size, we considered a summarization scenario with a single attribute (having an unbalanced hierarchy with 31 nodes and 16 leaves). We varied the database size from ~ 50 to ~ 5000000, with $\times 10$ increments. For these experiments, we set *t-var* to 0.

As these experiments show, as long as the tuples in the table are selected using the same value distribution, independent of the size of the table, the dilution and diversity stays the same. Figure 6.14(a),(b),(c) and (d) show the obtained results. Note that the time cost of the original scheme increases faster than the time cost of *tRedux* supported summarization as the table size increases.

Effects of the Number of Nodes in the Input Hierarchies

In order to study the impact of the number of nodes in the input hierarchy, we selected 6 different hierarchies with different number of nodes (57, 115, 230, 460, 921 and 1843 nodes) but having the same height (13 levels). For each of the 6 considered cases, we analyzed 3 different random generated hierarchies and the presented results are the averages of these. For these experiments, we maintain *t-var* at 0.

The results in Figure 6.15(a) and (b) show that the dilution and diversity



(a) (b) (c) (d)

Figure 6.14: (a) Dilution, (b) diversity, (c) partition size variance, and (d) time as a function of the number of tuples.

behaviors of *tRedux* are not affected by the number of nodes. As shown in Figure 6.15(c), partition size variance *p-var* is also not affected by the changes in the number of nodes. As shown in Figure 6.15(d), the number of nodes affects the process in terms of execution time (because the algorithm needs to consider more nodes as candidates for the summary); the benefits of the *tRedux* scheme is more apparent for larger hierarchies.

Effects of the Heights of the Input Hierarchies

For these experiments, we used 8 different hierarchies with the same numbers of nodes (460 nodes of which 256 are leaves), but with different numbers of levels (8 through 17). For each of these cases, we experimented with 3 different random generated hierarchies; the presented results are averages. The table has 1024 tuples and has *t-var* of 0.

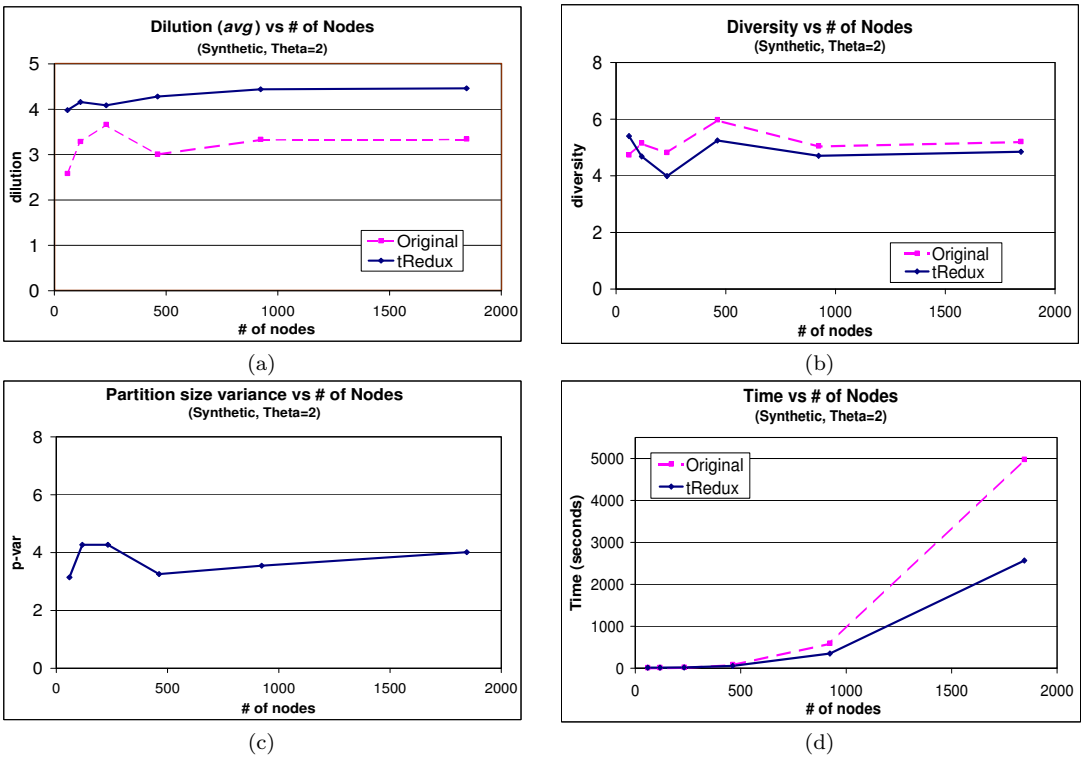


Figure 6.15: (a) Dilution, (b) diversity, (c) partition size variance, and (d) time as a function of hierarchy size.

As Figure 6.16(a) shows dilution of *t-Redux* is not affected by the height of the hierarchy. This is largely because of the fact that the partition size variance *p-var* is not affected by the changes in the hierarchy height (Figure 6.16(c)). The diversity of the summaries, however, increases with the height of the hierarchy (Figure 6.16(b)): since diversity is measured by the distances of the nodes in the hierarchy, when the height increases, the diversity also increases. The height of the hierarchy does not affect the time cost of the summarization process for both *tRedux* and original alternatives (Figure 6.16(d)).

Effect of the Number of Attributes in the Summary

Figure 6.17 shows the effect of the number of attributes. Each of the plots includes results from many experiments (varying two data sets, number of tuples in the table and *t-var*, number, sizes, and heights of hierarchies) in

