Εθνικό Μετσόβιο Πολυτεχνείο
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ &
ΥΠΟΛΟΓΙΣΤΩΝ

A Distributed Framework For Multimedia Using Small World Communities

---

SEYYED-MOHAMMAD JAVADI-MOGHADDAM

A THESIS PRESENTED TO THE NATIONAL TECHNICAL UNIVERSITY OF ATHENS IN FULFILMENT OF THE THESIS REQUIREMENT FOR THE DEGREE OF DOCTOR OF COMPUTER

SUPERVISOR: PROF. STEPHANOS KOLLIAS

February 2016

Εθνικό Μετσόβιο Πολυτεχνείο
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ &
ΥΠΟΛΟΓΙΣΤΩΝ

Ph.D. Degree Examination

for

SEYYED-MOHAMMAD JAVADI-MOGHADDAM

A Distributed Framework

For Multimedia Using Small World Communities

Thesis Committee: Stefanos Kollias

Andreas Stafylopatis

Giorgos Stamou

The Ph.D. Proposal has been examined and approved on

Stefanos Kollias (Professor) Andreas Stafylopatis (Professor)

Nikolaos Ouzounoglou (Professor) Konstantinos Karpouzis (Researcher A, ICCS-NTUA)

George Stamou (Assistant Professor) George Matsopoulos (Associate Profssor)

Phivos Mylonas (Assistant Professor, Ionian University)

Seyyed-Mohammad Javadi-Moghaddam

School of Electrical and Computer Engineering

National Technical University of Athens

Abstract

The continuous growth of multimedia content available all over the web is raising the importance of a distributed framework for searching it. A novel two-tier structure is introduced in this work, which focuses on the community concept to facilitate creation of ontological small worlds that can effectively assist the search task. As a result, user queries are forwarded to nodes that are likely to contain the relevant resources. Evaluation of the framework proves that the small world character of the proposed structure provides queries with better route selection and searching efficiency.

Keywords: Distributed Multimedia

ΠΕΡΙΛΗΨΗ

Η διατριβή έχει τίτλο « A Distributed Framework for Multimedia Using Small World Communities, Ένα Κατανεμημένο Πλαίσιο για Αναζήτηση και Πρόσβαση σε Πολυμεσικό Περιεχόμενο με χρήση του Μοντέλου των Κοινοτήτων Μικρού Κόσμου» και πραγματεύεται ένα νέο μοντέλο για αναζήτηση και πρόσβαση σε πολυμεσικό περιεχόμενο, βασισμένο στην θεώρηση μικρών κοινοτήτων χρηστών, το οποίο επιτρέπει την πλέον αποτελεσματική και κατανεμημένη υλοποίηση της αναζήτησης του περιεχομένου.

Πιο συγκεκριμένα, η διατριβή περιλαμβάνει τα επόμενα:

Στο Κεφάλαιο 2 εξετάζονται οι εφαρμογές κατανεμημένης αναζήτησης πολυμεσικών δεδομένων, τα οποία περιγράφονται με βάση το περιεχόμενό τους και τα μεταδεδομένα τους. Στο Κεφάλαιο 3 εξετάζεται η χρήση σημασιολογικής πληροφορίας και ειδικότερα οντολογιών για αναπαράσταση της γνώσης, επεκτείνοντάς την μελέτη σε ασαφείς οντολογίες και οντολογίες πολυμεσικών πληροφοριών. Στο Κεφάλαιο 4 μελετάται η έννοια της ομοιότητας και της συσχέτισης των πληροφοριών, παρουσιάζοντας κριτήρια ομοιότητας με βάση την οντολογική πληροφορία και ασαφείς σημασιολογικές σχέσεις. Στο Κεφάλαιο 5 προτείνεται ένα κριτήριο για ασαφή σύγκριση περιεχομένου εγγράφων σε μορφή XML. Στο Κεφάλαιο 6 μελετώνται μοντέλα σύνδεσης μικρού κόσμου και αναλύεται η έννοια των κοινοτήτων σε αυτά. Το Κεφάλαιο 7 περιγράφει την προτεινόμενη αρχιτεκτονική που βασίζεται στις κοινότητες του μικρού κόσμου και στα κριτήρια ασαφούς σημασιολογικής ομοιότητας που περιγράφηκαν προηγούμενα. Μια πειραματική μελέτη και αξιολόγηση της χρήσης της προτεινόμενης αρχιτεκτονικής που περιλαμβάνει και σύγκριση με βασικές μεθόδους συσταδοποίησης δίδεται στο

Κεφάλαιο 8, δείχνοντας την αποτελεσματικότητα της προτεινόμενης αρχιτεκτονικής. Συμπεράσματα και προτεινόμενες επεκτάσεις δίδονται στο Κεφάλαιο 9.

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ

Διανομή πολυμέσων

Acknowledgements

Firstly, I am grateful to the God for the good health and wellbeing that were necessary to complete my thesis.

I would also like to express my sincere gratitude to my advisor Prof. Kollias for the continuous support of my PhD study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my PhD study.

I would also like to thank my family: my wife, to my son and daughter, and to my father in law and mother in law for supporting me spiritually throughout writing this thesis and my life in general.

Contents

List of Figures

Chapter 1

Introduction

Over the last decades, a media availability explosion has emerged with respect to multimedia information. The consumers can simultaneously access widely digital media processing electronic devices. On the other hand, the resulting large volumes of multimedia and multimedia metadata increased the amount of volume that multimedia storage. This growth of multimedia volume has encouraged researchers to utilize distributed models for multimedia information management.

Moreover, the fault tolerance in distributed systems is advantageous because failure of one component does not result in a complete system failure. In distributed systems, if a component fails, in most cases only part of the system will be disabled, and most of it will active. In fact, this failure will just reduce system performance.

According to the aforementioned, a distributed approach is preferable due to increase of actual data storage and to avoidance of complete loss of system function in the presence of system component failures.

To design a distributed multimedia framework, some factors play an important role to achieve best performance. The first one is desired decentralization in both system function and data storage. Secondly, it is important to have admissible performance reduction when the system remains at least partially functional. Another essential factor is to have a flexible indexing technique that can apply to many different indexing scenarios. Moreover, an effective technique is needed to insert new elements into the index of the database grows. Finally, to retrieve data, we should have a search method and admissible generalization that can be applied to all multimedia sources (e.g., video, audio, etc.).

This work discusses a new underlying infrastructure to deliver exactly the above factors. It is based on the theory of small worlds, which uses the concept of peer similarity to build a network of multimedia data nodes containing information of similar peers who can, in general, exist anywhere in a network.

1. Multimedia Data Management Processes

Multimedia data management forms the basis of four processes: recording, mining, indexing, retrieval and replay. The recording of multimedia data can be done by using camcorders, microphones, scanners, electronic ink, and so on. After data collection, structured storing is one of the most important tasks in multimedia data management. In computer science, structured storing consists of data mining,

indexing and retrieval. Data mining is based on extracting useful and relevant information from the original data related to one or more media. Mined information can include metadata such as semantic and temporal information, summaries, structures, etc. Extracted information is usually related to indexing, which attaches one or more indexes to each multimedia document. These indexes can be used to classify multimedia based on content. To run a query of a user, the retrieval process matches the information of the indexed archive with the query information. Finally, the replay task is related to presentation of the data to users and to propose functionalities for interacting with the displayed multimedia documents.

2. General characteristics of a distributed multimedia retrieval system

A distributed multimedia retrieval system has some general characteristics as the following:

- Desirable decentralization in both system function and data storage.
- Admissible performance reduction when the system remains partially functional.
- Having a flexible indexing technique for the system designer to apply to many different indexing scenarios.
- An effective technique to insert new elements into the index of the database grows.
- A search method for specific data within the distributed index
- Admissible generalization so that the indexing method can be applied to types of data (e.g., video, audio, etc.).

3. Problem Description

In most of the proposed models for distributed multimedia, there is a broker element that is closer to centralized concept than the distribute concepts. When we want to use a distributed multimedia system, we need continuous data transfers over relatively long periods of time. As well as, there is a need for quality of service (QoS) management. A further requirement is to enable interactions between dispersed

groups of users. To reach all of them, use a model *without centralized element* is unavoidable.

Moreover, to retrieve content, most models must send a request to a list of servers that have been selected based on specific criteria and then choose the best answer. So when we have an infrastructure environment for content retrieval, a large number of messages is needed. Even in cases when the models use summarized requests, we still need to send the message to a lot of servers. There are some models that use small world networks to decrease number of messages but due to lack of attention to the community of the small world, they have not significantly reduced this number.

Since, increasing number of messages in a distributed system would provide a reduced performance, *decreasing the number of these messages* is very effective in improving the situation.

One of the important parameters in a distributed environment is system response time. Clearly, smaller times are more reasonable. This parameter specially plays an important role in search and retrieval. As the search or retrieval time gets smaller, the system performance will be better. So *a more appropriate framework would need a small retrieval time.*

4. Goals of This Work

Lately, some researchers have proposed to use small-world networks for distributed multimedia information. They have imitated social networks. As humans keep track of descriptions of their friends and acquaintances, every media object can store descriptions of the objects that have maximum similarity with itself. Community plays an important role in these small world networks. Whenever, the members of community are many and similar, the query time order will decrease.

In this work, we describe appropriate enhancements for community creation and usage in multimedia search and indexing approach. To improve the efficiency of this approach, we focus on designing distributed multimedia frameworks which reduce the complexity of information sharing and time searching on a large-scale network. The system relies on ontologies to describe the structure and semantics of the multimedia properties. We link nodes with fuzzy similar ontological interests, so as to make search more focused and efficient.

Our distributed multimedia framework is based on small-world networks decreasing centralization, and consequently increasing efficiency. In this context, each society of nodes operates as an independent server; so, inserting or retrieving of requests can be made within each community and this can *increase decentralization*.

To perform content retrieval in this framework, at first the members of a special community are locally examined. Then if nothing is found, other communities related to this community are examined, *decreasing average response time*. Since we only send the request to communities related to the first community, *the average number of messages is also reduced*.

5. Structure of thesis

In addition to this introduction, this manuscript is composed of seven chapters. Chapter 2 presents the distributed collection models and their properties. In chapter 3, related ontologies are studied, using a fuzzy morphological framework. Chapters 4 and 5 focus on content similarity and related procedures. Chapter 6 presents the properties of small-world networks and then a number of models for this type of network are provided. In chapter 7, the proposed framework is described in detail, presenting the creation of communities and the placement of nodes. Finally, evaluation and future work are presented in chapters 8 and 9.

Chapter 2

Distributed Multimedia Collection

1. Introduction

In order to deal with proliferation of multimedia content and metadata, some researchers have advised the use of distributed architectures where information can be stored on multiple remote servers in different data centers across various geographical locations. Of course, there are many reasons for having such a distributed multimedia service environment. These reasons include reliability of the services, geographical demands and company strategies. Usually, some data servers store all multimedia data. The web servers or presentation servers present the data to the end users' workstations. The efficiency principle to minimize resource consumption must be balanced in the development of a distributed multimedia system. At the same time, it must provide the most relevant results for a user's query in the shortest time.

On the other hand, relevant multimedia content is located in environments that consist of big number of machines with different capabilities, each hosting large multimedia collections. One of the key issues for the management and retrieval of relevant information in this environment is efficient content indexation. Indexing can be done according to a set of algorithms, which generate diverse and heterogeneous multimedia metadata and in which resource consuming is high. To design a distributed multimedia system, there are some choices that must be determined for indexing (Özsu and Valduriez 2011), such as performing using a fixed or a variable set of algorithms, algorithms executed over the entire multimedia collection, or, only over a filtered sub-collection, the whole set of algorithms being itself filtered, or not, before their effective execution, indexing in a distributed manner, on the same location as content, or in a centralized one, by transferring content to an indexation server (e.g., providing web services), a distributed or a centralized placement of the multimedia metadata.

In the following, we present the data collection and its management system. Then, we focus on a distributed model and on requirements for this model.

2. Data collection

Many systems commonly use data collection, which vary greatly in scale, and in sensing modalities. To acquire data, two questions are important: what data is collected, and what mechanism is used for this. The data can be collected by diverse sources such as sensors, radars, or simply being the output of computer applications.

On the other hand, the set of components and protocols which can serve all of them are not common. But there are some characteristics which are mostly common:

- Data acquisition: it refers to the inputs which are acquired from the environment. It can be done by some procedures that are presented to the user.
- Data transformation: In many cases, filtering, transforming, or other data processing may be needed.
- Data transmission: transmitting data to users or applications is a procedure met in most applications. During transmission, high volumes of raw data are transferred, with data volumes exceeding the capacity available for transmission.

3. Multimedia data

Multimedia data comprises of two parts. The first includes multimedia components, such as text, graphics, animation, sounds, and video. The second is multimedia metadata and semantic annotation of multimedia content that is a key-enabler for creating improved services on multimedia content. There are a various models for metadata which have different purpose, goal, scope and level of detail.

3.1. Multimedia Data Types

3.1.1. Text

The basic element of most multimedia titles is text. Two aspects of text are of concern to the developer. The first is the way in which text is presented to the user so that it should be easy to read and well designed. This point includes the font, colour, and text size. The second aspect is what lies behind the text; that is included in the interactive “links” that the user does not see which can lead to setting additional related information.

3.1.2. Images

Images are one of the most important components of multimedia applications. Usually, visual representations are conveying information which is more effective than text. Images can be generated in two ways: as bitmaps and as vector-drawn graphics. Bitmaps are used for photo-realistic images and are commonly used in multimedia titles. The software for creating them is readily available to most users.

Vector graphic images correspond to set of instructions for re-creating the image as an object including geometric elements, such as lines, circles, arcs, and angles. To store these instructions, relatively little storage and smaller graphic file sizes are needed. Indeed, the real advantage with vector graphics is that the same image may be resized, moved, or rotated while maintaining its original quality and proportions.

3.1.3. Video

Video overlay boards can capture video frames, as well as play them. Analogue and digital are two types of video that are used in multimedia. Animations and digital video include sequences of bitmapped graphic scenes (frames), rapidly played back. The most commonly used technique based on animation is to create a series of images that are displayed in rapid succession. This type of animation is called "frame-based" involving a different image, or frame, for each view and works like a filmstrip. Another common type of animation is "cast-based" animation, in which the background image remains the same, but some instructions are considered to move individual objects appearing on that background across the background. Both techniques involve the appearance of flat images moving on the screen. Accordingly, they are called 2-D animation.

3-D animation is another type of animation frequently used for virtual reality. Three-dimensional objects in this type are created using a mathematical model. To create 3-D animation, first a model must be created; it is followed by drawing the object in several views, using specified coordinates along x, y, and z-axes. Then it becomes more realistic by adding shading and "rendering" the image, which involves blending the background, model, light sources, and textures to make cohesive frame transitions.

3.1.4. Audio

Another important component of multimedia is sound, this may take two types: analogue and digital audio, which can be converted to each other by using an analogue-to-digital converter (ADC). Digitised sound is sampled sound. A sample of sound is taken and stored as digital information in every nth fraction of a second. The sampling rate expresses the number of the samples that is taken. On the other hand, the amount of information stored in each sample is considered as the sample size. The quality of digital sound depends on parameters, such as the number of channels

recorded, the sampling size and the sampling rate. Indeed digital data represents the instantaneous amplitude of a sound at discrete slices of time.

3.1.5. Composite Objects

Composite multimedia data can be created by combining basic multimedia data types and other composite multimedia data. A new type can be formed by mixing the types physically or logically. A new storage format is defined by a physical mix, where data such as audio and video intermix, creating i.e. compound objects. A logical mix defines a new data type, while retaining individual data types and storage formats, i.e. complex objects. Moreover, composite data can contain additional control information, which describes how the information should be rendered to the client.

3.2. Characteristics of Multimedia Data

3.2.1. Temporality

Temporal requirements can be embedded in some multimedia data types such as video, audio, and animation sequences, which have implications on their storage, manipulation and presentation. The temporal layout, orchestrating the data presentation is dictated by temporal structures. Basic temporal structures produce serial and parallel presentation of data. Moreover, the user can define presentations which associate a presentation time and duration (timeline model) with each multimedia object, eliminating the requirement for temporal structures.

3.2.2. Spatiality

Multimedia data have spatial constraints in terms of their content. Usually, there are some spatial relationships between individual objects in multimedia, such as an image or a video frame. These spatial relationships produce constraints which can be used for searching for an object. Virtual environments, or 3D, can be considered by spatial structures. Spatial constraints are used to control 3D-object movement and inter object spatial relationships. There are some special tools for graphical user interfaces to define spatial relationships.

3.2.3. Need for Storage Space and Fast Transmission

Big volumes of data are another aspect that characterises multimedia information. For instance, an uncompressed image of 1024:728 pixels at 24 bits per pixel requires a storage capacity of about 2 Mbytes. But the storage requirement could be reduced to about 0.1 Mbytes by using a 20:1 compression ratio. The potential for handling big volumes of data involved in multimedia information systems become apparent when one wants to run long movies or collections of movies.

3.2.4. Need for Content-Based Access

The limitations of textual descriptions of a multimedia experience and the massive information available from it, cause some problems for information retrieval. In addition, the limitations of textual descriptions pose the need for content-based access to multimedia information.

The potential information overload means that users may find it difficult to make precise requests during information retrieval. The limitations of textual descriptions also imply the need for content-based access to multimedia information.