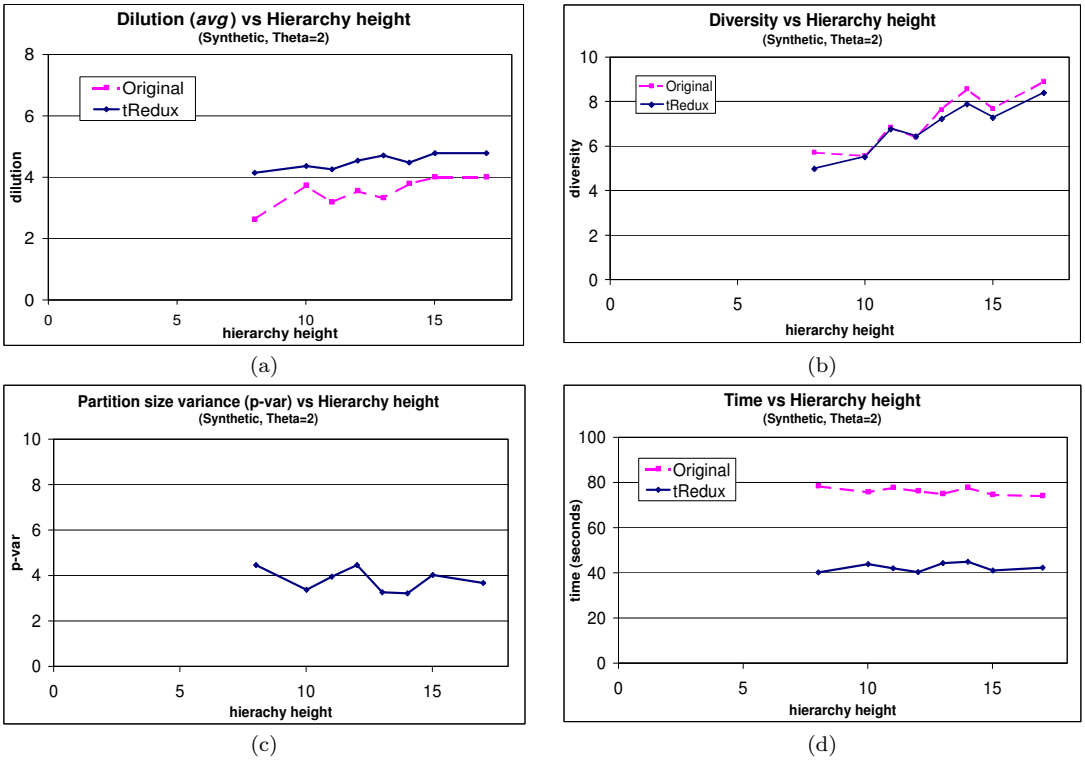


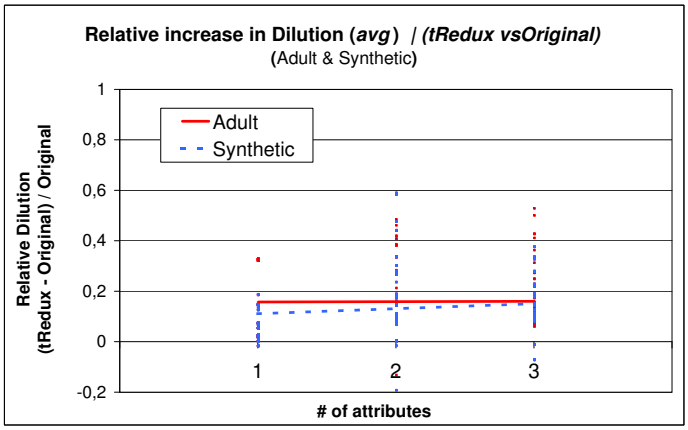
Figure 6.16: (a) Dilution, (b) diversity, (c) partition size variance, and (d) time as a function of hierarchy height.

a single chart; these experiments are clustered in terms of the number of attributes in the data being summarized and trend lines are drawn to help observe the general trends.

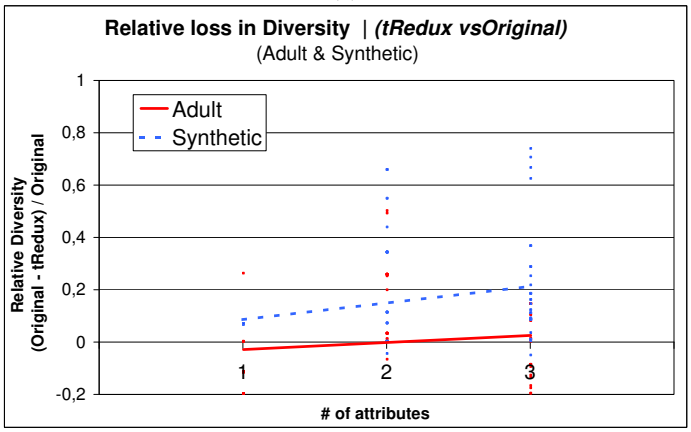
Figure 6.17(a) shows that, for both Adult (red line) and synthetic (dashed blue line) data sets, the amount of dilution increases with the number of attributes, but the loss due to *tRedux* stays more or less constant, $\leq 20\%$. On the synthetic data, diversity shows a $\sim 10\%$ increase in loss when the number of attributes increase from 1 to 3; on the Adult data set, however, the impact of the number of attributes is rather negligible. It is also interesting to note that, on the Adult data set, the loss in diversity due to the use of *tRedux* is very close to 0. As Figure 6.17(c) shows, the execution time gain due to the use of *tRedux* increases with the number of attributes; for the Adult set the gain increases from around 30% (i.e., $\sim 1.5\times$ as fast as the original scheme) for the case with a single attribute to more than 40% (i.e., almost $2\times$ as fast as the original scheme) for the case with three

attributes.

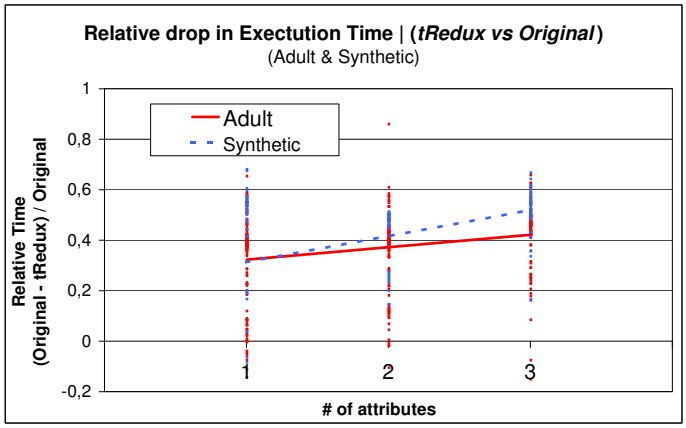
In conclusion we have shown that, by pre-processing input value hierarchies before they are used in metadata supported table summarization process, we can significantly reduce the summarization cost, without causing significant quality degradations in the resulting table summaries. We have also introduced a novel hierarchy summarization approach, *tRedux*, tailored towards this task and have shown that this approach provides the best time gain vs. quality trade-off against alternative schemes. We studied how the pre-conditions could affect the table summarization process and show which parameters could change and vary the results of the entire process. We evaluated different experiments, considering all the possible parameters (different number hierarchies and different datasets), and compared *tRedux* results with the k-METIS famous clustering approach, highlighting the benefits that we can obtain using our approach. Therefore, *tRedux* provides a novel metadata pre-processing method that can reduce significantly the amount of time needed by a table summarization process, providing also a parameter (the *tolerance value*) that permits to leverage the size of the partitions, and therefore, to adapt the method to different summarization needs.



(a)



(b)



(c)

Figure 6.17: Dilution, diversity and relative execution time with varying number of attributes.

Chapter 7

Conclusions

In this thesis we explored the importance of the meta-data hierarchies in the retrieval and exploration process of large data corpora; we initially explained the motivations of our work and analyzed the most relevant related works, studying the problems that affect the existing approaches.

Then, we introduced novel mechanisms to formalize the meta-data content knowledge in order to enrich the pure structural information with information extracted from the considered corpora [21].

Based on these formalization approaches, we provided novel solutions (based on different data format, user needs and tasks) that produce adapted meta-data hierarchies that best represent the considered content knowledge. In fact, we have shown that, if it is necessary, an adaptation of input meta-data hierarchies can apport enormous benefits to retrieval processes, reducing the overall redundancy in the indexed contents (that can be now efficiently associated to the meta-data nodes), without causing quality degradations in the produced structure. We also studied the advantages and disadvantages of the proposed methods, by introducing different measures that quantify the losses and gains with respect to the most relevant properties of a meta-data hierarchy. We analyzed the previous existing works on meta-data evaluations and, based on them, we proposed many different measures that can greatly analyze the generated structures.

Moreover, we evaluated our works with many different data sets, including real and synthetic ones, proving the effectiveness of the proposed methods and comparing their performances with other alternative schemes. We also analyzed the feedback of human users by user studies that showed that the proposed algorithms are able to adapt the meta-data in new compact and understandable structures from a human point of view.

In particular, considering the text environment, we have introduced a novel context-based narrative interpretation [24]; this narrative is then appropriately segmented and summarized based on the data corpus content. We showed that this approach provides significant benefits in terms of redundancy reduction and domain coverage improvement in comparison with standard alternative aggregation schemes.

On the other hand, if the adaptation method is used for distilling value hierarchies for data tables (and support table summarization process) we have shown that is possibile to significantly reduce the entire process cost, again without causing significant quality degradations in the resulting adapted meta-data hierarchy [20]. The idea that guided our work was to adapt the original method introduced for text documents by analyzing the constraints and the needs imposed by the data table format. In particular, considering that the exploration of very large data tables is generally not realized through taxonomical structures (but only uses them as background knowledge), it is unrelevant to propose a method that permits to select the number of documents. For this reason, we optimize the proposed method by using a clustering approach that does not require any specific output number of clusters (that can be, from the other side, very useful in a text environment where the user can have specific requirements) And again, the result provided the best time gain versus quality trade-off against alternative schemes.

Moreover, we proposed novel exploration approaches that leverage the adapted meta-data hierarchies to provide a more effective navigation within the data space. The proposed methods tightly integrate the semantic context(extracted from the previously adapted meta-data structures) with the keywords extracted from the available data corpora.

In the text environment, this has been obtained by introducing a novel keywords-by-concepts (KbC) graph, which is a weighted graph constructed relying on a spreading activation like technique by a tight integration of the adapted domain knowledge (i.e the adapted hierarchy, considered as the semantic context) with the most domain relevant keywords extracted from the text corpus. KbC graph is then leveraged for developing a novel Context-based Search and Navigation (CoSeNa) system for context-aware navigation and document retrieval [25, 26]. The unique aspect of our approach is that it mines emerging topic correlations within the data, exploiting both statistical information coming from the documents' corpus and the structured knowledge represented by the input adapted meta-data. The case studies and experiments, presented in this thesis, showed how this approach enables contextually-informed strengthening and weakening of semantic links

between different concepts.

In the same way, the adapted meta-data hierarchy can be positively used for data table exploration purposes, by relying on the idea of summarization. In fact, our approach takes as input a data table and (using the previously calculated adapted meta-data hierarchy) returns a reduced version of it, permitting the user to analyze only few entries that represent the general data trends. The result provides tuples with less precision than the original, but still informative of the content of the database, permitting the user to easily explore the data knowledge. Moreover, this reduced form can not only be presented to the user for exploration but it could also be used as input for advanced data mining processes. In fact, with this method, each tuple in the original table is represented, in the summary, with a more generic tuple that summarizes its knowledge; on the other hand, each new tuple in the summary represents a maximal set of original tuples that can be expressed by it minimizing the information loss. We finally showed that the presented summarization-based exploration approach can be able to lead the user in a more effective navigation within the data tables by minimizing the overall redundancy in the data (such as approximate functional dependencies and other patterns) and the information loss (due to the reduction in details).

All the presented adaptation methods of hierarchical meta-data structures can be applied to the many applications (especially in web environments) that need to organize the data by using hierarchical taxonomies. In fact, by using the proposed methods, it is possible to obtain a meta-data structure that best represents the indexed contents, and therefore can positively help the user navigate within the data space. Moreover, considering the many very dynamic environments (where the indexed data rapidily evolve), it is also possible to periodically update and re-structure the considered meta-data by automatically running the proposed adaptation methods on the new generated contents and keep the the taxonomy always aware of the indexed data.

7.1 Future Works

It is important to notice that, within the examined problem, many others solutions could be studied and analyzed in order to improve the efficacy of the obtained structures in guiding the exploration of the data corpora.