3.2.5. Collaborative Support Environments

In collaborative environments, interaction of multimedia information involves long-duration operations, and sometimes, more than single users. To facilitate the provision of concurrency control algorithms, it is expected that most multimedia data are likely to be accessed in a read-only mode.

4. Multimedia Metadata

Metadata includes a content abstraction of the multimedia object that represents its content. This representation can be a textual description of the semantic content, or feature representations of perceptual content that are extracted automatically from the digitized object. Metadata is used to structure, enhance and enrich the information related to the object and also to effectively search and retrieve digital items. More detailed metadata needs more complex structure. Moreover, to reach a common understanding of metadata needs following the related standards.

Comprehensive review of the multimedia standards is given in (Stanchev 2009, Allasia and Gallo 2009). For instance, the adoption of MPEG-7 ("MPEG-7")

suggests description of metadata related to digital items, while MPEG-21 ("MPEG-21") helps in describing metadata related to a governed content. Using the above standards, improves the expressivity of the query language for multimedia items.

4.1. Types of Metadata

The large number of metadata elements can be classified in many ways that exist across standards (Troncy, Huet, and Schenk 2011). According to the intended application areas and use cases these elements are grouped in different ways according to existing standards. The classification used in the following has been proposed by (Troncy, Huet, and Schenk 2011) and has been compiled from the structure of various metadata standards.

- **Identification information**

This type of information usually includes IDs and titles related to the content (working titles, titles used for publishing, etc.).

- **Production information**

Metadata which is related to creation of content can be described by this information. Metadata such as location and time of capture, as well as, persons and organizations contributing to the production, belong to this type.

- **Rights information**

Rights information describes the rights holders of the content, as well as the permitted uses, modifications, distributions, etc. It can include elements ranging from a simple reference to a license to a very detailed description of the conditions and permitted use of each of the segments.

- **Publication information**

Information of this class describes previous use of content and related information.

- **Process-related information**

This group includes standards describing the history of production and post-production steps that occurred, such as information about captureing, digitization, encoding and editing steps in the workflow.

- **Content-related Information**

This type of metadata describes the structure of content and related information.

- **Context-related Information**

This class plays an important role when data includes semantic issues. For instance, the semantics of multimedia content is in many cases strongly influenced by the context, in which this content can be used and consumed.

- **Relational/enrichment information**

Metadata, which describe the links between the content and external data sources, such as other multimedia content or related textual sources, belong to this class.

5. Data Management in Distributed Data Collection Systems

5.1. Distributed Data Archiving

Data archiving plays an important role in any application which requires access to past data; for instance, in a pure streaming system, where persistent queries are posted and act as filters on arriving data, implementing a data archiving would be necessary to avoid missing anything that turns out to be important. Firstly, a central application which may be needed acts as an archival store for keeping information. But, if communication bandwidth is limited, especially in contrast to storage bandwidth local to the remote system, using a central application would be not efficient. In this case, storing data in a distributed fashion, and only retrieve it for the application when necessary will be more efficient. In a distributed application, data is naturally stored in various storage places.

5.2. Indexing and Querying

A query mechanism would be necessary when data is to be archived and be later retrieved by a specific application. There are some challenges in implementing such a query mechanism, due to the characteristics of the system in which it is implemented. The primary challenge refers to large numbers of nodes, each of stores which moderate amounts of data. In this case, efficiently locating query matches, and routing queries without exhaustively searching across nodes, are big challenges.

Another challenge refers to locating the requested information and forwarding a query to the appropriate node, especially when the amount of data archived on each

node is large, as is the case in some higher-speed applications, and the number of nodes is modest.

5.3. Analysis, Modeling and Prediction

A usual form of data analysis refers to indexing and querying data are organized according to their characteristics and are then divided according to whether or not these match an application-specific query. In such cases, prediction and compression can be used the modeling monitored phenomena. It may happen that some queries refer to the past, while others refer to future. In most cases, there is a large number of monitored variables, or features which for prediction. Then, it is necessary to choose significant features to incorporate into the model, while ignoring irrelevant ones. In cases where these features do not significantly change over time, they are ready to follow by human experts. But, in other cases, where these features change over time, machine learning techniques have been proposed for model derivation and updating.

6. Multimedia Data Management Processes

Multimedia data management forms the basis of the following processes: recording, mining, indexing, retrieving and replaying. The recording of multimedia data can be done with camcorders, microphones, scanners, electronic ink, etc. Following data collection, structured storing is one of the most important tasks in multimedia data management, including data mining, indexing and retrieval. Data mining is based on extracting useful and relevant information from the original data related to one or more media. Mining information provides metadata, such as semantic and temporal information, summaries, structures, etc. Extracted information is usually related to indexing, which attaches one or more indices to each multimedia document. These indexes can be used to classify multimedia based on content. When running a user query, the retrieval process matches the information of the indexed archive to the query information. Finally, the replay task is related to presentation of data to users, proposing functionalities for interacting with the displayed multimedia documents.

6.1. Requirements for Multimedia Data Management

A multimedia data management system provides a suitable environment for using and managing multimedia data information. Hence, it must include functions

for data definition and creation, data retrieval, data access and organization, data independence, privacy, integration, integrity control, version control and concurrency support when applied to various multimedia data types. The functional requirements can be classified into two categories: data representation requirements and data manipulation requirements.

6.2. Data Representation Requirements

6.2.1. Support for Generalization/Specialization Hierarchy

Support for generalization/specification hierarchy is a major requirement imposed on multimedia applications that is used to define the type, subtype, and instance relationships between various entities. Moreover, support for generalization/specification hierarchy facilitates schema evolution. For example, any document, which is constructed according to a new schema, is also an instance of the old schema.

6.2.2. Attribute Specification

Specifying properties of objects should be supported. These properties are termed attributes. For example, support for specifying the properties of a document, such as its title, author, font size, etc. should be provided. These properties are called attributes of the document.

6.2.3. Specification of Operations

The ability to specify the operations that can be performed on multimedia data is another requirement. These operations include changing the contents of multimedia data, retrieving the contents of data, etc.

6.2.4. Support for Composite Objects

The support for composite objects is a major requirement for modeling multimedia applications. For example, a document may be composed of front matter, body, and back matter.

6.2.5. Object Sharing

Object sharing is especially necessary for multimedia data as the amount of storage space required to store a document might be quite large. Sharing is the capability for different multimedia data to share parts of their content.

6.2.6. Data Independence

Data and management functions must be separated from application programs.

6.3. Data Manipulation Requirements

6.3.1. Integration and Integrity Control

Integration expresses that there is no need to duplicate data during different program invocations requiring it. A computing model that has proven suitable for this is the client-server one. In addition, consistency of data is ensured by making integrity control during a transaction. This is done through constraints imposed on the transactions.

6.3.2. Concurrency Control

Multimedia data consistency is checked by concurrency control through rules, which impose some form of execution order on concurrent transactions. A transaction is defined as a sequence of instructions which are executed, either completely, or not at all, while also defining the appropriate granularity.

6.3.3. Persistence

The ability of data objects to survive through different transactions and program invocations is termed persistence. The simplest method to achieve persistence is to store multimedia files in an operating system framework, e.g. a cloud. Usually, multimedia data are classified into persistent and transient data. Moreover, persistent data is stored after transaction updating, to keep persistency. Transient data are used, only during program, or transaction execution and are removed afterwards.

Privacy

Unauthorized access and modifications of stored data should be restricted.

6.3.4. Query Support

The query mechanism is an important requirement for multimedia data. This mechanism selects a subset of the data objects according to the user's description. Various attributes, possibly keyword-based, or content-oriented, are issues that are usually used in the query.

6.4. Efficient Capture, Access and Presentation of Multimedia Data

Essentially, capturing and presentation of various types of multimedia data must be supported. In addition, ability to manage the indices, to allocate and de-allocate pages on disk, to move the pages to and from disk should be provided.

6.5. Data Availability

The fact that a data object can be retrieved from alternative storage devices, as well as from different portions of the same device is usually known as data availability. It means that, despite of some storage medium failure, the requested object can be retrieved, since the same object can be also found in another repository. Data availability is usually done by data replication, this may cause two drawbacks. First, replication in a high degree consumes the storage capacity. Second, replicas `consistency must be kept. This means that any update of the object has to be mirrored on each of the replicas. Updating can be very time consuming, due to required high access times.

7. Managing and Querying Distributed Multimedia Metadata

To improve services based on multimedia content, and metadata, and semantic annotation of multimedia content is an essential issue that has to be acquired in real time and possibly stored in different, heterogeneous locations. Moreover, the enormous amount of media increases the usage of decentralized infrastructures and also promotes their adoption. This happens in peer-to-peer (P2P) content sharing systems that minimize the impact of a single point of failure, fostering scalability, reliability and efficiency. Therefore, it is advisable to use distributed architectures where information can be stored on multiple remote servers; the later compute queries in parallel, more efficiently than centralized metadata repositories. However, in these architectures, queries are sent to each server, which is not truly efficient because it ca be possible to know in advance that only some servers contain the desired information.

7.1. Requirements

As a whole, the key issues required for metadata management are the following:

- Insertion/removal

The system should have the ability to insert and delete metadata. The insertion is done when the content is first introduced into the system at the content servers. In addition, removal of metadata is executed to avoid the inconsistency of keeping metadata of content that does not exist.

- Appropriate access

Components of the system that make use of metadata should have appropriate access to the metadata. Therefore, the metadata manager should provide a well-defined interface, through which users of metadata can access it in a transparent manner.

- Updating of metadata

When there is a change on the content, its metadata should also be updated, this can affect the characteristics of the content recorded in the metadata.

- Generation of metadata

The metadata manager should have the ability to create new content, as a version of the original content.

- Caching of metadata

To provide fast access to metadata, caching of metadata at the proxies plays an important role. It is done to increase the efficiency of access to the metadata.

- Extracting missing features

In some cases, there is not all information required to serve client requests, or to decide on the process of adaptation. Therefore, the metadata manager should employ appropriate feature extraction to further enrich metadata. This feature is used to extract the missing information.

- Metadata standard conversion

When metadata different from the one that is used in the system, is inserted, it needs to be converted to the appropriate form. The appropriate transformation is needed to be applied on the metadata in order to make it usable by the system.

8. Generic multimedia framework

Figure 1 outlines a generic architecture for distributed multimedia content.

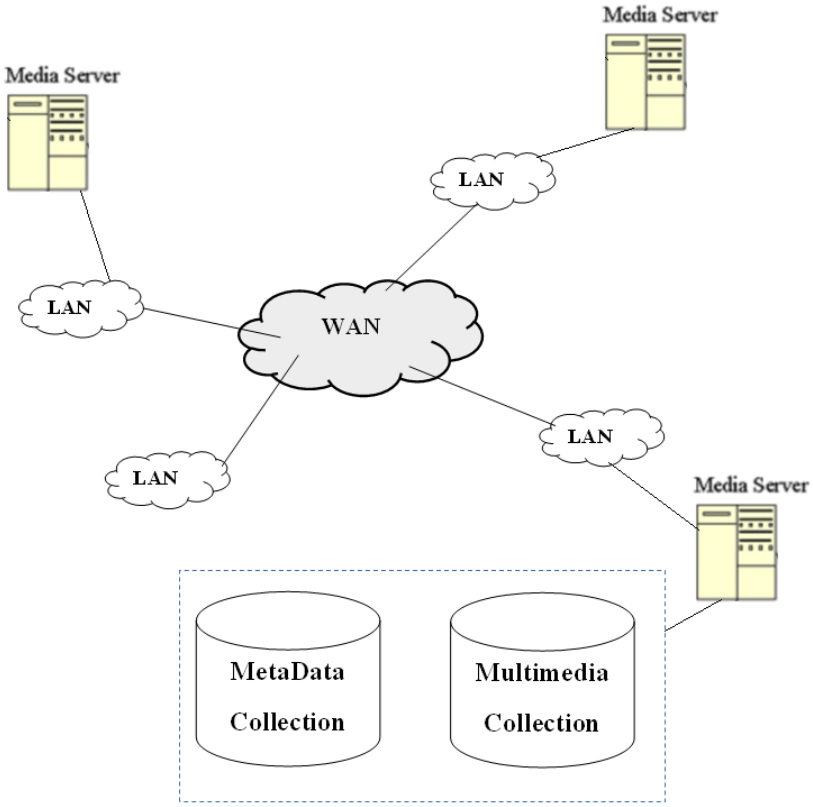


Figure 1. A generic architecture for distributed multimedia content

This architecture consists of several parts:

- Media server contains multimedia files and their characteristics. It includes metadata and media items, such as a piece of text, an image, a video, or an audio snippet.
- A multimedia collection has several pieces of multimedia content. Each multimedia collection is stored on a dedicated server that is used to acquire the remote site information.
- Content metadata includes information about media characteristics.
- A metadata collection includes all content metadata describing the objects of the multimedia collection.

9. General characteristics of a distributed multimedia retrieval system

A distributed multimedia retrieval system has some general characteristics, such as the following:

- Desirable decentralization in both system function and data storage.

- Admissible performance reduction when the system remains partially functional.
- Providing the system designer with a flexible indexing technique that can be applied to many different indexing scenarios.
- An effective technique to insert new elements into the index of the database.
- A search method for specific data within the distributed index
- Admissible generalization so that the indexing method can be applied to various types of data (e.g., video, audio, etc.).

This work discusses a new underlying infrastructure to deliver the performance listed above. It is based on the theory of small worlds, which uses the concept of peer similarity to build a network of multimedia data nodes containing information of similar peers who can, in general, exist anywhere on a network.

References

Allasia, Walter, and Francesco Gallo. 2009. “Indexing and Retrieval of Multimedia Metadata on a Secure DHT.” *Informatica* 33: 85–100.

“MPEG-21.” *ISO/IEC 21000 - Information Technology Multimedia Framework*. http://www.chiariglione.org/mpeg/.

“MPEG-7.” *ISO/IEC 15938 - Information Technology Multimedia Content Description Interfaces*. http://www.chiariglione.org/mpeg/.

Özsu, MT, and P Valduriez. 2011. *Principles of Distributed Database Systems*. http://books.google.com/books?hl=en&lr=&id=TOBaLQMuNV4C&oi=fnd&pg=PR7&dq=Principles+of+Distributed+Database+Systems&ots=LpDgiFXR06&sig=zRGZ1lZ-Kmjjjs17WO-R4uGyxhk.

Stanchev, Peter L. 2009. “Multimedia Standards.” In *Advances in Multimedia*.

Troncy, Raphael, Benoit Huet, and Simon Schenk. 2011. *Multimedia Semantics Metadata, Analysis and Interaction*.

Chapter 3

Multimedia Ontologies

1. Introduction

In most computer science fields, research on semantic relations, and ontologies has become increasingly widespread. In the past, this term has been limited to the philosophical area. But it plays a specific role in Artificial and computation Intelligence, Computational Linguistics, and Data science. In particular, ontologies have been effectively used in knowledge engineering (Bouillet et al. 2007), knowledge representation (Studer, Grimm, and Abecker 2007), qualitative modeling(Han and Stoffel 2011), language engineering (Maniraj and Sivakumar 2010), database design (LePendu et al. 2008), information modeling (Hughes, Crichton, and Mattmann 2009), information integration (Wache and Voegele 2001)(Fonseca, Davis, and Câmara 2003), object-oriented analysis (Evermann and Wand 2005)(Siricharoen 2007), information retrieval and extraction (Styltsvin 2006)(Kara et al. 2012), knowledge management and organization (Jurisica, Mylopoulos, and Yu 1999)(Sureephong and Chakpitak 2008), agent-based systems design (Isern, Sànchez, and Moreno 2007)(Sheldon 2003).

In some cases, the use of ontologies implies familiar activities like conceptual analysis and domain modeling, which are executed in order to provide better access to related information. Definitions of terms and concepts, as well as relationships between them can enable better processing in data related.

2. Definition of ontologies

Neches et al. (Neches, Fikes, Finin, & Gruber, 1991) define ontologies to comprise the basic terms and relations of the vocabulary of a thematic area, as well as the rules for combining terms and relations to define extensions of this vocabulary.

Wielinga and Alberts report two more definitions taken from the literature:

- An (AI-) ontology is a theory of what entities can exist in the mind of a knowledgeable agent (Wielinga and Schreiber 1993).
- An ontology for a body of knowledge concerning a particular task or domain describes a taxonomy of concepts for that task, or domain, that define the semantic interpretation of the knowledge (Alberts 1994).

Ontologies belong to the knowledge level that it is exactly the degree of such dependence which determines the reusability and therefore the *value* of knowledge.
There is another definition of ontologies proposed by Tom Gruber reported in (Uschold and Gruminger 1996):

Ontologies are agreements about shared conceptualizations. Shared conceptualizations include conceptual frameworks for modelling domain knowledge; content-specific protocols for communication among inter-operating agents; and agreements about the representation of particular domain theories. In the knowledge sharing context, ontologies are specified in the form of definitions of representational vocabulary. A very simple case would be a type hierarchy, specifying classes and their subsumption relationships. Relational database schemata also serve as ontologies by specifying the relations that can exist in some shared database and the integrity constraints that must hold for them.

According to this definition, ontologies and conceptualizations are kept clearly distinct. Therefore, an ontology is not a *specification* of a conceptualization, but it is an agreement *about* a conceptualization. Therefore, as suggested in (Guarino and Giaretta 1995), we can have different degrees of detail in this agreement depending on the purpose of the ontology:

An ontology is an explicit, partial account of a conceptualization.