For example, considering the importance of the user's understanding of the adapted structures, an interesting future direction of this thesis is about the labeling strategy; in fact, considering that the main aim of these adapta-

tion techniques is to produce a hierarchy that can be positively used by the users to explore the data sets, the clarity of the labels of each new generated concept-node is essential. Indeed, an *effective* meta-data structure needs to be clear and explanatory, from a user point of view, in order to describe, as precisely as possible, the knowledge of each node and discriminate, as much as it can, their meaning with respect to all the other concepts in the hierarchy (in order to permit a better navigation into the data and reduce the overall redundancy).

Thus, instead of considering only the structural constraints imposed by the original hierarchy, we are planning to extract, from the considered context, for each retrieved cluster of concept nodes, a term (or a set of terms) that can more effectively describe its content. In fact, the main idea is that the knowledge expressed by a cluster of nodes could be better described by a keyword (or set of keywords), extracted from the data corpus, instead of a label created by merging the original ones. In fact, as in the natural language, a complex concept identified by a set of terms could be described by another word that best summarizes the meaning of the concept defined by the set of considered terms.

Thus, we are planning to investigate this problem by analyzing the semantical relationships that exist between *a set of hierarchy nodes* (instead of considering each one of them separately) and the considered data contents, in order to extract the terms that could better represent them in the adapted hierarchy. This is an evolution of the proposed strategy that considers each node in the original hierarchy as a separate entity and does not take into account their similarities (or in same cases even overlappings) retrieved by the concept clustering step.

Lastly, as already explained, the proposed methods only condense the input meta-data hierarchies and cannot add new concepts to the considered structures: thus, we are planning to explore this possibility by applying statistical approaches that analyze the data associated to each cluster and extract a set of features that are highly correlated it (and also very poorly correlated with all the others). These features can be therefore transformed into new concept nodes (representing them with the most discriminating keyword labels) and added to the existing structure based on occurrences' information.

Moreover, we are also planning to use machine learning techniques and alternative background knowledge (as other meta-data hierarchy that describe the same domain) to improve the efficiency of the clustering step and better analyze the similarities among concepts. In particular, we are planning to extend our meta-data adaptation method in order to consider many

different domain hierarchy structures and therefore distill a meta-data by integrating the knowledge expressed by all of them.

Bibliography

[1] Charu C. Aggarwal, Stephen C. Gates, and Philip S. Yu. On the merits of building categorization systems by supervised clustering. In *KDD'1999 – Proceedings of the Fifth ACM SIGKDD*, pages 352–356, San Diego, US, 1999. ACM Press.

[2] Gagan Aggarwal, Krishnaram Kenthapadi Tomas Feder, Rajeev Motwani, Rina Panigrahy, Dilys Thomas, and An Zhu. Approximation algorithms for k-anonymity. *Journal of Privacy Technology*, 2005.

[3] Rayner Alfred and Dimitar Kazakov. Data summarization approach to relational domain learning based on frequent pattern to support the development of decision making. In *ADMA*, pages 889–898, 2006.

[4] A. Asuncion and D.J. Newman. UCI machine learning repository, 2007.

[5] Xiangzhi Bai and Fugen Zhou. Edge detection based on mathematical morphology and iterative thresholding. In *CIS*, pages 953–962, 2006.

[6] Regina Barzilay, Noemie Elhadad, and Kathleen R. McKeown. Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*, 17, 2002.

[7] Marcia J. Bates. The design of browsing and berrypicking techniques for the online search interface. *Online Review*, 13(5):407–424, 1989.

[8] Roberto J. Bayardo and Rakesh Agrawal. Data privacy through optimal k-anonymization. In *Proc. of ICDE*, pages 217–228, 2005.

[9] Michael W. Berry. Large scale sparse singular value computations. *International Journal of Supercomputer Applications*, 6:13–49, 1992.

[10] Krishna Bharat and Monika R. Henzinger. Improved algorithms for topic distillation in a hyperlinked environment. pages 104–111, 1998.

[11] Branimir Boguraev and Mary S. Neff. Discourse segmentation in aid of document summarization. In *HICSS*, 2000.

[12] Christopher Brewster, Fabio Ciravegna, and Yorick Wilks. User-centred ontology learning for knowledge management. In *NLDB '02*, pages 203–207. Springer-Verlag, 2002.

[13] Francesco Buccafurri, Filippo Furfaro, Domenico Saccà, and Cristina Sirangelo. A quad-tree based multiresolution approach for two-dimensional summary data. In *SSDBM'2003*, pages 127–140, USA, 2003.

[14] Adam L. Buchsbaum, Glenn S. Fowler, and Raffaele Giancarlo. Improving table compression with combinatorial optimization. *J. ACM*, 50(6):825–851, 2003.

[15] A.L. Buchsbaum, F. Caldwell, K.W. Church, G.S. Fowler, and S. Muthukrishnan. Engineering the compression of massive tables: an experimental approach. In *SODA*, pages 175–184, 2000.

[16] Paul Buitelaar, Philipp Cimiano, and Bernado Magnini, editors. *Ontology Learning from Text: Methods, Evaluation and Applications*. IOS Press, 2005.

[17] J.W. Byun, A. Kamra, E. Bertino, and N. Li. Efficient k-anonymization using clustering techniques. In *DASFAA*, 2007.

[18] K. Selçuk Candan, Huiping Cao, Yan Qi, and Maria Luisa Sapino. System support for exploration and expert feedback in resolving conflicts during integration of metadata. *VLDB J.*, 17(6):1407–1444, 2008.

[19] K. Selçuk Candan, Huiping Cao, Yan Qi, and Maria Luisa Sapino. Table summarization with the help of domain lattices. In *Proc. of CIKM*, pages 1473–1474, 2008.

[20] K. Selçuk Candan, Mario Cataldi, and Maria Luisa Sapino. Reducing metadata complexity for faster table summarization. In *EDBT 2010, 13th International Conference on Extending Database Technology, Lausanne, Switzerland, March 22-26, 2010, Proceedings*, pages 240–251, 2010.

[21] K. Selçuk Candan, Mario Cataldi, Maria Luisa Sapino, and Claudio Schifanella. Structure- and extension-informed taxonomy alignment.

In *Proceedings of the 4th International VLDB Workshop on Ontology-based Techniques for DataBases in Information Systems and Knowledge Systems, ODBIS 2008, Co-located with the 34th International Conference on Very Large Data Bases*, pages 1–8, 2008.

[22] K. Selçuk Candan, Jong Wook Kim, Huan Liu, and Reshma Suvarna. Discovering mappings in hierarchical data from multiple sources using the inherent structure. *Knowledge and Information Systems*, 10(2):185–210, 2006.

[23] Stuart K. Card and Jock Mackinlay. The structure of the information visualization design space, 1996.

[24] Mario Cataldi, K. Selçuk Candan, and Maria Luisa Sapino. Anita: A narrative interpretation of taxonomies for their adaptation to text collections. In *CIKM'10: Proceedings of the 19th Conference on Information and Knowledge Management*, 2010.

[25] Mario Cataldi, Claudio Schifanella, K. Selçuk Candan, Maria Luisa Sapino, and Luigi Di Caro. Cosena: a context-based search and navigation system. In *MEDES '09: International ACM Conference on Management of Emergent Digital EcoSystems, Lyon, France, October 27-30, 2009*, 2009.

[26] Mario Cataldi, Claudio Schifanella, K. Selçuk Candan, Maria Luisa Sapino, and Luigi Di Caro. Context-informed knowledge extraction from document collections to support user navigation. In *Submitted to Internation Journal (under revision)*, 2010.

[27] Soumen Chakrabarti, Mukul Joshi, and Vivek Tawde. Enhanced topic distillation using text, markup tags, and hyperlinks. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 208–216, New York, NY, USA, 2001. ACM.

[28] Surajit Chaudhuri and Umeshwar Dayal. An overview of data warehousing and olap technology. *SIGMOD Record*, 26(1):65–74, 1997.

[29] Hao Chen and Susan Dumais. Bringing order to the web: automatically categorizing search results. In *CHI '00*, pages 145–152. ACM, 2000.

[30] Venkata Snehith Cherukuri and K. Selçuk Candan. Propagation-vectors for trees (pvt): concise yet effective summaries for hierarchical data and trees. In *LSDS-IR '08*, pages 3–10, 2008.

[31] Ed Huaihsin Chi and John Riedl. An operator interaction framework for visualization systems. In *INFOVIS '98: Proceedings of the 1998 IEEE Symposium on Information Visualization*, pages 63–70, Washington, DC, USA, 1998. IEEE Computer Society.

[32] P. Cimiano, A. Hotho, and S. Staab. Learning concept hierarchies from text corpora using formal concept analysis. *Journal of Artificial Intelligence Research*, 24:305–339, 2005.

[33] Philipp Cimiano, Aleksander Pivk, Lars Schmidt-Thieme, and Steffen Staab. Learning taxonomic relations from heterogeneous evidence, 2004.

[34] Mark Connor and Jon Herlocker. Clustering items for collaborative filtering, 2001.

[35] John M. Conroy and Dianne P. O'leary. Text summarization via hidden markov models. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 406–407, New York, NY, USA, 2001. ACM.

[36] Hang Cui, Ji-Rong Wen, Jian-Yun Nie, and Wei-Ying Ma. Probabilistic query expansion using query logs. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pages 325–332, New York, NY, USA, 2002. ACM.

[37] Alfredo Cuzzocrea, Domenico Saccà, and Paolo Serafino. A hierarchy-driven compression technique for advanced olap visualization of multidimensional data cubes. In *DaWaK*, pages 106–119, 2006.

[38] AnHai Doan, Pedro Domingos, and Alon Y. Halevy. Reconciling schemas of disparate data sources: a machine-learning approach. In *SIGMOD '01: Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, pages 509–520, New York, NY, USA, 2001. ACM.

[39] AnHai Doan, Pedro Domingos, and Alon Y. Levy. Learning source description for data integration. In *WebDB (Informal Proceedings)*, pages 81–86, 2000.

[40] AnHai Doan, Jayant Madhavan, Pedro Domingos, and Alon Y. Halevy. Ontology matching: A machine learning approach. In *Handbook on Ontologies*, pages 385–404. 2004.

[41] Noemie Elhadad and Kathleen R. McKeown. Towards generating patient specific summaries of medical articles. In *NAACL 2001*, pages 31–39, 2001.

[42] D. Ellis. A behavioral approach to information retrieval system design. *J. Doc.*, 45(3):171–212, 1989.

[43] Martin Ester, Hans-Peter Kriegel, Joerg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. of ACM SIG-KDD*, pages 226–231, 1996.

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

[45] Alfredo Favenza, Mario Cataldi, Maria Luisa Sapino, and Alberto Messina. Topic development based refinement of audio-segmented television news. In *NLDB '08*, pages 226–232, Berlin, Heidelberg, 2008. Springer-Verlag.

[46] Fellbaum. *WordNet: An Electronic Lexical Database*. The MIT Press, May 1998.

[47] Gary William Flake, Robert E. Tarjan, and Kostas Tsioutsiouliklis. Graph clustering and minimum cut trees. *Internet Mathematics*, 1:385–408, 2004.

[48] Achille Fokoue, Aaron Kershenbaum, Li Ma, Edith Schonberg, and Kavitha Srinivas. K.: The summary abox: Cutting ontologies down to size. In *ISWC*, pages 343–356, 2006.

[49] W.G. Charles G.A. Miller. Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6(1), 1991.

[50] Susan Gauch, Jason Chaffee, and Alaxander Pretschner. Ontology-based personalized search and browsing. *Web Intelli. and Agent Sys.*, 1(3-4):219–234, 2003.

[51] Gabriel Ghinita, Panagiotis Karras, Panos Kalnis, and Nikos Mamoulis. Fast data anonymization with low information loss. In *Proc. VLDB*, pages 758–769, 2007.

[52] Roy Goldman and Jennifer Widom. Dataguides: Enabling query formulation and optimization in semistructured databases. In *VLDB'97*, pages 436–445, 1997.

[53] Jade Goldstein, Mark Kantrowitz, Vibhu Mittal, and Jaime Carbonell. Summarizing text documents: sentence selection and evaluation metrics. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 121–128, New York, NY, USA, 1999. ACM.

[54] Yihong Gong. Generic text summarization using relevance measure and latent semantic analysis. In *in Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2001.

[55] F. A. Grootjen and Th. P. van der Weide. Conceptual query expansion. *Data Knowl. Eng.*, 56(2):174–193, 2006.