In (Gutiérrez et al. 2001) an ontology is defined as "a formal, explicit specification of a shared conceptualization". *Conceptualization* means an abstract model of events with relevant concepts. *Explicit* expresses the type of concepts identified, and the constraints on their use. *Formal* declares the machine readable property. *Shared* reflects that ontology is based on consensual knowledge so that the group of view of the target phenomenon is accepted.

In (Sanroma 2010) an ontology is also defined as an explicit specification of a conceptualization. Ontologies are designed for being used in applications that need to process the content of information, as well as, to reason about it, instead of just presenting information to humans. They permit greater machine interpretability of content than that supported by XML, RDF and RDF Schema (RDF-S), by providing additional vocabulary along with a formal semantics.

From a structural point of view, an ontology is composed by disjoint sets of *concepts, relations, attributes* and *data types* (Sanroma 2010). *Concepts* are sets of real world entities with common features. *Relations* are binary associations between concepts. There exist inter-concept relations, which are common to any domain and domain-dependent associations. *Attributes* represent quantitative and qualitative features of particular concepts, which take values in a given scale defined by the *data type*. Concepts are classes organized in one or several taxonomies, linked by means of transitive *is-a* relationships (taxonomical relationships). Multiple inheritances (i.e. the fact that a concept may have several hierarchical ancestors or subsumers) are also supported. Binary relations can be defined between concepts. In those cases, the concept for the origin of the relation represents the domain and those in the destination, the *range*. Those relationships may fulfill properties such as symmetry or transitivity. Some standard languages have been designed to codify ontologies. They are usually declarative languages, based on, either first-order logic, or on description logic. Some examples are KIF, RDF (Resource Description Framework), KL-ONE, DAML+OIL and OWL (Web Ontology Language).

3. Ontology Classification

The basis issue in ontology construction is ontological commitments. Since they come from different sources and have different flavours, different types of ontologies are distinguishable. On the lowest level of complexity, a vocabulary of a knowledge base is defined. In addition, we can define a domain ontology by defining a set of types and basic relations such as sub/super class relations, including the extensional description of a domain vocabulary, typology and class hierarchy or lattice. In the following, we summarize different sources of ontological commitments and the resulting ontologies, as is proposed in (Wielinga and Schreiber 1993):

Level 1*: domain terminology*

Level 2*: representational commitments (e.g. class, relation, etc.)*

Level 3*: domain-model oriented commitments (e.g. structural models, anatomical models)*

Level 4*: task-type oriented commitments (e.g. constraints)*

Level 5*: method oriented commitments (e.g. fix knowledge)*

Different classes of knowledge bases, with different scope, generality and reusability can be defined by using different ontologies, with different generality and partitioning of the application knowledge base.

Another categorisation of ontologies can be made according to their subject of conceptualisation. An interesting classification was proposed by Guarino (Guarino 1998), who classified types of ontologies according to their level of dependence on a particular task or point of view:

- **Top-level ontologies**: describe general concepts like space, time, event, which are independent of a particular problem or domain. It is called upper ontologies or foundational ontologies, to describe very abstract and general concepts that can be shared across many domains and applications. Moreover, they use philosophical notions to describe top-level concepts for all things that exist, such as "physical object" or "abstract object". Their generality does not permit their direct use in applications. The best examples for top-level ontologies are the Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) (Gangemi, Guarino, and Masolo 2002) and the Suggested Upper Merged Ontology (SUMO) (Niles and Pease 2001).

- **Domain-ontologies**: describe the vocabulary related to a generic domain by specializing the concepts introduced in the top-level ontology. There is a lot of examples of this type of ontologies in e-commerce: UNSPSC, NAICS, the biomedicine SNOMED CT, MESH, etc. The knowledge in these types of ontologies is captured within a specific domain of discourse, such as medicine or geography, or the knowledge about a particular task, such as diagnosing or configuring. According to this point, the scope of these types is much narrower and more specific than top-level ontologies. To develop domain ontologies, there is much work that has be done in medicine, genetics, geographic and environment information, tourism, as well as in cultural heritage and museum exhibition.

- **Task ontologies**: describe the vocabulary related to a generic task, or activity, by specializing top-level ontologies. A class of tasks, such as configuration, requires additional knowledge elements. For instance, knowledge about constraints on components is generally needed in configuration tasks. Therefore, additional ontological commitments are introduced committed to a certain class of tasks. In the

ideal case, the notions in a task ontology are described with reference to a domain, in a neutral manner, while the conceptualisation in a domain ontology is kept strictly task-independent. Task ontologies have been suggested for scheduling and planning, monitoring in a scientific domain, intelligent computer-based tutoring, missile tracking, execution of clinical guidelines.

- **Application ontologies**: the concepts are often determined with reference to roles played by domain entities while performing a certain activity, like a replaceable unit or a spare component. In addition, they have limitation on reusing and dependency on the particular scope and requirements of a specific application. A specific vocabulary can be provided by an application ontology to describe a certain task in a particular application context. Therefore, concepts depending both on a particular domain and a task are typically described by application ontologies.

4. Fuzzy ontologies

Some ontological applications use vague and imprecise information. The semantic-based applications in the Semantic Web such as e-commerce, knowledge management and web portals are some of these instances. The conceptual formalism that is commonly found in many application domains is not sufficient to represent imprecise information. Similarly, the need of giving a different interpretation according to context emerges.

Incorporating fuzzy logic into ontologies is a possible solution to handle uncertain data. The aim of fuzzy set theory introduced by L. A. Zadeh (L. Zadeh 1965) is to describe vague concepts through a generalized notion of sets, according to which an object may belong to a certain degree (typically a real number in the interval [0,1]) to a set. In literature, a first application has been made in the context of medical document retrieval (Parry 2004), adding a degree of membership to all terms in the ontology, to overcome the *overloading* problem. Another proposal is an extension of the domain ontology with the fuzzy concept (Lee, Jian, and Huang 2005), for Chinese news summarization.

Fuzzy knowledge plays an important role in many domains that face a huge amount of imprecise and vague knowledge and information, such as text mining, multimedia information management, medical informatics, machine learning, and

human natural language processing. Fuzzy ontologies contain fuzzy concepts and fuzzy memberships. Lee et al. proposed an algorithm to create fuzzy ontologies and applied it to news summarization (Lee, Jian, and Huang 2005). Abulaish et al. proposed a fuzzy ontology framework in which a concept descriptor is represented as a fuzzy relation which encodes the degree of a property value using a fuzzy membership function (Abulaish and Dey 2007).

A fuzzy ontology is an ontology extended with fuzzy values assigned to entities and relations of the ontology.

Definition 1: *Let U be the universe of discourse, $U = \{u_1, u_2, \ldots, u_n\}$, where $u_i \in U$ is an object of U and let A be a fuzzy set in U, then the fuzzy set A can be represented as:*

$$A=\{(u_1, f_A(u_1)), (u_2, f_A(u_2)),\ldots,(u_n, f_A(u_n))\} \quad \textbf{(1)}$$

where f_A, $f_A : U \rightarrow [0; 1]$, is the membership function of the fuzzy set A; $f_A(u_i)$ indicates the degree of membership of u_i in A.

Below, we give the definition of a fuzzy ontology presented in (Silvia Calegari and Ciucci 2006).

Definition 2: *A* fuzzy ontology *is an ontology extended with fuzzy values assigned through the two functions*

$$\textbf{g :(Concepts} \cup \textbf{Instances)} \times \textbf{(Properties} \cup \textbf{Property\_value)} \rightarrow \textbf{[0, 1]} \quad \textbf{(2)}$$

$$\boldsymbol{h: Concepts \cup Instances \rightarrow [0, 1]} \quad \textbf{(3)}$$

*w*here g is defined based on the relations and h is defined based on the concepts of the ontology.

4.1. Applications of Fuzzy Ontologies

A goal of using a fuzzy ontology is the direct handling of concept modifiers into the knowledge domain, which has the effect of altering the fuzzy value of a property (L. a. Zadeh 1972). During last years, a large number of different approaches have been proposed about the problem of efficient query refinement. Extending the set of concepts already present in a query with others, which can be derived from an ontology, is a usual practice. Using a fuzzy ontology, a threshold value can be

established to extend queries with instances of concepts that satisfy the chosen value (Silvia Calegari and Ciucci 2006). This value is defined by the domain expert.

Another usage of fuzzy value associated to concepts has been in the context of medical document retrieval, in which such a value is used to limit the problems due to overloading of a concept in an ontology (Parry 2004). In addition, it causes reducing the number of documents found, by hiding those that do not fulfil the request of the user.

4.2. Fuzzy Concept Network

One of the most important actions which appears in fuzzy ontologies is the updating of the fuzzy values given to the concepts, or to the relations set by the expert during ontology definition. Every time that a query is performed such as updating is needed. Many techniques have been proposed for updating (Raghavan and Wong 1986)(Xu and Croft 2000). When a query is performed the dynamical behavior of a fuzzy ontology is examined, through the introduction of new concepts. A technique for updating and initializing the fuzzy values has been proposed in (S. Calegari and Loregian 2006). It determines a semantic correlation among the entities that is used in a query.

Definition: A correlation is a binary and symmetric relation between entities. It is characterized by a fuzzy value: corr : $O \times O \rightarrow [0,1]$, where the set $O = \{o_1, o_2, \ldots, o_n\}$ is the set of the entities contained in the ontology.

By using this definition, the degree of relevance for entities is determined. Moreover, for each existent correlation, there is an updating formula.
To integrate the correlation values into the fuzzy ontology, when a query is executed new correlations are created, or updated, altering their weights. The expert assigns a fuzzy weight to the concepts and to correlations, during the definition of the ontological domain.

A concept network comprises n nodes and a set of directed links (S Calegari and Farina 2007). Each node represents a concept, or a document and each link is labelled with a real number in [0, 1]. In the following, there is a definition that has been reported in (S Calegari and Farina 2007).

Definition: A Fuzzy Concept Network (FCN) is a complete weighted graph N_f = *{O, F, m}, where O denotes the set of the ontology entities.* The edges among the nodes are described by the function F: O × O → [0, 1]. If F (o_i, o_j) = 0 then the entities are considered uncorrelated. In particular F: = corr. Each node o_i is characterized by a membership value defined by the function m: O → [0, 1], which determines the importance of the entity by its own in the ontology. By definition F(o_i, o_i) = m(o_i).

4.3. A Fuzzy Ontology Framework with Imprecise Knowledge

Imprecision in an ontology is primarily detected in relations and properties. More specifically, an ontology relation connecting two concept instances is regarded as imprecise when this connection may, in certain domains, contexts and application scenarios, be considered to contain some vague characteristic. The existence and meaning of this vagueness depends on the intended meaning of the relation and the vast range of possible meanings. An exhaustive mapping between relation and imprecision types is very difficult.

Nevertheless, there are two broad kinds of ontology relations in which imprecision may be assigned a comprehensible meaning: hierarchical relations and associative ones. Hierarchies organize entities in a tree-structure by defining a partial ordering of them according to some relation. Common types of hierarchical relations are taxonomies and metrologies. A taxonomical relation is the well-known is-a relation which associates an entity of a certain type to another entity of a more general type. This association can be regarded as imprecise as, while it denotes that the meaning of an entity is more specific than that of another one, the actual level of this specialization, is unclear. Therefore, imprecision in taxonomy reflects the absence of information on how "close" the meaning of the child entity to that of the parent is.

A metrological relation, on the other hand, is based on the is-part-of relation, which associates parts with their wholes. In general, part-of relations are not by default transitive. Therefore, a metrological relation can be defined as a part-of relation that is considered transitive in a given context. The nature and possible meaning of imprecision in a metrological relation may be derived from the fact that the part is always "less" than the whole. This "less", when not determined explicitly in a qualitative, or quantitative, way, is the imprecise aspect of the relation. What this imprecision actually stands for can be determined by the fact that, in general, a part-of

relation attempts to take into account the degree of differentiation of the parts with respect to the whole.

Finally, associative relations relate entities in a non-hierarchical way. Imprecision in such a relation reflects the lack of accurate information on the strength of the association between entities, either due to the relation's inherent vagueness, or to the absence of accurate information on some relation's characteristics. As far as properties are concerned, these may be regarded as imprecise in two cases:

- When the property may relate its concept instances to literal values in an imprecise way.
- When the literal values to which the property relates its concept's instances may be expressed in an imprecise way.

Imprecision in the first case is similar to one of associative ontology relations, as the only difference between a property and a relation is that the latter relates instances to each other, instead of instances and literal values. Consider, for example, the property *category* of the concept Book that takes as values strings denoting subject categories. The property practically links book instances to relevant specific categories. Since relevance is an inherently vague notion, the property may be regarded as imprecise. There is a lot of research on fuzzy ontologies (Silvia Calegari and Sanchez 2007)(Change, Huang, and Sandnes 2007). In (Silvia Calegari and Sanchez 2007) it is shown how a Fuzzy Ontology based approach can improve semantic document retrieval. After formally defining a Fuzzy Knowledge Base, it is a special type of new non-taxonomic fuzzy relationships are introduced, called semantic correlations. These correlations, first assigned by experts, are updated after querying, or when a document has been inserted into a database. To represent fuzzy knowledge more effectively, Zhai (Zhai et al. 2008) presents a new series of fuzzy ontology models that consist of fuzzy domain ontologies and fuzzy linguistic variable ontologies, considering semantic relationships of concepts, including set relation, order relation, equivalence relation and semantic association relation. Chang (Change, Huang, and Sandnes 2007) has proposed a multimedia information retrieval system, with a fuzzy information ontology strategy, to further improve the usability of mobile handheld devices. (Sanchez and Yamanoi 2006) have presented several connections between Fuzzy Logic, Semantic Web, and its components (Ontologies, Description Logics). Lately, Alexopoulos (Alexopoulos et al. 2010) has proposed a framework for

fuzzy ontologies with imprecise knowledge. In this model, a Fuzzy Ontology is defined as a tuple O $\{C, I, FHR, FAR, FP, FLV, FVP\}$ where:

- C is a set of concepts.
- I is a set of instances. Each instance belongs to at least one concept.
- FHR and FAR are sets of fuzzy hierarchical and fuzzy associative relations. Each fuzzy relation $fr \in \{FHR \cup FAR\}$ is a function $I \rightarrow [0,1]$.
- FP is a set of fuzzy properties. Each fuzzy property $fp \in FP$ is a function $I \rightarrow F(X)$, $F(X)$ being the set of all fuzzy sets in the universe of discourse X.
- FLV is a set of fuzzy linguistic variables. Each $flv \in FLV$ is a tuple $\{u, T, X, m\}$ in which u is the name of the variable, T is the set of linguistic terms of u that refer to a base variable whose values range over a universal set X and m is a semantic rule that assigns to each linguistic term $t \in T$ its meaning $m(t)$ which is a fuzzy set on X.

FVP is a set of fuzzy valued properties. Each fuzzy valued property $fvp \in FVP$ is a function $I \rightarrow T$ where T is the set of the linguistic terms of a fuzzy linguistic variable $flv \in FLV$.

5. Multimedia ontologies

Multimedia content can be expressed using ontologies. Harmonizing the seemingly isolated conceptual and perceptual worlds is necessary for semantic processing of multimedia data. In this sense, concepts include abstractions of perceptual observations that have an interesting consequence. These abstractions comprise multimedia metadata; semantic annotation of multimedia content is very important to archive, retrieve, and manage the multimedia content.

Several metadata models and metadata standards have been proposed, in which the scope and level of detail is different. Saathoff (Simou, Saathoff, and Dasiopoulou 2006) has suggested the Multimedia Metadata Ontology (M3O) that is a framework in which both semantic and low-level metadata are integrated. This model is specially useful for representing rich metadata. It uses Semantic Web technologies to provide the infrastructure for representing high-level semantic annotation, as well as annotation with low-level features, extracted from the multimedia content. Common vocabularies representing shared knowledge within a specific domain can be defined with ontologies which use final list of terms and concepts (Noy and McGuinness

2001). Using several vocabularies for semantic annotation of multimedia content is not rich enough and suitable for describing it. Accordingly, development of, extended, multimedia enriched ontologies, also known as multimedia ontologies is a response to this necessity.

In the following, we list some tasks which are related to generation of multimedia ontologies reported in (Sjekavica, Obradović, and Gledec 2013):

- *Annotation – tagging or labeling multimedia content;*
- *Analysis – ontology-driven semantic analysis of multimedia content;*
- *Retrieval – context-based image retrieval;*
- *Personalization – recommendation and filtering of multimedia content based on user preferences;*
- *Algorithms and processes control – modeling multimedia procedures and processes;*
- *Reasoning – personalization and retrieval for creating autonomous content applications.*

In this section we provide an overview of the most common ontologies that are developed for use in the multimedia domain and for annotation of multimedia content.

5.1. COMM

The Core Ontology for Multimedia (COMM) (Arndt et al. 2009) is an ontology implemented in OWL DL. COMM has been proposed to enable and facilitate multimedia annotation. It uses DOLCE as its underlying foundational ontology and it has been built according to the re-engineering MPEG-7 standard.