[56] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. CURE: an efficient clustering algorithm for large databases. pages 73–84, 1998.

[57] David Harel and Yehuda Koren. On clustering using random walks. In *FSTTCS*, pages 18–41, 2001.

[58] Marti A. Hearst. Texttiling: A quantitative approach to discourse segmentation. Technical report, 1993.

[59] Marti A. Hearst. Texttiling: segmenting text into multi-paragraph subtopic passages. *Comput. Linguist.*, 23(1):33–64, March 1997.

[60] Sven Helmer. Measuring the structural similarity of semistructured documents using entropy. In *VLDB '07*, pages 1022–1032, 2007.

[61] Andreas Heß. An iterative algorithm for ontology mapping capable of using training data. In *ESWC*, pages 19–33, 2006.

[62] Thomas Hofmann. The cluster-abstraction model: Unsupervised learning of topic hierarchies from text data. In *In IJCAI*, pages 682–687, 1999.

[63] Eduard H. Hovy. Automated discourse generation using discourse structure relations. *Artif. Intell.*, 63(1-2):341–385, 1993.

[64] David A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the Institute of Radio Engineers*, 40(9):1098–1101, September 1952.

[65] Carlos A. Hurtado, Claudio Gutiérrez, and Alberto O. Mendelzon. Capturing summarizability with integrity constraints in olap. *ACM Trans. Database Syst.*, 30(3):854–886, 2005.

[66] Carlos A. Hurtado and Alberto O. Mendelzon. Reasoning about summarizability in heterogeneous multidimensional schemas. In *ICDT*, pages 375–389, 2001.

[67] Ullrich Hustadt, Boris Motik, and Ulrike Sattler. Reducing shiq-description logic to disjunctive datalog programs. In *KR*, pages 152–162, 2004.

[68] Vijay S. Iyengar. Transforming data to satisfy privacy constraints. In *Proc. of ACM SIG-KDD*, pages 279–288, 2002.

[69] Anil K. Jain and Richard C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.

[70] H. Jing. Summary generation through intelligent cutting and pasting of the input document, 1999.

[71] Hongyan Jing and Kathleen R. McKeown. Cut and paste based text summarization. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 178–185, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[72] Thorsten Joachims, Dayne Freitag, and Tom Mitchell. Webwatcher: A tour guide for the world wide web, 1996.

[73] George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20:359–392, 1998.

[74] Stefan Kaufmann. Cohesion and collocation: Using context vectors in text segmentation. In *ACL*, 1999.

[75] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146, New York, NY, USA, 2003. ACM.

[76] Jong Wook Kim and K. Selçuk Candan. Cp/cv: concept similarity mining without frequency information from domain describing taxonomies. In *CIKM '06*, pages 483–492, 2006.

[77] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.

[78] Kevin Knight and Daniel Marcu. Summarization beyond sentence extraction: A probabilistic approach to sentence compression, 2002.

[79] Krishna Kummamuru, Rohit Lotlikar, Shourya Roy, Karan Singal, and Raghu Krishnapuram. A hierarchical monothetic document clustering algorithm for summarization and browsing search results. In *WWW '04*, pages 658–665. ACM, 2004.

[80] Steve Lawrence. Context in web search. *IEEE Data Engineering Bulletin*, 23:25–32, 2000.

[81] Dawn Lawrie, W. Bruce Croft, and Arnold Rosenberg. Finding topic words for hierarchical summarization. In *SIGIR '01*, pages 349–357. ACM, 2001.

[82] Dawn J. Lawrie and W. Bruce Croft. Generating hierarchical summaries for web searches. In *SIGIR '03*, pages 457–458, 2003.

[83] Kristen LeFevre, David J. DeWitt, and Raghu Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In *Proc. of ACM SIGMOD*, pages 49–60, 2005.

[84] Tao Li, Shenghuo Zhu, and Mitsunori Ogihara. Hierarchical document classification using automatically generated hierarchy. *J. Intell. Inf. Syst.*, 29(2):211–230, 2007.

[85] Wen-Syan Li and K. Selçuk Candan. Semcog: A hybrid object-based image and video database system and its modeling, language, and query processing. *TAPOS*, 5(3):163–180, 1999.

[86] Henry Lieberman. Letizia: An agent that assists web browsing. In *INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE*, pages 924–929, 1995.

[87] Ming-Ling Lo, Kun-Lung Wu, and Philip S. Yu. Tabsum: A flexible and dynamic table summarization approach. *ICDCS*, 2000.

[88] H.P. Luhn. The automatic creation of literature abstracts. *IBM Journal*, 2:159–165, 1958.

[89] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *Proc. of ICDE*, 2006.

[90] J. B. Macqueen. Some methods of classification and analysis of multivariate observations. In *Proceedings of the Fifth BSMSP*, pages 281–297, 1967.

[91] J. Madhavan, P. A. Bernstein, and E. Rahm. Generic schema matching with cupid. In *Proc. VLDB*, 2001.

[92] Jayant Madhavan, Philip A. Bernstein, and Erhard Rahm. Generic schema matching with cupid. In *VLDB '01: Proceedings of the 27th International Conference on Very Large Data Bases*, pages 49–58, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[93] Alexander Maedche and Steffen Staab. Ontology learning for the semantic web. *IEEE Intelligent Systems*, 16(2):72–79, 2001.

[94] Rila Mandala, Takenobu Tokunaga, and Hozumi Tanaka. Combining multiple evidence from different types of thesaurus for query expansion. In *Proc of ACM SIGIR'99*, pages 191–197, 1999.

[95] Daniel Marcu. From discourse structures to text summaries. The Proceedings of the ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization, pages 82-88, Madrid, Spain, July 11, 1997.

[96] Kathleen R. McKeown. *Text generation: using discourse strategies and focus constraints to generate natural language text.* Cambridge University Press, New York, NY, USA, 1985.

[97] Adam Meyerson and Ryan Williams. On the complexity of optimal k-anonymity. In *Proc. PODS*, pages 223–228, 2004.

[98] Renée J. Miller, Laura M. Haas, and Mauricio A. Hernández. Schema mapping as query discovery. In *VLDB '00: Proceedings of the 26th International Conference on Very Large Data Bases*, pages 77–88, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[99] Renée J. Miller, Mauricio A. Hernández, Laura M. Haas, Ling-Ling Yan, C. T. Howard Ho, Ronald Fagin, and Lucian Popa. The clio project: Managing heterogeneity. *SIGMOD Record*, 30(1):78–83, 2001.

[100] Tova Milo and Sagit Zohar. Using schema matching to simplify heterogeneous data translation. In *VLDB '98: Proceedings of the 24rd International Conference on Very Large Data Bases*, pages 122–133, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[101] P. Mitra, G. Wiederhold, and J. Jannink. Semi-automatic integration of knowledge sources. In *2nd International Conference on Information Fusion (FUSION 1999)*, 1999.

[102] Prasenjit Mitra, Gio Wiederhold, and Martin L. Kersten. A graph-oriented model for articulation of ontology interdependencies. Technical report, Stanford, CA, USA, 1999.

[103] Dunja Mladenic'. Using text learning to help web browsing. In *In Proc. SIGCHI*, pages 1–893, 2001.

[104] David J. Mooney, Sandra Carberry, and Kathleen F. McCoy. The generation of high-level structure for extended explanations. In *Proceedings of the 13th conference on Computational linguistics*, pages 276–281, 1990.

[105] Johanna D. Moore and Cécile L. Paris. Planning text for advisory dialogues. In *Proceedings of the 27th annual meeting on Association for Computational Linguistics*, pages 203–211, 1989.

[106] Raymond T. Ng and Jiawei Han. Clarans: A method for clustering objects for spatial data mining. *IEEE Transactions on Knowledge and Data Engineering*, 14(5):1003–1016, 2002.

[107] Wee Keong Ng and Chinya V. Ravishankar. Relational database compression using augmented vector quantization. In *ICDE*, pages 540–549, 1995.

[108] Andrew Nierman and H. V. Jagadish. Evaluating structural similarity in xml documents. In *WebDB*, pages 61–66, 2002.

[109] Christopher Olston, Ed H. Chi, and H. Chi. Scenttrails: Integrating browsing and searching on the web. *ACM Transactions on Computer-Human Interaction*, 10:177–197, 2003.

[110] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. In *WWW'98*, pages 161–172, 1998.

[111] Luigi Palopoli, Domenico Saccà, and Domenico Ursino. An automatic technique for detecting type conflicts in database schemes. In *CIKM '98: Proceedings of the seventh international conference on Information and knowledge management*, pages 306–313, New York, NY, USA, 1998. ACM.

[112] Luigi Palopoli, Giorgio Terracina, and Domenico Ursino. Dike: a system supporting the semi-automatic construction of cooperative information systems from heterogeneous databases. *Software: Practice and Experience*, 33(9):847–884, 2003.

[113] Torben Bach Pedersen, Christian S. Jensen, and Curtis E. Dyreson. Supporting imprecision in multidimensional databases using granularities. In *SSDBM*, pages 90–101, 1999.

[114] Pietro Perona and Jitendra Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:629–639, 1990.

[115] Simone Paolo Ponzetto and Roberto Navigli. Large-scale taxonomy mapping for restructuring and integrating wikipedia. In *IJCAI'09*, pages 2083–2088. Morgan Kaufmann Publishers Inc., 2009.

[116] Simone Paolo Ponzetto and Michael Strube. Deriving a large-scale taxonomy from wikipedia. In *AAAI*, pages 1440–1445, 2007.

[117] Josep M. Pujol, Ramon Sangüesa, and Jordi Delgado. Extracting reputation in multi agent systems by means of social network topology. In *AAMAS '02: Proceedings of the first international joint conference on Autonomous agents and multiagent systems*, pages 467–474, New York, NY, USA, 2002. ACM.

[118] Kunal Punera, Suju Rajan, and Joydeep Ghosh. Automatically learning document taxonomies for hierarchical classification. In *WWW '05*, pages 1010–1011, New York, NY, USA, 2005. ACM.

[119] Yan Qi and K. Selçuk Candan. Cuts: Curvature-based development pattern analysis and segmentation for blogs and other text streams. In *HYPERTEXT '06*, pages 1–10, New York, NY, USA, 2006. ACM.

[120] Yan Qi, K. Selçuk Candan, and Maria Luisa Sapino. Ficsr: feedback-based inconsistency resolution and query processing on misaligned data sources. In *Proc. of ACM SIGMOD*, pages 151–162, NY, USA, 2007.

[121] Yan Qi, K. Selçuk Candan, Junichi Tatemura, Songting Chen, and Fenglin Liao. Supporting olap operations over imperfectly integrated taxonomies. In *Proc. of ACM SIGMOD*, 2008.

[122] Yonggang Qiu and Hans-Peter Frei. Concept based query expansion. In *SIGIR '93*, pages 160–169. ACM.

[123] R. Rada, H. Mili, E. Bicknell, and M. Blettner. Development and application of a metric on semantic nets. *IEEE Transactions on Systems, Man, and Cybernetics In Systems*, 19:17–30, 1989.

[124] Dragomir Radev, Hongyan Jing, and Malgorzata Budzikowska. Centroid-based summarization of multiple documents: Sentence extraction, utility-based evaluation, and user studies. In *In ANLP/NAACL Workshop on Summarization*, pages 21–29, 2000.

[125] Davood Rafiei, Daniel L. Moise, and Dabo Sun. Finding syntactic similarities between xml documents. *Database and Expert Systems Applications*, 0:512–516, 2006.

[126] E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal*, 2001.

[127] Guillaume Raschia Régis Saint-Paul and Noureddine Mouaddib. General purpose database summarization. In *Proc. VLDB*, pages 733–744, 2005.

[128] Guillaume Raschia Régis Saint-Paul and Noureddine Mouaddib. Database summarization: The saintetiq system. In *Proc. of ICDE*, pages 1475–1476, 2007.

[129] Philip Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*, 11:95–130, 1999.

[130] R. Richardson, A. F. Smeaton, A. F. Smeaton, J. Murphy, and J. Murphy. Using wordnet as a knowledge base for measuring semantic similarity between words. Technical report, AICS, 1994.