There are two main Ontology Design Patterns (ODP) which are used to design COMM: Descriptions and Situations (DnS) and Ontology for Information Object (OIO). Most parts of the MPEG-7 standard is covered by the ontology. In addition, COMM uses the same naming convention for all MPEG-7 descriptors formalized as in the MPEG-7 standard. On the other hand, describing multimedia analysis steps, something that is not possible in MPEG-7, is allowed by the explicit representation of algorithms in the multimedia patterns. Modularization is done for the ontology, including the core module and the modules specialized on each media type, i.e. visual, text, media, localization and data type modules; this minimizes execution overhead when processing data. Moreover, a Java Application Programming Interface

(API) is provided by COMM which enables an MPEG-7 class interface for the construction of metadata at runtime, also simplifying creation of multimedia content of annotation.

5.2. Ontology for Media Resources 1.0

This ontology comprises a set of properties describing media resources (core vocabulary) and their mapping to a set of metadata formats which describe media resources published on the Web(W. Lee, W. Bailer, T. Bürger, J.P. Evain, V. Malaisé, T. Michel, F. Sasaki, and Söderberg, F. Stegmaier 2012). The W3C Media Annotations Working Group has developed the ontology for media resources 1.0. The mapping is used to provide an interoperable set of metadata which enable different applications to share and reuse these metadata. The ontology can unify mappings to common media formats, by providing a large set of mappings to 18 multimedia metadata formats (Dublin Core, MPEG-7, IPTC, Exif, OGG, etc.) and six multimedia container formats (3GP, FLV, QuickTime, MP4, OGG, WebM). The annotation properties comprise terms such as identifier, title, creator, date, location, description, keyword, rating, copyright, target audience, format, and etc. Since the set of properties is equivalence to existing formats, a mapping table is considered to define one-way mappings between the ontology's properties and the metadata fields of other standards.

5.3. Multimedia Metadata Ontology (M3O)

M3O is an ontology to annotate structured multimedia content on the web and to unblock its semantics by making it machine-readable and machine-understandable. It has been proposed by Saathoff and Scherp (Saathoff and Scherp 2010). M3O provides a generic modeling framework to integrate existing multimedia metadata formats and metadata standards. Since it is based on Semantic web technologies, it can be easily integrated with today's presentation formats like SMIL, SVG or Flash. M3O integrates and represents metadata and data structures that underlie the existing approaches, rather than replacing any of the existing models.

There are five principal requirements which are provided by M3O's patterns:

- Identification of resource.
- Separation between information objects and realizations.

- Annotation of information objects and information realizations.
- Decomposition of information objects and information realizations.
- Representation of provenance information.

In M3O, data structures are represented in a form of patterns based on the foundational ontology DOLCE+DnS Ultralight (DUL) that provide these requirements. DnS, Information, Realization, and Data Value patterns are reused in the M3O. Moreover, Annotation and Decomposition patterns are provided by M3O and used by the ontology to distinguish between the information object and its realization. The representation of high-level semantic annotation with background knowledge and the annotation with low-level features extracted from multimedia content are supported by M3O. It has been aligned with COMM, Ontology for Media Resources and EXIF.

5.4. Large-Scale Concept Ontology for Multimedia (LSCOM)

LSCOM was designed by Smith (Smith et al. 2006) to satisfy multiple criteria of utility, coverage, feasibility, and observability in diverse broadcast news video data sets. It defines a formal vocabulary that includes more than 2.000 concepts for the annotation and retrieval of broadcast news video. The LSCOM considers the significant interest in news video as an important multimedia source of information. In LSCOM, concepts are related to objects, activities and events, scenes and locations, people, programs, and graphics.

6. Ontology Usages

Ontologies have been developed for many purposes (Cross, 2004), so as to provide reusability and information sharing in software systems. In information query, ontologies may be used to provide metadata to find more relevant sources. Global standardized ontologies are developed by researchers in areas such as e-Commerce or geographical information systems.

To enable multimedia content to be discovered and exploited by services, agents and applications, it needs to be semantically described. Generating descriptions of multimedia content is inherently problematic because of the volume and complexity of the data, its multidimensional nature and the potentially high subjectivity of human-generated descriptions.

Ontologies are formal descriptions of the abstraction of a domain that uses to expressing multimedia content. The key to semantic processing of multimedia data lays in harmonizing the seemingly isolated conceptual and perceptual worlds. Concepts are formed in human minds through a complex refinement process of personal experiences. Raw data go through a process of refinement to result in mental models. The models are further abstracted over a large number of observations, to give rise to concepts, which are labeled with linguistic constructs to facilitate communication. The fact that concepts are abstractions of perceptual observations has an interesting consequence. A concept gives rise to the expectation of some perceptible media properties on its embodiment in a multimedia artifact.

References

Abulaish, Muhammad Abulaish Muhammad, and Lipika Dey Lipika Dey. 2007. *A Fuzzy Ontology Generation Framework for Handling Uncertainties and Nonuniformity in Domain Knowledge Description*. *2007 International Conference on Computing Theory and Applications ICCTA07*. doi:10.1109/ICCTA.2007.6.

Alberts, LK. 1994. "YMIR: a Sharable Ontology for the Formal Representation of Engineering Design Knowledge." In *Formal Design Methods for CAD, IFIP Transactions*. http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:YMIR+:+A+SHARABLE+ONTOLOGY+FOR+THE+FORMAL+REPRESENTATION+OF+ENGINEERING-DESIGN+KNOWLEDGE#0.

Alexopoulos, Panos, Manolis Wallace, Konstantinos Kafentzis, and Dimitris Askounis. 2010. "Utilizing Imprecise Knowledge in Ontology-based CBR Systems by Means of Fuzzy Algebra." *International Journal of Fuzzy Systems* 12 (1): 1–14.

Arndt, Richard, R Troncy, Steffen Staab, and Lynda Hardman. 2009. "COMM: A Core Ontology for MultimediaAnnotation." In *Handbook on Ontologies*. http://link.springer.com/chapter/10.1007/978-3-540-92673-3_18.

Bouillet, Eric, Mark Feblowitz, Zhen Liu, Anand Ranganathan, and Anton Riabov. 2007. "A Knowledge Engineering and Planning Framework Based on OWL Ontologies." *International Competition on Knowledge Engineering for Planning and Scheduling (ICKEPS)*. http://choices.cs.uiuc.edu/~ranganat/Pubs/ICKEPS07_CamRdy.pdf.

Calegari, S, and F Farina. 2007. "Fuzzy Ontologies and Scale-free Networks Analysis." *International Journal of Computer Science & Application* 4 (2). http://www.tmrfindia.org/ijcsa/v8i10.pdf.

Calegari, S., and M. Loregian. 2006. "Using Dynamic Fuzzy Ontologies to Understand Creative Environments." In *Felxible Query Answering System*.

Calegari, Silvia, and Davide Ciucci. 2006. "INTEGRATING FUZZY LOGIC IN ONTOLOGIES." *International Conference on Enterprise Information Systems (ICEIS)*: 66–73.

Calegari, Silvia, and Elie Sanchez. 2007. "A Fuzzy Ontology-approach to Improve Semantic Information Retrieval." *On Uncertainty Reasoning for the Semantic*. http://pdf.aminer.org/000/251/792/an_integration_of_fuzzy_and_two_valued_logics_on_natural.pdf.

Change, Tsun-Wei, Yo-Ping Huang, and Frode-Eika Sandnes. 2007. "A Fuzzy Ontology Strategy for Multimedia Data Management." *BCS SGAI Journal*

Expert Update: 2086–2091. doi:10.1109/ICSMC.2007.4413738. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4413738.

Evermann, Joerg, and Yair Wand. 2005. "Ontology Based Object-oriented Domain Modelling: Fundamental Concepts." *Requirements Engineering* 10 (2) (January 13): 146–160. doi:10.1007/s00766-004-0208-2. http://link.springer.com/10.1007/s00766-004-0208-2.

Fonseca, Frederico, Clodoveu Davis, and G Câmara. 2003. "Bridging Ontologies and Conceptual Schemas in Geographic Information Integration." *Geoinformatica* 7: 355–378. http://link.springer.com/article/10.1023/A:1025573406389.

Gangemi, Aldo, Nicola Guarino, and Claudio Masolo. 2002. "Sweetening Ontologies with DOLCE." *The 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web*: 166–181. http://link.springer.com/chapter/10.1007/3-540-45810-7_18.

Guarino, Nicola. 1998. "Formal Ontology in Information Systems." *FOIS Conference* (June): 3–15. http://www.openontology.net/download/dot/ontopage/sub_SmWe2001a.pdf.

Guarino, Nicola, and Pierdaniele Giaretta. 1995. "Ontologies and Knowledge Bases Towards a Terminological Clarification." In *Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing*. http://books.google.com/books?hl=en&lr=&id=8xiJLQSDGfEC&oi=fnd&pg=PA25&dq=Ontologies+and+Knowledge+Bases+Towards+a+Terminological+Clarification&ots=hlafmw8Zal&sig=keDSDStZT-QG9mnRbhYSpX3Qlek.

Gutiérrez, Miguel Esteban, Asunción Gómez-pérez, Oscar Muñoz García, and Boris Villazón Terrazas. 2001. "Ontology Access in Grids with WS-DAIOnt and the RDF (S) Realization." *Design*: 4–7.

Han, Dong, and Kilian Stoffel. 2011. "Ontology Based Qualitative Case Studies for Sustainability Research." *Proceedings of the AI for an Intelligent Planet on - AIIP '11*: 1–8. doi:10.1145/2018316.2018322. http://dl.acm.org/citation.cfm?doid=2018316.2018322.

Hughes, J. Steven, Daniel J. Crichton, and Chris a. Mattmann. 2009. "Ontology-based Information Model Development for Science Information Reuse and Integration." *2009 IEEE International Conference on Information Reuse & Integration* (August): 79–84. doi:10.1109/IRI.2009.5211603. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5211603.

Isern, David, D Sànchez, and A Moreno. 2007. "An Ontology-driven Agent-based Clinical Guideline Execution Engine." *Artificial Intelligence in Medicine*: 49–53. http://link.springer.com/chapter/10.1007/978-3-540-73599-1_6.

Jurisica, Igor, John Mylopoulos, and Eric Yu. 1999. "Using Ontologies for Knowledge Management: An Information Systems Perspective." *Society For Information*. ftp://www-vhost.cs.toronto.edu/pub/eric/eric/asis99.pdf.

Kara, S, Ö Alan, O Sabuncu, and S Akpınar. 2012. “An Ontology-based Retrieval System Using Semantic Indexing.” *Information Systems* (July). http://www.sciencedirect.com/science/article/pii/S030643791100113X.

Lee, Chang-Shing, Zhi-Wei Jian, and Lin-Kai Huang. 2005. “A Fuzzy Ontology and Its Application to News Summarization.” *IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics : a Publication of the IEEE Systems, Man, and Cybernetics Society* 35 (5) (October): 859–80. http://www.ncbi.nlm.nih.gov/pubmed/16240764.

LePendu, P, Dejing Dou, GA Frishkoff, and Jiawei Rong. 2008. “Ontology Database: A New Method for Semantic Modeling and an Application to Brainwave Data.” *Scientific and Statistical Database Management(SSDBM)*: 313–330. http://link.springer.com/chapter/10.1007/978-3-540-69497-7_21.

Maniraj, V., and R. Sivakumar. 2010. “Ontology Languages – A Review.” *International Journal of Computer Theory and Engineering* 2 (6): 887–891. doi:10.7763/IJCTE.2010.V2.257. http://www.ijcte.org/show-33-297-1.html.

Niles, Ian, and Adam Pease. 2001. “Towards a Standard Upper Ontology.” *Proceedings of the International Conference on Formal Ontology in Information Systems(FOIS)*: 2–9. doi:10.1145/505168.505170. http://portal.acm.org/citation.cfm?doid=505168.505170.

Noy, NF, and DL McGuinness. 2001. “Ontology Development 101: A Guide to Creating Your First Ontology.” http://liris.cnrs.fr/~amille/enseignements/Ecole_Centrale/What is an ontology and why we need it.htm.

Parry, David. 2004. “A Fuzzy Ontology for Medical Document Retrieval.” In *The Australian Workshop on DataMining and Web Intelligence (DMWI)*. Vol. 32. http://dl.acm.org/citation.cfm?id=976458.

Raghavan, V. V., and S. K. M. Wong. 1986. “A Critical Analysis of Vector Space Model for Retrieval.” *The American Society for Information Science* 37: 279–287.

Sanchez, Elie, and Takahiro Yamanoi. 2006. “Fuzzy Ontologies for the Semantic Web.” *Flexible Query Answering Systems*: 691–699. http://www.springerlink.com/index/9Q70413417016520.pdf.

Sanroma, Montserrat Batet. 2010. “ONTOLOGY-BASED ONTOLOGY SEMANTIC EMANTIC CLUSTERING.”

Sheldon, FT. 2003. “An Ontology-based Software Agent System Case Study.” *Information Technology: Coding and Computing*: 500 – 506. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1197580.

Simou, N, C Saathoff, and S Dasiopoulou. 2006. “An Ontology Infrastructure for Multimedia Reasoning.” *... Content Processing and ...*. http://www.springerlink.com/index/55716106005r5228.pdf.

Siricharoen, WV. 2007. “Ontologies and Object Models in Object Oriented Software Engineering.” *IAENG International Journal of Computer Science*. http://www.iaeng.org/IJCS/issues_v33/issue_1/IJCS_33_1_4.pdf.

Sjekavica, Tomo, Ines Obradović, and Gordan Gledec. 2013. “Ontologies for Multimedia Annotation: An Overview.” In *7th European Computing Conference (ECC),* 123–129. http://www.wseas.us/e-library/conferences/2013/Paris/ECCS/ECCS-17.pdf.

Studer, R, S Grimm, and A Abecker. 2007. “Knowledge Representation and Ontologies Logic, Ontologies and Semantic Web Labguages.” In *Semantic Web Services: Concepts, Technologies, and Applications*. http://www.computer.org/csdl/mags/ex/2001/02/x2046.pdf.

Styltsvin, Henrik Bulskov. 2006. “Ontology-based Information Retrieval.” *... of the 14th International Conference on Information ...*. http://www.tuke.sk/paralicj/papers/IIS03.pdf.

Sureephong, Pradorn, and Nopasit Chakpitak. 2008. “An Ontology-based Knowledge Management System for Industry Clusters.” In *Global Design to Gain a Competitive Edge,* 1–10. http://link.springer.com/chapter/10.1007/978-1-84800-239-5_33.

Uschold, Mike, and Michael Gruminger. 1996. “Ontologies: Prnciples, Methods and Applications.” *Knowledge Engineering Review* 11.

W. Lee, W. Bailer, T. Bürger, P.A. Champin, J. J.P. Evain, V. Malaisé, T. Michel, F. Sasaki, and J. Strassner Söderberg, F. Stegmaier. 2012. “Ontology for Media Resources 1.0.” *W3C Recommendation*.

Wache, H, and T Voegele. 2001. “Ontology-based Integration of Information-a Survey of Existing Approaches.” *Ontologies and Information Sharing*: 108–117. http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Ontology-Based+Integration+of+Information+—+A+Survey+of+Existing+Approaches#0.

Wielinga, BJ, and AT Schreiber. 1993. “Reusable and Sharable Knowledge Bases: a European Perspective.” In *First International Conference on Building and Sharing of Very Large-Scaled Knowledge Bases*. http://hcs.science.uva.nl/usr/wielinga/postscript/Wielinga:93e.ps.

Xu, Jinxi, and W. Bruce Croft. 2000. “Improving the Effectiveness of Information Retrieval with Local Context Analysis.” *ACM Transactions on Information Systems* 18 (1) (January 1): 79–112. doi:10.1145/333135.333138. http://portal.acm.org/citation.cfm?doid=333135.333138.

Zadeh, L. a. 1972. “A Fuzzy-Set-Theoretic Interpretation of Linguistic Hedges.” *Journal of Cybernetics* 2 (3) (January): 4–34. doi:10.1080/01969727208542910. http://www.tandfonline.com/doi/abs/10.1080/01969727208542910.

Zadeh, LA. 1965. “Fuzzy Set.” *Information and Control*.

Zhai, Jun, Yiduo Liang, Jiatao Jiang, and Yi Yu. 2008. “Fuzzy Ontology Models Based on Fuzzy Linguistic Variable for Knowledge Management and Information Retrieval.” *Intelligent Information Processing IV* 288: 58–67. http://www.springerlink.com/index/j71124717520hj43.pdf.

Chapter 4

Fuzzy Semantic Similarity

1. Introduction

Similarity plays a fundamental role in theories of knowledge and behavior. It serves as an organizing principle by which individuals classify object concepts, and make generalizations. Indeed, the concept of similarity is ubiquitous in psychological theory. It underlies the accounts of stimulus and response generalization in learning. It is employed to explain errors in memory and pattern recognition, and it is central to the analysis of connotative meaning. Many similarity models have been proposed in the literature. Some models ((Cross, 2004)(Rodriguez & Egenhofer, 2003)) measure the distance between ontological concepts. Other ones ((Zhai, Liang, Jiang, & Yu, 2008)(Jiang, 1997)) compute Corpus Statistics and Lexical Taxonomy.

Similar or dissimilar data appear in different ones: rating of pairs, sorting of objects, communality between associations, errors of substitution, correlation between occurrences. Analysis of such data attempts to explain the observed similarity relations and to capture the underlying structure of the objects under study (Tversky, 2004). In computational linguistics, the computation of semantic similarity between concepts has been a very active trend. It expresses how much words extracted from documents or textual descriptions are alike. Semantically, similarity is usually computed by taxonomical relations between concepts. For example, bronchitis and flu are similar because both are disorders of the respiratory system. However, words can be related in other non-taxonomical ways (e.g. diuretics help to treat hypertension). In those more general cases, semantics are discussed.