[131] Soo Young Rieh and Hong Xie. Analysis of multiple query reformulations on the web: the interactive information retrieval context. *Inf. Process. Manage.*, 42(3):751–768, 2006.

[132] Jacques Robin and Kathleen McKeown. Empirically designing and evaluating a new revision-based model for summary generation. *Artif. Intell.*, 85(1-2):135–179, 1996.

[133] J. J. Rocchio. Relevance feedback in information retrieval. In G. Salton, editor, *The Smart retrieval system - experiments in automatic document processing*, pages 313–323. Englewood Cliffs, NJ: Prentice-Hall, 1971.

[134] Ian Ruthven and Mounia Lalmas. A survey on the use of relevance feedback for information access systems. *Knowl. Eng. Rev.*, 18(2):95–145, June 2003.

[135] Giovanni Maria Sacco. Dynamic taxonomies: A model for large information bases. *IEEE TKDE*, 12(3):468–479, 2000.

[136] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. In *Information Processing and Management*, pages 513–523, 1988.

[137] Pierangela Samarati. Protecting respondents' identities in microdata release. *IEEE Trans. Knowl. Data Eng.*, 13(6):1010–1027, 2001.

[138] Mark Sanderson. Word sense disambiguation and information retrieval. In *SIGIR '94*, pages 142–151. Springer-Verlag New York, Inc., 1994.

[139] Mark Sanderson and Bruce Croft. Deriving concept hierarchies from text. In *SIGIR'99*, pages 206–213, 1999.

[140] Jacques Savoy. Ranking schemes in hybrid boolean systems: a new approach. *J. Am. Soc. Inf. Sci.*, 48(3):235–253, 1997.

[141] P. Schmitz. Inducing ontology from flickr tags. In *WWW '06*, 2006.

[142] Eran Segal, Daphne Koller, and Dirk Ormoneit. Probabilistic abstraction hierarchies. In *In Advances in Neural Information Processing Systems 14*. MIT Press, 2001.

[143] K. Selçuk Candan, Huiping Cao, Yan Qi, and Maria Luisa Sapino. Alphasum: size-constrained table summarization using value lattices. In *Proc. EDBT*, pages 96–107, 2009.

[144] Azadeh Shakery and ChengXiang Zhai. Relevance propagation for topic distillation uiuc trec 2003 web track experiments. In *TREC*, pages 673–677, 2003.

[145] Shashank Pandit Shashank. Navigation-aided retrieval. In *Proc of WWW'07*, pages 391–400.

[146] Dou Shen, Jian tao Sun, Hua Li, Qiang Yang, and Zheng Chen. Document summarization using conditional random fields.

[147] Ted Pedersen Siddharth Patwardhan, Satanjeev Banerjee. Unsupervised word sense disambiguation using contextual semantic relatedness. In *SemEval'07*, pages 390–393, 2007.

[148] Karen Sparck Jones. Notes and references on early automatic classification work. *SIGIR Forum*, 25(1):10–17, 1991.

[149] Amanda Spink, Rider I. Building, Dietmar Wolfram, and Tefko Saracevic. Searching the web: the public and their queries. *J. of the American Society for Information Science and Technology*, 52:226–234, 2001.

[150] Gerd Stumme and Alexander Mädche. FCA-Merge: Bottom-up merging of ontologies. In *Proc. 17th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 225–234, Seattle (WA US), 2001.

[151] Lei Tang, Huan Liu, Jianping Zhang, Nitin Agarwal, and John J. Salerno. Topic taxonomy adaptation for group profiling. *ACM TKDD*, 1(4):1–28, 2008.

[152] Jaime Teevan, Christine Alvarado, Mark S. Ackerman, and David R. Karger. The perfect search engine is not enough: a study of orienteering behavior in directed search. In *SIGCHI'04*, pages 415–422. ACM, 2004.

[153] Simone Teufel and Marc Moens. Sentence extraction as a classification task. In *Proceedings of the ACL'97/EACL'97 Workshop on Intelligent Scallable Text Summarization*, Madrid, Spain, July 1997.

[154] Warren S. Torgerson. *Theory and methods of scaling.* R.E. Krieger Pub. Co.

[155] Dingding Wang, Tao Li, Shenghuo Zhu, and Chris Ding. Multi-document summarization via sentence-level semantic analysis and

symmetric matrix factorization. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 307–314, New York, NY, USA, 2008. ACM.

[156] Richard Wheeldon and Mark Levene. The best trail algorithm for assisted navigation of web sites. In *In Proc. LA-WEB Conference on Latin American Web Congress*, 2003.

[157] Kun-Lung Wu, Shyh-Kwei Chen, and Philip S. Yu. Dynamic refinement of table summarization or m-commerce. In *WECWIS*, pages 179–186, 2002.

[158] J. Xu, W. Wang, J. Pei, X. Wang, B. Shi, and A. Fu. Utility-based anonymization using local recoding. In *Proc. of ACM SIG-KDD*, pages 785–790, 2006.

[159] Jinxi Xu and W. Bruce Croft. Improving the effectiveness of information retrieval with local context analysis. *ACM Trans. Inf. Syst.*, 18(1):79–112, 2000.

[160] Chin yew Lin and Eduard Hovy. From single to multi-document summarization: A prototype system and its evaluation. In *In Proceedings of the ACL*, pages 457–464. MIT Press, 2002.

[161] Wen-Tau Yih, Joshua Goodman, Lucy Vanderwende, and Hisami Suzuki. Multi-document summarization by maximizing informative content-words. In *IJCAI*, pages 1776–1782, 2007.

[162] Oren Zamir and Oren Etzioni. Web document clustering: A feasibility demonstration. pages 46–54, 1998.

[163] Xiang Zhang, Gong Cheng, and Yuzhong Qu. Ontology summarization based on rdf sentence graph. In *WWW*, pages 707–716, USA.

[164] Ying Zhao and George Karypis. Evaluation of hierarchical clustering algorithms for document datasets. In *Data Mining and Knowledge Discovery*, pages 515–524. ACM Press, 2002.

[165] Lina Zhou. Ontology learning: state of the art and open issues. *Information Technology and Management*, 8(3):241–252, 2007.

[166] Jacob Ziv and Abraham Lempel. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, 23:337–343, 1977.