The assessment of similarity (from a domain independent point of view) has many direct applications such as, word-sense disambiguation (Hirst & St-Onge, 1998), information retrieval (Rodriguez & Egenhofer, 2003) and ontology learning (Sanchez, Batet, & Valls, 2010). In the biomedical domain, the performance of Information Retrieval tasks can be improved by semantic similarity computation (Sanchez et al., 2010). Similarity assessment is generally based on the estimation of semantic evidence observed in one or several knowledge or information sources.

In the research literature, there are three different terms: similarity, distance and relatedness. To distinguish between similarity and relatedness, (Cross, 2004) illustrates that similarity is a special case of relatedness. The terms such as *car* and *gasoline* appear to be more closely related than the terms like *car* and *bicycle***,** even though *car* and *bicycle* are more similar. This one example shows one kind of

relatedness based on a functional relationship such as 'car uses gasoline." There are numerous other kinds of semantic relatedness based on the type of relationship between concepts such as subsumption (e.g., vehicle-car) and meronymy (e.g. car-wheel). One of the most natural approaches (Ou, West, Lazarescu, & Clay, 2005) to determine semantic similarity in an ontology is to use its graphical representation and measure the distance between nodes corresponding to words or concepts being compared. The number of edges in the shortest path between two concepts measures the distance between them. The shorter this distance is the more similar the concepts semantically are.

2. Formal Definition of Similarity

There is not a direct formula to define the similarity measure, but it can be derived from a set of assumptions about similarity. To provide a formal definition, Lin (Lin, 1998) considered some intuitions about similarity.

Intuition 1: The similarity between A and B is related to their commonality. The more commonality they share, the more similar they are.

Intuition 2: The similarity between A and B is related to the differences between them. The more differences they have, the less similar they are.

Intuition 3: The maximum similarity between A and B is reached when A and B are identical, no matter how much commonality they share.

According to these intuitions, Lin has proposed this formula to compute similarity:

$$sim(A,B) = \frac{log\ P(common\ \ (A,B))}{log\ P(description\ \ (A,B))} \qquad \textbf{(1)}$$

where common (A, B) is a proposition that states the commonalities between A and B, description (A, B) is a proposition that describes what A and B are. P(x) is the probability that a randomly selected object belongs to x.

3. String Similarity

A string similarity measure quantifies similarity between two text strings for approximate string matching, or comparison. One can define a similarity measure

between two strings and rank the words in the word list in descending order of their similarity to the given word. The similarity measure implies that words derived from the same root as the given one should appear early in the ranking. There are three string similarity measures (Lin, 1998). The first one is defined as follows:

$$sim_{edit}(x,y) = \frac{1}{1+editDist\ (x,y)} \quad \textbf{(2)}$$

where editDist (x,y) is the minimum number of character insertion and deletion operations needed to transform one string to the other.

The second similarity measure is based on the number of different trigrams in the two strings:

$$sim_{tri}(x,y) = \frac{1}{1+|tri\,(x)|+|tri\,(y)|-2\times|tri\,(x)\cap tri\,(y)|} \quad \textbf{(3)}$$

where tri(x) is the set of trigram in x. For example, tri (operation) = {ope,per,era,rat,ati,tio,ion}.

The third similarity measure is based on formal definition of similarity under the assumption that the probability of a trigram occurring in a word is independent of other trigrams in the word:

$$sim(x,y) = \frac{2\times\sum_{t\in tri\,(x)\cap tri\,(y)} log\ P(t)}{\sum_{t\in tri\,(x)} log\ P(t)+\sum_{t\in tri\,(y)} log\ P(t)} \quad \textbf{(4)}$$

4. Ontology-based Similarity

As stated before, a formal specification of a shared conceptualization is provided by ontologies (Gomez-Perez, Fernández-López, & Corcho, 2004). Ontologies represent a very reliable and structured knowledge source, being machine readable and constructed from the consensus of a community of users or domain experts.

In this section, we cover approaches, completely or partially, relying on ontologies, to compute semantic similarity/relatedness.

4.1. Edge counting-based measures

According to these measures, it is the geometrical model of semantic pointers provided by ontologies that is exploited. Indeed, ontologies can be considered as a directed graph in which concepts are interrelated mainly by way of taxonomic (is-a) and, in some cases, non-taxonomic links (Sanchez et al., 2010). Input terms are mapped to ontological concepts by their textual labels. Evaluating the minimum *Path Length* connecting their corresponding ontological nodes by means of is-a links is a straightforward method to calculate the similarity between terms (Resnik, 1995).

One problem that is posed by these measures is that they sometimes have a difficult interpretation according to computational linguistics. So, several variations and improvements of this edge counting approach have been proposed. From a certain point of view, in addition to this absolute distance between terms, Wu and Palmer (Wu & Palmer, 1994) proposed that the relative depth in the taxonomy of concepts corresponding to the evaluated terms is an important dimension, because concept specializations become less distinct, as long as they are recursively refined. Therefore, equally distant pairs of concepts belonging to an upper level of the taxonomy should be considered less similar than those belonging to a lower level. Wu and Palmer's measure computes the number of is-a links (N1 and N2) from each term to their Least Common Subsumer (LCS) (*i.e.*, the most concrete taxonomical ancestor who subsumes both terms) and also the number of is-a links from the LCS to the root (N3) of the ontology.

$$sim_{Wu\ \&\ Palmer}\ (c_1, c_2) = \frac{2 \times N_3}{N_1 + N_2 + 2 \times N_3} \quad \textbf{(5)}$$

Al-Mubaid and Nguyen (Al-Mubaid & Nguyen, 2009) proposed a measure that is computed by the combination of the minimum *path length* and the *taxonomical depth.* First, the clusters for each of the branches in the hierarchy are defined with respect to the root node. Then, the common specificity of two terms is measured by subtracting the depth of their LCS from the depth D_c of the cluster.

$$CSpec(a, b) = D_c\ - depth(LCS(a, b)) \quad \textbf{(6)}$$

It must be noted that this method considers that the pairs of concepts found in lower levels are more similar than the ones found in higher levels, following Wu and Palmer's approach. So, the proposed distance measure (*sem*) is defined as follows:

$$dis_{sem}(a,b) = \log((min_{\forall i}|path_i(a,b)| - 1)^{\alpha} \times (CSpec)^{\beta} + k) \qquad \textbf{(7)}$$

where $\alpha>0$ and $\beta>0$ are the contribution factors of the path length and the common specific features and *k* is a constant.

Al-Mubaid `s approach is often considered as a *hybrid* approach in the literature, because it considers the combination of several structural characteristics (such as path length, depth and local density) and assigns weights to balance the contribution from each component to the final similarity value. The disadvantage of this approach is that it depends on the empirical tuning of weights according to the ontology and input data, although its accuracy for a concrete scenario is higher than that of basic edge-counting measures.

Simplicity is the main advantage of the presented measures. They only rely on the geometrical model of an input ontology, whose evaluation requires a low computational cost. In general, any ontology-based measure depends on the degree of completeness, homogeneity and coverage of the semantic links represented in the ontology (Sanchez et al., 2010). So, they require rich and consistent ontologies like WordNet to work properly (Sanroma, 2010). General massive ontologies such as WordNet, with a relatively homogeneous distribution of semantic links and good inter-domain coverage, are the ideal environments to apply those measures. Wide and detailed ontologies such as WordNet incorporate multiple taxonomical inheritances, resulting in several taxonomical paths which are not taken into account, because this approach only considers the shortest path between concept pairs, even if it is for the concrete case of taxonomic path-based measures.

The disadvantage of this approach is that other features also influencing the concept semantics, such as the number and distribution of common and non-common taxonomical ancestors, are not considered. On the other hand, the problem of path-based measures relies on the notion that all links in the taxonomy represent a uniform distance. It must be noted that semantic distance between concept

specializations/generalizations in ontologies practically depend on the degree of granularity and taxonomic detail implemented by the knowledge engineer.

4.2. Feature-based measures

These methods are regarding the fact that taxonomical links in an ontology do not necessary represent uniform distances. They consider the degree of likeness between sets of features that are built to describe the two terms compared. Therefore, feature-based approaches assess similarity between concepts as a function of their properties. This is based on Tversky's model of similarity, which, derived from set theory, takes into account common and non-common features of compared terms. Common features tend to increase similarity and non-common ones tend to diminish it (Tversky, 1977).

In Rodriguez and Egenhofer (Rodriguez & Egenhofer, 2003) , similarity is computed as the weighted sum of similarities between synsets, meronyms and neighbor concepts (those linked via semantic pointers) of evaluated terms.

$$\mathrm{sim}_{\mathrm{r}\,\&\,e}(\mathrm{a},\mathrm{b}) = \mathrm{w}.\,\mathrm{S}_{\mathrm{synsets}}\ (\mathrm{a},\mathrm{b}) + \mathrm{u}.\,\mathrm{S}_{\mathrm{meronyms}}\ (\mathrm{a},\mathrm{b}) + \mathrm{v}.\,\mathrm{S}_{\mathrm{neighborhods}}\ (\mathrm{a},\mathrm{b}) \quad (8)$$

Where *w*, *u* and *v* weight the contribution of each component, which depends on the characteristics of the ontology. Meronyms refer to matching of concepts via part-of relationships.

In Tversky (Tversky, 1977) concepts and their neighbors (according to semantic pointers) are represented by synsets. The similarity is computed as:

$$\mathrm{sim}_{\mathrm{tve}}(\mathrm{a},\mathrm{b}) = \frac{|\mathrm{A}\cap\mathrm{B}|}{|\mathrm{A}\cap\mathrm{B}|+\gamma(\mathrm{a},\mathrm{b})|\mathrm{A}\backslash\mathrm{B}|+\big(1-\gamma(\mathrm{a},\mathrm{b})\big)|\mathrm{B}\backslash\mathrm{A}|} \quad (9)$$

where *A*, *B* are the synsets for concepts corresponding to *a* and *b*, *A\B* is the set of terms in *A* but not in *B* and *B\A* the set of terms in *B* but not in *A*. The term |A| is the cardinality of the set A. Finally, γ *(a, b)* is computed as a function of the depth of *a* and *b* in the taxonomy as follows :

$$\gamma(a,b) = \begin{cases} \frac{\text{depth (a)}}{\text{depth (a)+depth (b)}}, \ \text{depth(a)} \leq \text{depth(b)} \\ 1 - \frac{\text{depth (a)}}{\text{depth (a)+depth (b)}}, \ \text{depth(a)} > depth(b) \end{cases} \quad \textbf{(10)}$$

In Petrakis *et al*., (Petrakis, Varelas, Hliaoutakis, & Raftopoulou, 2006) a feature-based function called *Xsimilarity* relies on the matching between synsets and concept descriptions extracted from WordNet (*i.e.*, words extracted by parsing term definitions). They consider that two terms are similar if the synsets and descriptions of their concepts and those of the concepts in their neighborhood (following semantic links) are lexically similar. The similarity function is expressed as follows:

$$\boldsymbol{S}im(a,b) = \begin{cases} 1, \ if \ S_{synsets}\,(a,b) > 0; \\ \max\{S_{neig\,hbor\,hoods}\,(a,b), S_{descriptions}\ (a,b)\}, if\ (a,b) = 0. \end{cases} \quad \textbf{(11)}$$

where $S_{neighborhoods}$ is calculated as follows:

$$S_{neig\,hbor\,hoods}\,(a,b) = \max \frac{|A_i \cap B_i|}{|A_i \cap B_i|} \quad \textbf{(12)}$$

with each different semantic relation type (*i.e.*, *is-a* and *part-of* in WordNet) being computed separately (*i* denotes the relation type) and the maximum (joining all the synsets of all concepts up to the root of each hierarchy) being taken. $S_{descriptions}$and $S_{synsets}$are both computed as:

$$\text{S(a, b)} = \frac{|\text{A} \cap \text{B}|}{|\text{A} \cup \text{B}|} \quad \textbf{(13)}$$

where *A* and *B* denote synsets or descriptions sets for terms *a* and *b*.

One problem of this approach is dependency on the weighting parameters that balance the contribution of each feature, because of lack of clear criteria to assign values. In all cases, those parameters should be tuned according to the nature of the ontology and even to the evaluated terms. This hampers their applicability as a general purpose solution. Only the proposal by Petrakis (Petrakis et al., 2006) does not depend on weighting parameters, as the maximum similarity provided by each single feature is taken. Even though this adapts the behavior of the measure to the

characteristics of the ontology and to the knowledge modeling, the contribution of other features is omitted, if only the maximum value is taken at each time.

4.3. Information Content-based measures

In addition to acknowledging some of the limitations of edge-counting approaches, Resnik (Resnik, 1995) proposed to complement the taxonomical knowledge provided by an ontology with a measure of the information distribution of concepts computed from corpora. He exploited the notion of Information Content (IC) associating to each concept of the taxonomy the corresponding probability of appearance, which is computed from their occurrences in a given corpus. Concretely, the IC of a term *a* is computed from the inverse of its probability of occurrence, *p*(*a*) (13). In this manner, infrequent words are considered more informative than common ones.

$$IC(a) = -\log P(a) \quad \textbf{(14)}$$

According to Resnik, semantic similarity depends on the amount of shared information between two terms, a dimension which is represented by *their LCS* in the ontology. The more specific the subsumer is (higher IC), the more similar the terms are, as they share *more information*. Two terms are maximally dissimilar if an LCS does not exist (*i.e.*, in terms of edge-counting, it would not be possible to find a path connecting them). Otherwise, their similarity is computed as the IC of the LCS (14).

$$sim_{res}(a,b) = IC(LCS(a,b)) \quad \textbf{(15)}$$

One of the problems of Resnik's proposal is that any pair of terms having the same LCS results in exactly the same semantic similarity. The measure proposed by Jiang and Conrath(Jiang, 1997) is based on quantifying, in some way, the length of the taxonomical links as the difference between the IC of a concept and its subsumer. When comparing term pairs, they compute their distance by subtracting the sum of the IC of each term alone from the IC of their LCS.

$$dis_{j\&c}(a,b) = \big(IC(a) + IC(b)\big) - 2 \times sim_{res}(a,b) \quad \textbf{(16)}$$

The coherence of the IC computation with respect to the taxonomical structure is an aspect that should be ensured in order to maintain consistency of similarity computation.

It is important to note that IC-based measures need, in order to behave properly, that the probability of appearance *p* monotonically increases as one moves up in the taxonomy. This will ensure that the subsumer's IC is lower than its specializations. This is achieved by computing *p(a),* as the probability of encountering *a* and any *specializations* of *a* in the given corpus. In practice, each individual occurrence of any word in the corpus is counted as an occurrence of each taxonomic class containing it (Pirró, 2009). So, this approach forces the recursive computation of all the appearances of the subsumed terms to obtain the IC of the subsume

$$p(a) = \frac{\sum_{w \in \mathrm{W(a)}} count\ (w)}{N} \quad \textbf{(17)}$$

where $W(a)$ is the set of words in the corpus whose senses are subsumed by a, and N is the total number of corpus words that are present in the taxonomy.

5. Semantic Similarity in a Taxonomy

According to the standard argumentation of information theory (Ross, 1976), the information content of a concept c can be quantified as the negative of its log likelihood, −log p(c). Notice that in this way, informativeness decreases as probability increases. Therefore, the more abstract a concept, is the lower its information concept gets. Consequently, information content will be zero for a unique top concept.

Resnik (Resnik, 1995) proposed a measure of semantic similarity in an is-a taxonomy, based on the notion of information content. The semantic similarity between two classes is not an issue about the class of them. For example, when we want to compute similarity of rivers and ditches, we are not going to compare the set of rivers with the set of ditches. Instead, we are comparing a generic river and a generic ditch. Therefore, if we want to determine similarity of two classes C and C′, we must notice x∈ C and x′ ∈ C′ are independent because the selection of a generic C is not related to the selection of a generic C′. The amount of information contained in x and x′ is computed by this formula (Lin, 1998):

$$-\log P(C) - \log P(C') \quad \textbf{(18)}$$

where P(C) and P (C′) are probabilities that an object belongs to C and C′. If C_{com} is the most specific class that subsumes both C and C′, the similarity between x∈ C and x′∈ C′ is defined as follows:

$$sim(x + x') = \frac{2 \times \log P(Ccom)}{\log P(C) + \log P(C')} \quad \textbf{(19)}$$

5.1. Fuzzy Semantic Similarity

The semantic similarity measure may be generalized to a fuzzy semantic similarity measure, if the weights of the relation link are replaced by membership degrees indicating the strength of the relationships between the parent and child concepts.

5.1.1. Fuzzy Sets and Weak Fuzzy Similarity Relation

The theory of Fuzzy Sets proposed by Zadeh (Zadeh, 1965) has achieved a great success in various fields. Song (Song, Ma, Liu, Lian, & Zhang, 2007) showed that the degree of similarity between two concepts in an ontology is neither necessarily symmetric, nor necessarily transitive. He has proposed a weak fuzzy similarity relation, as a generalization of a fuzzy similarity relation.

Definition 3: A weak fuzzy similarity relation: A weak fuzzy similarity relation is a mapping,
*sim :U×U ->[0,1]***,** such that for *x, y, z*∈ *U, it possesses*
(a) Reflexivity: *sim(x, x) = 1*
(b) Conditional symmetry: *if sim(x, y) >0 then sim(y, x)>0*
(c) Conditional transitivity: *if sim(x, y) ≥ sim(y, x)>0* and *sim(y,z)≥sim(z,y)>0* then *sim(x,z) ≥sim(z,x)*

The weak fuzzy similarity relation can be used to measure similarity between two ontological concepts, because conceptual similarity has a property of reflexivity. In

addition, ontologies structure has a property of conditional symmetry and conditional transitive.

5.1.2. Fuzzy Similarity Measure

To compute similarity in anhierarchy ontology, we should determine the way that ancestral concepts which are relative closely to a concept be identified. Moreover, it must be described how shared information of a pair concept in the ontology is specified.

An ontology is usually shown by graph *G=(V, E)* where the nodes of the graph representing concepts. Each edge expresses a kind of semantic relation between two nodes that represent these two concepts. *V* is set of nodes and *E* is set of edges. *E* includes two sub-sets, which are a set of hierarchy component and a set of non-hierarchy component respectively. As can been seen in Figure 1, solid edges show hierarchy structure, and broken ones are non-hierarchy structure.

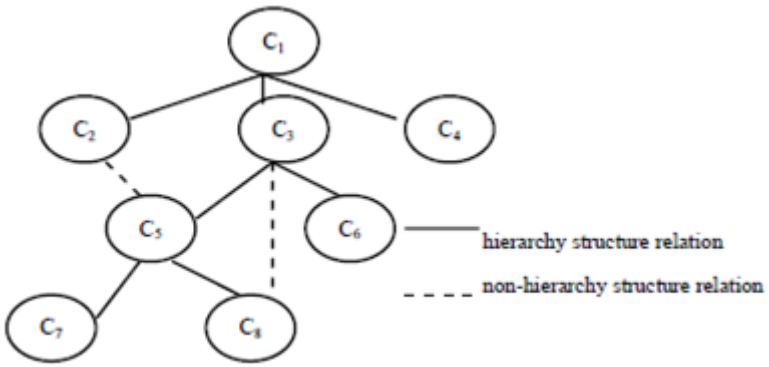


Figure 1: A fragment of an abstract ontology

Different semantic relations can be attained by assigning various weights according to edges, semantic type. Adjacency matrix is used to express the hierarchical structure of the ontology; α_1 and α_2 are expressed as immediate inclusion relation and IS-A relation respectively; T_{ij}' is used to express weights between concept *I* and concept *j;* Adjacency matrix *T'* is defined:

$$T = \begin{cases} 1 & if\ i = j \\ \alpha_1 & if\ i \neq j\ and\ (i,j) \in Inclusion \\ \boldsymbol{\alpha_2} & \boldsymbol{if\ i \neq j\ and\ (i,j) \in IS\_A} \\ \mathbf{0} & \boldsymbol{otherwise} \end{cases} \quad \textbf{(20)}$$

To consider the density of edges in weight computing, Sussna` s equation (Sussna, 1993) is used as follow:.

$$wt(c_1, c_2) = \frac{wt\,(c_1 \rightarrow_r c_2) + wt\,(c_2 \rightarrow_r c_1)}{2d} \quad \textbf{(21)}$$

$$wt(x \rightarrow_r y) = max_r - \frac{max_r - min_r}{n_r(x)} \quad \textbf{(22)}$$

In (Song et al., 2007) the non-hierarchical components of an ontology are represented by an additional adjacency matrix. In addition, different semantic relations correspond to different similarity factors. Assume that we have *k* different semantic relations R_1, R_2, ... ,R_k. Then, let β_1, β_2, ... β_k represent the corresponding similarity factors (weights). A non-hierarchical semantic relation adjacency matrix *S* is defined:

$$S_{ij} = \begin{cases} 1 & if\ i = j \\ \beta_i & if\ i \neq j\ and\ (i,j) \in R_i \\ 0 & otherwise \end{cases} \quad \textbf{(23)}$$

Therefore, to represent all semantic relations in graph *G*, we have $G = T \cup S$.

5.2. Ontology-based Fuzzy similarity with respect to Imprecise knowledge

In our framework, the different types of attributes a case may have are three, namely Fuzzy Property Attributes, Fuzzy Valued Property Attributes and Fuzzy Relation Attributes. From these, only the second type has been the subject of other relevant works and in particular of efforts in the area of Fuzzy case based reasoning (CBR) and Fuzzy Decision Making (Kahraman, 2007). Indeed, the common characteristic of these efforts is that they all use attributes with fuzzy values and a fuzzy pattern matcher for case similarity assessment. Yet, these approaches are incomplete since, as we have explained above, imprecision in ontology-based CBR may be manifested in other ways as well. For that, in this work, we focus on defining a case similarity framework that covers all of the aforementioned types of fuzziness.

Our approach towards such a definition includes the development of methods for assessing, for each fuzzy attribute type, the similarity between the values these may take. The key characteristic of these methods is that they regard similarity as an application-specific and highly subjective notion that cannot be effectively assessed

without taking in mind imprecision and the application's context. Therefore, for each fuzzy attribute type we first determine how imprecision affects similarity and what kind of contextual information (if any) is needed and then we proceed to suggest a proper similarity measure. On the other hand, the establishment of methods for the aggregation of different attribute-level similarities into a single similarity value is out of the scope of this work, as any such method from the already existing CBR approaches may be applied directly in our proposed framework without needing special adaptation.

5.2.1. Value Similarity for Fuzzy Property Attributes

For the purpose of our analysis, we consider in this paragraph a generic case retrieval scenario in which the stored cases are characterized by one fuzzy property attribute and have respective fuzzy sets of literals as values, while the requested case consists of a crisp set of literals regarding the same attribute. In this scenario, the similarity between a stored case and the requested one, based solely on the values of their fuzzy property attribute, is calculated by comparing their respective literal sets. In traditional CBR, where the two sets are crisp, this comparison is generally performed in two levels, namely the literal level and the set level. The first involves comparing each pair of the two sets' literals and finding their similarity, while the second involves aggregating these pair similarities into a single value denoting the similarity of the two sets. Literal-level similarity is usually assessed in a binary way that is it is 1, if the literals are the same, and 0, if not. Nevertheless, depending on the literal type, other similarity measures may be used as well. Set-level similarity in turn may be calculated in a number of ways depending on the application scenario. For example, in one scenario, it might be desirable that the requested case is fully similar to a stored one, when its value set is a subset of the latter's set, while in another scenario complete similarity might be desirable when the intersection of the two sets is a non-empty set. In any case the set-level similarity between the two value sets can be considered to be derived from a formula of the form

$$sim(RCVS, SCVS) = Agg(\{sim(rlj, slj)\}) \quad \textbf{(24)}$$

where *Agg* is a similarity aggregation function, *RCVS* and *SCVS* are the requested and stored cases' (crisp) value sets respectively and sim(rl_i, sl_j) is the similarity between the i*th* literal value of *RCVS* and the j*th* literal value of *SCVS* . The latter is usually 1 when the two literals are equal and 0 otherwise.

In our CBR framework, where imprecision is present, the fuzziness of the set *SCVS* is expected to play a role in the above computation. Our goal is to determine in what way this fuzziness should be incorporated in the above formula so that it covers all possible application scenarios. In doing that we consider the nature of the aggregation function *Agg* as well as the meaning of *SCVS's* fuzziness as defined in section 3.

More specifically, any manifestation of the function *Agg* regards the similarity sim(rl_i, sl_j) as a partial similarity between the sets *RCVS* and *SCVS* and consequently between the stored and the requested case.

When the set *SCVS* is fuzzy the strength of the association between any of its literals and the stored case is not fixed but it is determined by the corresponding fuzzy degrees. This means that in order for sim(rl_i, sl_j) to be considered as a partial similarity between the two cases it needs to be "adjusted" according to the fuzzy membership degree of the literal sl_j . Such an adjustment could take the following form:

$$sim\ (RCVS, SCVS)\ =\ Agg\ (\{sim\ (\ rli, slj\ )\ *\ SCVS\ (slj\ )\}) \qquad \textbf{(25)}$$

Where *SCVS* (sl_j) is the fuzzy membership degree of the j*th* literal value of *SCVS* . Nevertheless, the actual decision of whether this adjustment should be made depends on the given application scenario and particularly on whether the imprecision of the literal-case association is considered to play a role in similarity in that scenario. Therefore, in order to deal with the subjectiveness of the imprecision's role in similarity assessment we regard it as contextual information that needs to be captured and utilized within the similarity calculation formula. More specifically, given a set of fuzzy property attributes *FPA* we define Fuzzy Property Attribute Similarity Context as a function *fpasc* : *FPA* $\rightarrow\{0,1\}$. If *fpa* $\in$ *FPA* then *fpasc* (*fpa*) determines whether the imprecision of the attribute should be considered when comparing the value sets of a requested and a stored case for this attribute. A value of 0 denotes no consideration while a value of 1 the opposite. In the case of no consideration, the set

SCVS is considered crisp. Given that, the similarity assessment formula between a stored case *SC* and a requested one *RC* based solely on the values of the attribute *fpa* becomes:

$$\boldsymbol{sim}(\boldsymbol{RCVS}, \boldsymbol{SCVS}) = (\{\boldsymbol{sim}(\boldsymbol{rli}, \boldsymbol{slj}) * \boldsymbol{max}(\boldsymbol{SCVS}(\boldsymbol{slj}), \mathbf{1} - \boldsymbol{fpasc}(\boldsymbol{fpa}))\}) \quad \textbf{(26)}$$

5.2.2. Value Similarity for Fuzzy Valued Property Attributes

Similarly to above, we consider in this paragraph a generic case retrieval scenario in which the stored cases are characterized by one fuzzy valued property attribute and have respective single linguistic terms as values, while the requested case consists of a single linguistic term regarding the same attribute. In such a scenario, the similarity between a stored case and the requested one, based solely on the values of their fuzzy valued property attribute, is calculated by comparing their respective linguistic terms.

As suggested above, this type of comparison has been extensively examined in the literature; for example Chen and Hwang's *crisp score method* for defuzzifying fuzzy sets (S. J. Chen, 1992). This and similar approaches could be used within our framework as well.

5.2.3. Value Similarity for Fuzzy Relation Attributes

For the last type of case attributes, we consider again a retrieval scenario in which the stored cases are characterized by one fuzzy relation attribute and have respective ontology instances as values, while the requested case consists of a crisp set of ontology instances regarding the same attribute. The instances are considered to be derived from some fuzzy domain ontology. In such a scenario, the similarity between a stored case and the requested one, based solely on the values of their fuzzy relation attribute, is calculated by comparing their respective ontology instance sets. The only difference between this comparison and the one performed in the case of a fuzzy property attribute is that the compared set elements are ontology instances rather than literals. That is because for a single stored case the fuzzy related instances may be well regarded as a fuzzy set. This means that the above-derived similarity assessment formula. A may be applied here as well, reducing thus the problem into assessing the similarity between pairs of ontology instances. In traditional ontology-

based CBR, this is generally performed by utilizing the components of the domain ontology and particularly the relations that connect the instances. That is because most of these connections may, in certain contexts, indicate some kind of similarity between the instances. In our approach, where imprecision plays a central role, we claim that instance similarity is also influenced by the imprecision contained in the relations that connect them. Therefore, the fuzzy degrees of these relations need to somehow participate in the similarity assessment process.

Nevertheless, determining which fuzzy ontology relations, in what way and to what degree should participate in the assessment of instance similarity is a highly subjective and application-dependent task. That is because in different application scenarios and among different users, the contribution of the same fuzzy relation to the similarity between two instances might be totally different. Therefore, relevant contextual information needs to be modeled. With the above in mind, we define as parts of our semantic similarity framework the following components:

- The Fuzzy Ontology Relation Similarity Context (FORSC), a context model that enables the effective modeling of information regarding the expected role of the fuzzy ontology's relations in the instance similarity assessment process.
- An algorithm for the assessment of the similarity between any two fuzzy ontology instances based on the information contained in the ontology and the respective similarity context.

In particular, within an ontology an instance may be connected to another instance through a number of relations, or compositions of them. The aim of the Fuzzy Ontology Relation Similarity Context is to define whether and to what extent each of these connections should be interpreted as similarity ones.

As mentioned before, we can identify two broad kinds of ontology relations in which imprecision may be assigned a comprehensible meaning: hierarchical relations and associative ones. There are some models that use imprecise knowledge to compute similarity (Alexopoulos, Wallace, Kafentzis, & Askounis, 2010). Due to the fact that imprecise knowledge is acquired by experts, in such a situation, we will have to choose the best one. In previous works (Javadi-Moghaddam & Kollias, 2013), we have not only used imprecise knowledge to calculate similarity, but also given credit to the expert, as well as other factors. It is our belief that, if knowledge acquired by experts is going to be used to tackle such problems, one should consider validation of

the expert as a very important factor: considering such information as valid eventually means that all related beliefs are definitely true. These beliefs may deal with degrees of confidence related to the actual expert skills or to the actual confidence of the expert with respect to the specific topic under consideration. We have focused on the validity factor, taking into account fuzzy properties of the actual validation process. When an imprecise knowledge is present, we preferably compute similarity based on exploitation of fuzzy properties, due to the fact that the actual nature of imprecision is rather fuzzy. We have used fuzzy logic to compute the similarity of relevant cases and apply it for the retrieval case in ontology-based CBR (cased-based reasoning). In ontology-based CBR one typically utilizes two sets of case attributes: one set represents the database (DC), whereas another one forms the query (QC). To compute similarity, we should consider all members of these sets. We should also depict the effect of validation in all imprecision situations that we explained before. It is clear that the validation degree can be considered as a weight for all imprecise values, because it is a measure of our acceptability. At first, suppose d_i and q_j are literal of DC and QC respectively (such as value of property attributes that are expressed in an imprecise way). In order to compute the overall fuzzy semantic similarity by considering validation in the process, we propose the following equation:

$$Fsim(QC, DC) = Agg(sim(qi, dj) * max(max(Wvk * FDC(djk)), 1 - FIK) \quad \textbf{(27)}$$

where:

- Agg is a aggregation function,
- sim (q_i,d_j) is the similarity between the i_{th} literal value of QC and the j_{th} literal value of DC,
- FDC is the fuzzy membership degree of the j_{th} literal value of DC, and k_{th} expert.
- *FIK* value may be either 0 or 1, as it denotes whether imprecise knowledge is considered in similarity or not,
- W_{vk} is the degree of validation related to k_{th} expert.

In (21) we have also taken into account the *max* function according to Zadeh(Zadeh, 1965), because when we want to use a DC, two parameters are important for us: the fuzzy membership degree which shows the strength of the association between any DC and QC literals, as well as W_v, which expresses the validation of this degree.

In addition and as already mentioned, we should also take care of imprecise knowledge. Thus, when *FIK* value equals to 1, this means that we do take imprecision knowledge into account and therefore, its impact within the inner *max* function of (21) should be considered, in order to compute the overall similarity. Quite on the contrary, when *FIK* value equals to 0, this means that we shall not utilize imprecision knowledge in the process. Consequently, in this case 1-*FIK* value equals to 1 and the output of *max* function becomes 1, thus neutralizing its effect.

Then, to combine both situations, we utilize a meaningful fuzzy union operator, which in this case is depicted by the outer *max* function of (27).

Similarly to the above, we can use (27) with stored cases being expressed by one fuzzy valued property attribute and having respective single linguistic terms as values, while the requested case consists of a single linguistic term regarding the same attribute.

Finally, to compute similarity for Fuzzy Relation Attributes we should consider the validation degree for all relations as a weight (Hierarchical and associative relation). Therefore, by using an existing, well-known algorithm from the literature such as Fuzzy Ontology Relation Similarity Context (FORSC), we may calculate the final similarity. Alexopoulos (Alexopoulos et al., 2010) proposed the following operator for similarity context:

$$sco(R(a,b),f) = \begin{cases} R(a,b)^{1-f(R)} & ,0 \le f(R) \le 1 \\ R(a,b) \times (1+f(R), -1 \le f(R) < 0 \end{cases} \quad \textbf{(28)}$$

Where R belongs to fuzzy hierarchical or fuzzy associative relation, and *f* belongs to FORSC. Now to compute similarity by considering validation, we should insert the weight when we call the SCO function. Therefore, we'll have:

$$Rh = Sco(Wvh * HR, \text{ſ}), Ra = (Wva * AR, \text{ſ}) \quad \textbf{(29)}$$

- HR is hierarchical relation of concepts
- HA is associative relation of concepts
- W_{vh} is validation degree of hierarchical relation
- W_{va} is validation of associative relation
- R_h is hierarchical relation similarity context
- R_a is associative relation similarity context

In the end of the above process, we may compute the fuzzy relation $R_s = R_h \cup R_a$. According to R_s, we may obtain the overall semantic similarity by considering the actual validation.

References

Alexopoulos, P., Wallace, M., Kafentzis, K., & Askounis, D. (2010). Utilizing Imprecise Knowledge in Ontology-based CBR Systems by Means of Fuzzy Algebra. *International Journal of Fuzzy Systems*, *12*(1), 1–14.

Al-Mubaid, H., & Nguyen, H. a. (2009). Measuring Semantic Similarity Between Biomedical Concepts Within Multiple Ontologies. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *39*(4), 389–398. doi:10.1109/TSMCC.2009.2020689

Cross, V. (2004). Fuzzy Semantic Distance Measures Between Ontological Concepts. *IEEE Annual Meeting of the Fuzzy Information Processing (NAFIPS)*, *2*, 635–640.

Gomez-Perez, A., Fernández-López, M., & Corcho, O. (2004). Ontological engineering. *London et al*. Retrieved from http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Ontological+Engineering#0

Hirst, G., & St-Onge, D. (1998). Lexical chains as representations of context for the detection and correction of malapropisms. *Fellbaum*, 305–332. Retrieved from http://books.google.com/books?hl=en&lr=&id=Rehu8OOzMIMC&oi=fnd&pg=PA305&dq=Lexical+chains+as+representations+of+context+for+the+detection+and+correction+of+malapropisms&ots=IplgOiWXcb&sig=wPkuim3RHRIjgMLs3oJQEFMm37g

Javadi-Moghaddam, S.-M., & Kollias, S. (2013). The Important Role of Validation in Knowledge Intensive. *BCS SGAI journal Expert Update*.

Jiang, J. J. (1997). Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. *Computational Linguistics*.

Kahraman, C. (2007). Fuzzy decision-making applications (Special issue). *Approximate Reasoning*, *44*(2).

Lin, D. (1998). An information-theoretic definition of similarity. *International Conference on Machine Learning (ICML)*, 296–304. Retrieved from http://webdocs.cs.ualberta.ca/~lindek/papers/sim.pdf

Ou, M. H., West, G. A. W., Lazarescu, M., & Clay, C. (2005). Interactive knowledge validation and query refinement in CBR. In *PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE* (Vol. 20, p. 222). Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999. Retrieved from http://www.aaai.org/Papers/AAAI/2005/AAAI05-036.pdf

Petrakis, E. G. M., Varelas, G., Hliaoutakis, A., & Raftopoulou, P. (2006). X-Similarity : Computing Semantic Similarity between Concepts from Different

Ontologies object instrumentality. *Journal of Digital Information Management (JDIM)*, *4*, 233–237.

Pirró, G. (2009). A semantic similarity metric combining features and intrinsic information content. *Data & Knowledge Engineering*, (June 2009). Retrieved from http://www.sciencedirect.com/science/article/pii/S0169023X09000986

Resnik, P. (1995). Using information content to evaluate semantic similarity in a taxonomy. *the 14th International Joint Conference on Artificial Intelligence*, *1*. Retrieved from http://arxiv.org/abs/cmp-lg/9511007

Rodriguez, M., & Egenhofer, M. (2003). Determining semantic similarity among entity classes from different ontologies. *Knowledge and Data Engineering*, (1). Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1185844

S. J. Chen, C. L. H. (1992). Fuzzy Multiple Attribute Decision Making, Methods and Applications. *Springer Verlag*.

Sanchez, D., Batet, M., & Valls, A. (2010). Web-based semantic similarity: an evaluation in the biomedical domain. *International journal of software and Informatics*, *4*(1), 39–52. Retrieved from http://deim.urv.cat/~itaka/CMS/images/pdf/sanchezetalijsi2010.pdf

Sanroma, M. B. (2010). *ONTOLOGY-BASED ONTOLOGY SEMANTIC EMANTIC CLUSTERING*.

Song, L., Ma, J., Liu, H., Lian, L., & Zhang, D. (2007). Fuzzy Semantic Similarity Between Ontological Concepts. *Advances and Innovations in Systems, Computing Sciences and software Engineering*, 275–280. Retrieved from http://www.springerlink.com/index/LT72120883155821.pdf

Sussna, M. (1993). Word Sense Disambiguation Using a Massive of Computer for Free-text Semantic Indexing Network. *Proceedings of the second international conference on Information and knowledge management*, 67–74.

Tversky, A. (1977). Features of similarity. *Psychological Review*, *84*(4), 327–352. doi:10.1037/0033-295X.84.4.327

Tversky, A. (2004). *Preference , Belief , and Similarity*.

Wu, Z., & Palmer, M. (1994). VERB SEMANTICS AND LEXICAL SELECTION. *32nd annual Meeting of the Association for Computational Linguistics*, 133–138.

Zadeh, L. (1965). fuzzy set. *Information and control*.

Zhai, J., Liang, Y., Jiang, J., & Yu, Y. (2008). Fuzzy Ontology Models Based on Fuzzy Linguistic Variable for Knowledge Management and Information Retrieval. *Intelligent Information Processing IV*, *288*, 58–67. Retrieved from http://www.springerlink.com/index/j71124717520hj43.pdf

Chapter 5

A Fuzzy Similarity Measure for XML Documents

1. Introduction

With the continuous growth of the Internet, Extensible Markup Language (XML) has become more popular as a dominant standard for the representation and exchange of data over the Web. This has led to massive collections of XML data and has imposed an enormous opportunity and challenge for grouping XML documents based on their context and structure. Clearly, the computation of similarity plays an important role for grouping and clustering XML documents.

XML documents are comprised of nested elements. XML Documents can be labeled as a tag tree of element nodes and each XML element is represented by a node in the tree; the node is labeled by the name or value in the XML element. Element-sub element relationships are represented by edges in the tree. This structural information is an important criterion to compute similarity. On the other hand, the value and meaning in the nodes data are significant for measuring similarity. Therefore, to compare two XML documents, it is necessary to consider both the structure and the contents of them.

Over the past years, there have been proposed a large amount of XML document similarity estimation methods. Some approaches have focused on similarity of documents' structure (Aïtelhadj, Boughanem, Mezghiche, & Souam, 2012). Other researchers (Kurgan, Swiercz, & Cios, 2002) have computed similarity, based solely on the content of documents. Since structure and content of documents play an important role in assessing the similarity, the approaches based on only one of them do not provide very good results. Consequently, several authors have provided similarity algorithms based on both structure and content of XML documents (Kim, 2008). In most of structure-similarity based approaches, structure could be either a labeled tree corresponding to the original structure of the XML document (the whole structure of document), or a rooted ordered labeled tree summary. For example, the XML tree can be decomposed into path information called node paths, i.e., ordered sets of nodes from the root node to a leaf node. The most usual distance measure for tree-structured data is the tree edit distance (Bille, 2005). In this method, the edit distance is computed by considering alternative sequences of edit operations that can transform one object into the other. The cost of the operations in each sequence is considered, and the lowest cost sequence among these defines the edit distance

between the two objects. However, computing the tree edit distance can be very expensive both in terms of CPU cost and disc I/Os.

To compute similarity based on content, we should take into account the semantics of the elements. In most cases the mapping is generated by only stand-alone XML documents, without DTD or XML schema (Kurgan et al., 2002).

Similarity based on both structure and content can be computed by considering the semantics of elements and nested structures of XML documents (Park & Seo, 2005). Such a method measures the similarity between XML documents by considering their structures and content, using a three-layer matching: element matching, path matching and document matching. This model is based on the bag of tree paths; it can, however, lead to a high time complexity.

As mentioned before, edit distance is time-consuming and the similarity result may not be accurate in terms of semantics. For this reason, it has been proposed to measure similarity according to string matching. Li (G. Li, Liu, Feng, & Zhou, 2008) has proposed to transform tree structured data into strings with a one-to-one mapping. He has proven that the edit distance of the corresponding strings forms a bound for the similarity measures between trees, including tree edit distance, largest common sub trees and smallest common super-trees. Navarro (Navarro, 2001) gives a good overview of the edit distance for strings and its variants. Ukkonen (Ukkonen, 1992) introduces the q-gram distance as a lower bound for the string edit distance. The q-gram distance between two strings is based on the number of common sub strings of length q. Gravano et al. (Gravano, Ipeirotis, & Jagadish, 2001) present algorithms for approximate string joining based on edit distance and using q-grams for filtering.

In this work, we propose a new method for measuring the similarity between two XML documents in terms of their structure and content. The Sorensen–Dice's coefficient is used for computing the similarity of documents' structure. In addition, similarity of documents' contents is estimated by fuzzy analysis taking into account the names and positions of elements. To achieve a high method performance, we transform the trees into strings with an one-to-one mapping. Then, the similarity of documents' structure is found by simple string matching and that of documents' content is found by simply taking into account the names and positions of elements.

The overall algorithm runs in linear time with respect to the combined size of the two documents involved in the evaluation.

2. MAPING XML TO TREE

2.1. Ordered Labeled Tree for XML Documents

An XML document can be mapped to an ordered labeled tree where each node in the tree describes either an element or an attribute in the XML document (Behrens, 2000). The edges represent a hierarchical relationship between nodes that can relate two elements or an element to an attribute. They are also labeled by the tag name of the element or by the name of the attribute (Fig. 1).

```
<paper>
<author>...</author>
<journal>
<title>...</title>
<year>...</year>
</journal>
<conference>
<title>...</title>
<year>...</year>
<city>...</city>
</conference>
</paper>
```

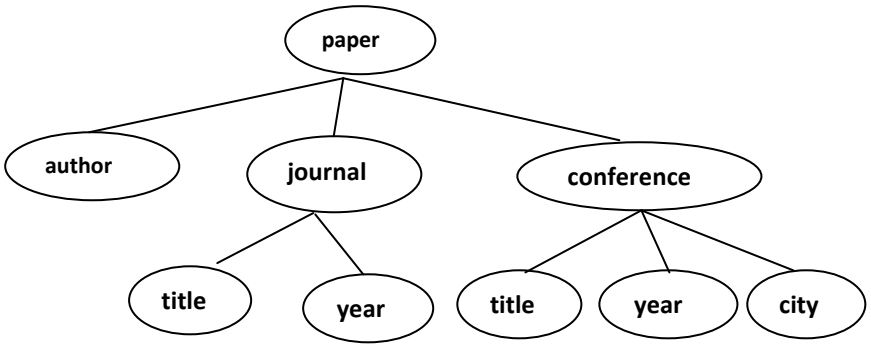


Figure 1. An XML document for publication.

2.2. Level Labeled Tree for XML Documents

Level labeled tree is an ordered labeled tree with the difference that the label of a node is the number of its level. Each edge of the tree represents a hierarchical inclusion relationship between either two elements or an element and an attribute. The root's level is 0 (Fig. 2).

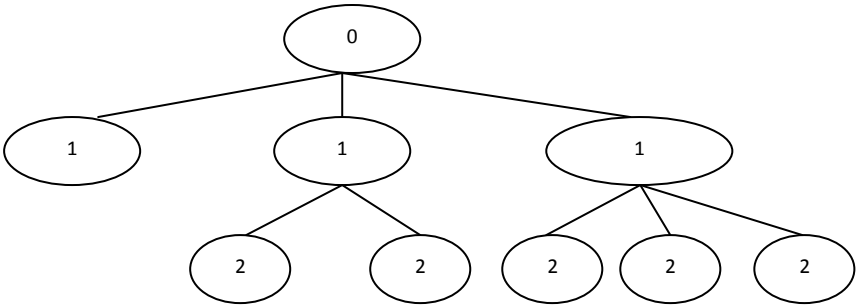


Figure 2. Level labeled tree for the XML document shown in Figure. 1

2.3. Weighted Tree

In order to compare two XML documents, not only should we consider the nodes' names similarity, but also notice the position of the nodes in the mapped XML tree (Kim, 2008). To do it, we construct a tree, each node of which corresponds to a node in the ordered labeled tree and is labeled with proper weight. Each edge in this tree indicates inclusion of the node corresponding to the child one in the node corresponding to the parent one, within the ordered labeled tree. In this paper, we assign weight 1 for the root node. In order to reflect a path to the weight criterion, a parent node gains more weight than a child node. A child node's weight is considered equal to the parent node's weight divided by m, if the parent has m ($m \geq 2$) child nodes. But if a parent has only one child, then the child node's weight becomes a half of parent node's weight in order to consider a path.

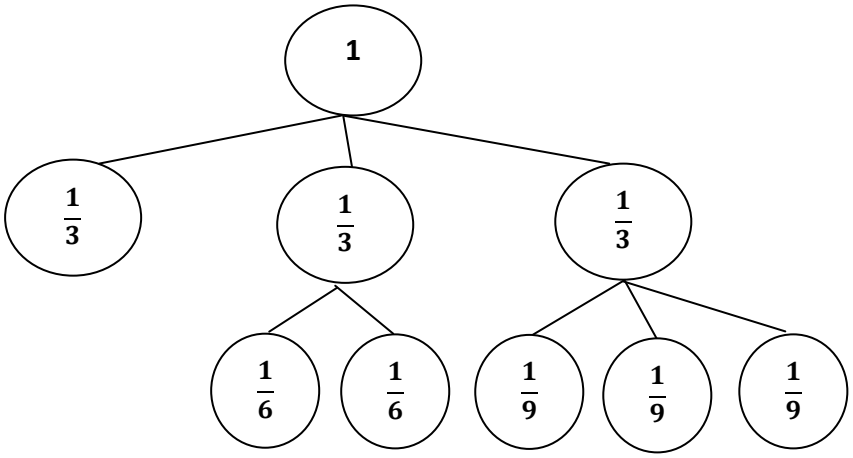


Figure 3. Weighted tree for XML document in Fig. 1

3. TREE 'S STRINGS

This section introduces a one-to-one sequencing method to transform trees to strings. Prufer proposed a method that computes an one-to-one correspondence by removing nodes from the tree, one at a time (H. Prüfer, 1918). Another way to

transform the tree to string is built by traversing on the tree. In this paper, we build strings for the tree by using depth first algorithm. To save the relationship between the nodes, we make three strings according to the meaning and position of the nodes.

3.1. Depth First Search (DFS)

One of the systematic methods of visiting the vertices of a graph is DFS. Given that we are currently visiting vertex u, in the next step, we visit a vertex adjacent to u which has not yet been visited yet. If no such vertex exists, then we return to the vertex visited just before u and the search is repeated until every vertex in that component of the graph has been visited. Time complexity of the initialization part of DFS is $\Theta(n)$, as every vertex must be visited once. Moreover, $\Theta(m)$ is time complexity of the main(recursive) part of the algorithm. In total, the algorithm's time complexity is $\Theta(m + n)$ where n shows the number of nodes that are discovered for the first time and m is the number of nodes that are visited for the second time.

3.2. Element String (ES)

The ES string is built by applying the DFS algorithm on the ordered labeled tree. This string compromises the objects that express semantics of the nodes. Below is an example according to Fig. 2:

ES: "paper,author,journal,conference,title,year,tile,year,city"

3.3. Numeral String (NS)

To build the NS, we should execute the DFS algorithm on a Level labeled tree, using a specific number for every level. By going down the tree, the number increases. According to NS, the position of each node can be defined. The NS for Fig. 2 can be computed as follows:

NS: "011122222"

3.4. Weight String (WS)

By executing the DFS algorithm on the weight tree, we can build the WS. Indeed, this string shows the amount by which the nodes can be influenced by root. We have used this amount as a coefficient for similarity. In the following, we show the WS for Figure 3.

WS: "1, 1/3, 1/3, 1/3, 1/6, 1/6, 1/9, 1/9, 1/9"

4. SIMILARITY BETWEEN STRINGS

4.1. String Edit Distance

Similarity between strings is usually evaluated by string edit distance. String edit distance operations include substitution, insertion, and deletion of a character with the cost of each operation being always assigned to one. Many researchers have proposed approaches to compute similarity search and similarity join based on edit distance over textual data (Gravano et al., 2001). Gravano et al. (Gravano et al., 2001) proposed q-grams to measure similarity match on textual data. A q-gram is a contiguous substring that has length q. In this approach, similarity is computed by using common q-grams. To improve the performance Li et al. (C. Li, Wang, & Yang, 2007) proposed a new technique called VGRAM that selects the high-quality grams from a collection of strings. A method to facilitate similarity searches, is given in (C. Li, Lu, & Lu, 2008). Indexing structures and specific merging algorithms are proposed.

4.2. String Matching

String matching can be done by two methods: accurate matching and approximate matching (fuzzy string searching). Accurate string matching is used when a query string is similar identically to another string. Approximate String Matching Algorithm (fuzzy string matching) is the technique of finding strings that match approximately (rather than exactly). In this paper, we compute string similarity by using approximate string matching. To estimate string matching, we apply the Sorensen–Dice's coefficient.

4.2.1. Sorensen- Dice's Coefficient

The Sorensen–Dice's coefficient is a statistical entity used for comparing the similarity of two samples. It is computed according to the number of common species between the two samples and to the number of species in the two samples. Given that A and B are the numbers of species, the similarity of two samples be calculated as follows:

$$Sim = \frac{2|A \cap B|}{|A|+|B|} \quad (1)$$

4.3. Similarity Measure

Dice's coefficient can be used to measure two strings' similarity (Kondrak et al., 2003). To do this, we need to calculate the bigrams of the two strings. A bigram is every sequence of two adjacent elements in a string of tokens, which are typically letters, syllables, or words. Taking the string "football", the set of bigrams would be {"fo", "oo", "ot", "tb","ba", "al","ll"}. The similarity of two strings is given by the formula:

$$\text{Sim}(x,y)=\frac{2n_t}{n_x+n_y} \quad (2)$$

Where n_t is the number of character bigrams found in both strings, n_x is the number of bigrams in string x and n_y is the number of bigrams in string y. For example, to calculate the similarity between," table" and "taken" , we would find the set of bigrams in each word:

{ta,ab,bl,le},{ta,ak,ke,en}

Each set has four elements, and the intersection of these two sets has only one element: ta. Inserting these numbers into the formula, we calculate, Sim = (2 • 1) / (4 + 4) = 0.25.

5. A FUZZY SIMILARITY ALGORITHM FOR XML

In this algorithm, we obtain the similarity of two XML documents according to structure and content of them. The similarity of documents' structure is found by simple string matching and that of documents' content is found by computing weights that take into account the names and positions of elements. First, two corresponding ordered labeled trees are built by the two XML documents. The nodes of these trees are the included weights and the level numbers beside the names of elements; they are computed at the same time. Next, three strings of the tree's nodes are obtained as mentioned in the previous chapters. The similarity is computed as follows:

$$\text{FSim}(dx,dy)= \text{CSim}(dx,dy) * Wc + \text{SSim}(dx,dy) * Ws \quad (3)$$

Wc+Ws=1,

where CSim is a content similarity, SSim is structure similarity, Wc and Ws are coefficients that define the importance of content similarity and structure similarity respectively. These coefficients are adjusted by the user.

5.1. Structure Similarity (SSim)

The structure similarity between two XML documents d1 and d2 can be calculated by the corresponding trees of d1 and d2. In this approach, the structure of a tree reflects the nested structure of an XML document. To compute the SSim, we use the numerical string that is composed at nodes' level. Kim (Kim, 2008) assumed that two documents are structurally similar, when they are structurally identical, or structurally contain each other. Therefore, after transferring tree to string, a matching algorithm is used to measure the similarity. In this paper, we calculate similarity by using approximate matching. Accordingly, we assess the structure similarity with respect to common edges of the two trees. First, NS is built for d1 and d2. Then, the bigrams of the two strings are calculated. The reason that bigrams are used is to preserve the nested structure of the XML document. Finally, we compute the SSim through (2).

5.2. Content Similarity (CSim)

Content similarity is obtained by using ES and equation (2). When we want to specify content similarity (CSim) for an XML document, we should consider the semantics of nodes besides an equality between them. As mentioned before, the weight string can represent the position of nodes according to the root; this can be considered as a weight for each node. Therefore, we can make a fuzzy set the members of which belong to name string, while the fuzzy memberships are obtained by the weight string. To compute the CSim we make a fuzzy set for each one of the two XML documents. Then we specify CSim by using equation (1) and fuzzy intersection. In this paper, we use Zadeh `s intersection (Zadeh, 1965). In equality of nodes' name, we use WordNet (Miller, 1995) to increase accuracy. Equality can be based on name or value of element, or on both of them. Introducing a coefficient λ, the equality is computed as follows:

$$Eql(S1,S2) = \lambda * Ename(S1,S2) + (1-\lambda) * Evalue(S1,S2), \quad (4)$$

where Ename and Evalue are two functions that compute the equality according to the name and value respectively. Coefficient λ is adjusted by the user, according to the importance assumed for each function.

The time complexity of DFS is O(n), where the number of nodes of a tree is n. The time complexity of pattern matching is O(n+m), when the length of the text is n, and the length of the pattern is m (Knuth, Morris, Jr, & Pratt, 1977). When the number of nodes of a tree is n, the time complexity to obtain fuzzy similarity between two XML documents is O (2n). Therefore, the time complexity of the algorithm is linear in the combined size of the two XML documents involved in the evaluation.

6. EVALUATION

In our experiments, we used the dataset of Lotus Hill Research Institute (LHI) (Yao, Yang, & Zhu, 2007). This dataset includes 1767 images that have classified into eight categories and 150 subcategories. In addition, each image has an annotation in XML format.

To evaluate our algorithm, we computed the similarity of XML files in identical and different categories.

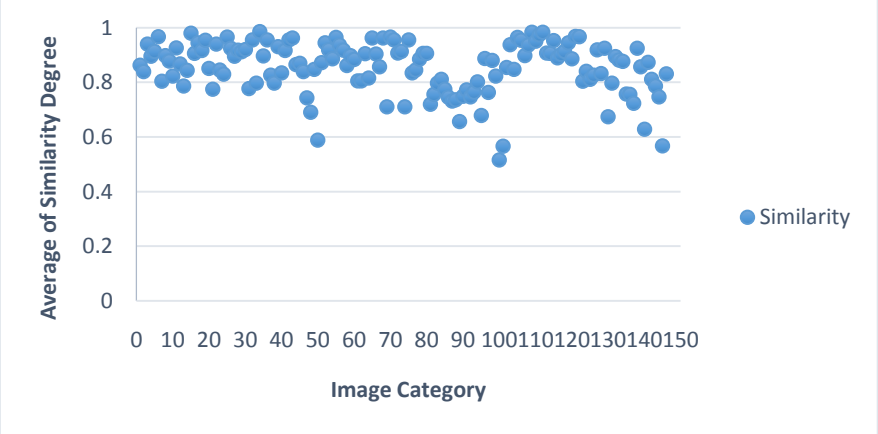


Figure 4. Similarity of images annotated in the same category

Fig. 4 shows the average similarity between the images annotated in the same categories. Clearly, the images placed within a category should have high similarity, which is the result properly obtained in Fig. 4.

We also experimented to determine the similarity between two XML documents in different categories. As is shown by the results of this comparison (Fig. 5), the similarity degree is mostly less than 0.5, as expected. Some cases that have a degree higher than 0.5 belong to situations where two categories are quite similar. For example, subcategory Street is in both Object and Segmentation categories.

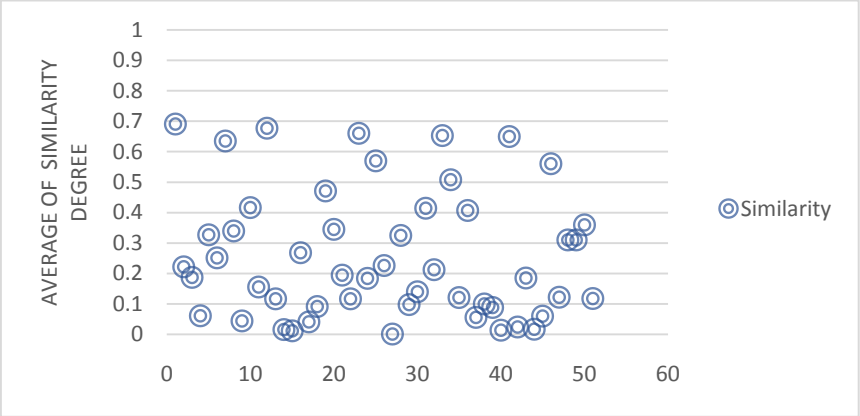


Figure 5. Computing the similarity of images annotated in different categories

In summary, we have formalized the problem of measuring similarity between two XML documents. The presented algorithm computes similarity by considering content and structure of XML documents. To compare two XML documents, we have used the strings made according to the ordered labeled trees of them. To increase the accuracy, we have used fuzzy metrics. The time complexity of the algorithm is linear w.r.t. the combined size of the two XML documents. Our future research includes the development of a clustering model based on the proposed approach and the experimental verification of the model.

References

Aïtelhadj, A., Boughanem, M., Mezghiche, M., & Souam, F. (2012). Using structural similarity for clustering XML documents. In *Knowledge and Information Systems* (Vol. 32, pp. 109–139). doi:10.1007/s10115-011-0421-5

Behrens, R. (2000). A grammar based model for XML schema integration. *British National Conferenceon Databases (BNCOD)*, 172–190. Retrieved from http://link.springer.com/chapter/10.1007/3-540-45033-5_13

Bille, P. (2005). A survey on tree edit distance and related problems. *Theoretical Computer Science*, *337*(1-3), 217–239. doi:10.1016/j.tcs.2004.12.030

Gravano, L., Ipeirotis, P., & Jagadish, H. (2001). Approximate string joins in a database (almost) for free. *VLDB*.

H. Prüfer. (1918). Neuer beweis eines satzes uber permutationen. *Archiv fur Mathematik und Physik*, *27*, 142–144.

Kim, W. (2008). XML document similarity measure in terms of the structure and contents. *COMPUTER ENGINEERING and APPLICATIONS (CEA)*, 205–212. Retrieved from http://www.wseas.us/e-library/conferences/2008/mexico/cea/36-CEA.pdf

Knuth, D., Morris, J., Jr, & Pratt, V. (1977). FAST PATTERN MATCHING IN STRINGS. *SIAM journal on computing*, *6*(2), 323–350. Retrieved from http://epubs.siam.org/doi/abs/10.1137/0206024

Kondrak, G., Hall, A., Tg, C., Marcu, D., Knight, K., & Rey, M. (2003). Cognates Can Improve Statistical Translation Models. *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics(HLT-NAACL)*, 46–48.

Kurgan, L., Swiercz, W., & Cios, K. (2002). Semantic Mapping of XML Tags Using Inductive Machine Learning. *ICMLA*. Retrieved from http://biomine.ece.ualberta.ca/papers/ICMLA2002.pdf

Li, C., Lu, J., & Lu, Y. (2008). Efficient merging and filtering algorithms for approximate string searches. *ICDE*. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4497434

Li, C., Wang, B., & Yang, X. (2007). VGRAM: Improving performance of approximate queries on string collections using variable-length grams. *VLDB*. Retrieved from http://dl.acm.org/citation.cfm?id=1325889

Li, G., Liu, X., Feng, J., & Zhou, L. (2008). Efficient Similarity Search for Tree-Structured. *SSDBM*, 131–149.

Miller, G. a. (1995). WordNet: a lexical database for English. *Communications of the ACM*, *38*(11), 39–41. doi:10.1145/219717.219748

Navarro, G. (2001). A guided tour to approximate string matching. *ACM computing surveys (CSUR)*. Retrieved from http://dl.acm.org/citation.cfm?id=375365

Park, U., & Seo, Y. (2005). An Implementation of XML Documents Search System based on Similarity in Structure and Semantics. *International Workshop on Challenges in Web Information Retrieval and Integration(WIRI)*, 97–103. doi:10.1109/WIRI.2005.8

Ukkonen, E. (1992). Approximate string-matching with q-grams and maximal matches. *Theoretical Computer Science*, *92*(1), 191–211. doi:10.1016/0304-3975(92)90143-4

Yao, B., Yang, X., & Zhu, S. (2007). Introduction to a large-scale general purpose ground truth database: methodology, annotation tool and benchmarks. *Energy Minimization Methods in Computer Vision (EMMCVPR)*, 169–183. Retrieved from http://link.springer.com/chapter/10.1007/978-3-540-74198-5_14

Zadeh, L. (1965). fuzzy set. *Information and control*.

Chapter 6

Small-World Networks

1. Introduction

Small-world networks attempt to mimic the characteristics of social acquaintance networks. In a sociological context, people connect to each other intelligently and not randomly. This means, when people search for new contacts, do not rely on seeking out random people. Instead, their existing contacts help them in finding other individuals who share similar interests and characteristics (e.g., friends, hobbyists, etc.), or who satisfy specific criteria (e.g., business needs, looks, etc.). Yet, in order for people to be able to perform intelligent referrals, they must keep some knowledge about their own set of contacts. Thus, each person retains stored in their brains some type of a description of each person they know. This sociological ideology is employed in the generation of small worlds. In a similar way that people use referrals from their current friends and contacts to continuously find persons who more closely share the same interests and/or characteristics, the small world nodes discussed here interact within the small world network to find their most similar small world peers.

In late 1960, Stanley Milgram performed quantitative studies on the structure of social networks (Milgram 1967). He did a simple experiment, as follows. He wrote a number of letters addressed to his stockbroker acquaintance in Boston, Massachusetts, and sent them to a random selection of people in Nebraska. He had some instructions so that the letters should be sent to their addressee (the stockbroker) by passing them from person to person. Moreover, they could be sent only to someone whom the passer new on a first-name basis. According to improbability of receiving the letters by recipients in Boston's stock, the letters were highly probably sent by recipients to whom they felt that nearer to the stockbroker in some social sense: perhaps someone they knew in the financial industry, or a friend in Massachusetts. The number of Milgram's letters that did eventually reach their destination was reasonable. Therefore, Milgram found that an average of six steps for a letter to get from Nebraska to Boston has been taken. Regarding these steps, he concluded that the average number of acquaintances separating the pairs of people was six. Furthermore, he surmised that a similar separation may characterize the relationship of any two people in the entire world. This position has been called "six degrees of separation" (Guare 1992).Clearly, the figure six in Milgram's experiment is probably not a very accurate one. There are certainly many possible sources of error in the experiment.

However, it can conclude as a general result that two randomly chosen human beings can be connected by only a short chain of intermediate acquaintances, this has been subsequently verified, and is now widely accepted. In terminology, this result is expressed as the **small-world effect**. Small-world characteristics have been used by various social networks ((M. E. Newman, Moore, and Watts 2000)(Watts and Strogatz 1998)).

Two characteristics which one might imagine on being contradictory are met in Social networks within networks of friends. The first is clustering, stating that the probability of two of your friends being friends, is higher than that of two people chosen from the population at random. Second, any two people can relate each other by going through only a short chain of intermediate acquaintances, as usual in small-worlds. Indeed, the small-world network model is a simple model of the structure of social networks, which includes characteristics of both regular lattices and random graphs. The model can be generated by a one-dimensional lattice with a low density of shortcuts between randomly selected pairs of points. By using these shortcuts the typical path length between any two points on the lattice is greatly reduced.

2. Regular Networks

These networks are termed regular, because there is the same number of links in each node. Indeed, regular networks are highly ordered. They can be represented as square, rings, trees, and stars. The network flocks where the behavior of each node depends upon the behavior of its nearest neighbors. The topological rule is that each node is linked to all of its nearest neighbors. A degree is considered for each network that determines the number of neighbors. There is no need to define a statistical rule for the network's degree distribution, because the number of degrees is the same for each node (as shown in Figure 1).

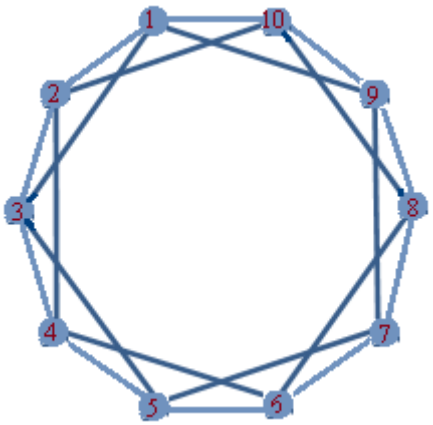


Figure 1: A regular ring network with four neighbors.

Notice that, each node is clustered to four near neighbor nodes. For example, nodes 1,2,3,10, and 9 form a cluster because they all connect to node #1. The circular form makes it easy to demonstrate the effects of short path lengths and clustering on connectivity. The nodes that are placed on opposite sides of the network are members of separate clusters that do not intersect. Clearly, there are relatively long path lengths between these remote nodes and are considered as low connectivity. On the other hand, two nodes that are within three nodes from each other on the circle (nodes #1 and #4) belong to separate, but intersecting clusters. This is equivalent to two groups of friends where some people belong to both groups. This kind of intersecting clustering enhances connectivity within the network. A regular network does not suggest high connectivity in many cases, because a long and circuitous route is required to reach many nodes.

3. Random networks

A very simple model of a random network can be made by taking N dots as nodes or vertices and drawing $\frac{1}{2}Nz$ lines ("edges") between randomly chosen pairs to represent these connections (Bollobás 1998). Clearly, a random graph shows the small-world effect. Given that a person A on such a graph has z neighbors, and each of A's neighbors also has z neighbors, then A has about z^2 second neighbors. By extending this parameter, person A also has z^3 third neighbors, z^4 fourth neighbors and so on.

Another important parameter in network is the diameter D which is the number of degrees of separation, which is needed to consider, in order to reach all N nodes in

the network. This is given by setting $z^D = N$, which implies that $D = \log N / \log z$. Clearly, logN increases only slowly with N, allowing the number of degrees to be quite small, even in very large systems. The problem is that people's circles of acquaintance tend to overlap to a great extent. Your friend's friends are likely also to be your friends, or to put it another way, two of your friends are likely also to be friends with one another. Therefore, in a real social network it is not true to say that person A has z^2 second neighbors, because many of those friends of friends are also themselves friends of person A. This property is termed clustering of networks. A clustering coefficient C is defined as the average fraction of pairs of neighbors of a node which are also neighbors of each other. In a fully connected network, in which everyone knows everyone else, C = 1; in a random graph $C = \frac{z}{N}$, which is very small for a large network.

The average distance ℓ between pairs of nodes is another parameter that is important in the networks. It is not the diameter D of the network, but it also scales at most logarithmically with number of nodes. In addition, the average distance is strictly less than, or equal to, the maximum distance, and so ℓ cannot increase any faster than D. So, the random network is poorly clustered, but the average distance between pairs of nodes is small.

4. Small-world models

4.1. The small-world model of Watts and Strogatz

As we have argued, random networks show the small-world effect, possessing average vertex-to-vertex distances which increase only logarithmically with the total number N of vertices, but they do not show clustering—the property that two neighbors of a vertex will often also be neighbors of one another.

The opposite of a random network, in some sense, is a regular network. The simplest instance of this network is a one-dimensional lattice—a set of vertices arranged in a straight line. If each node is connected to the k vertices closest to it, i.e., it shows the clustering property, the regular network is built (Figure 1).

For such a network we can calculate the clustering coefficient C exactly (M. E. J. Newman 2000). As long as $k < \frac{2}{3}N$, which it will be for almost all graphs, C can be computed by this equation:

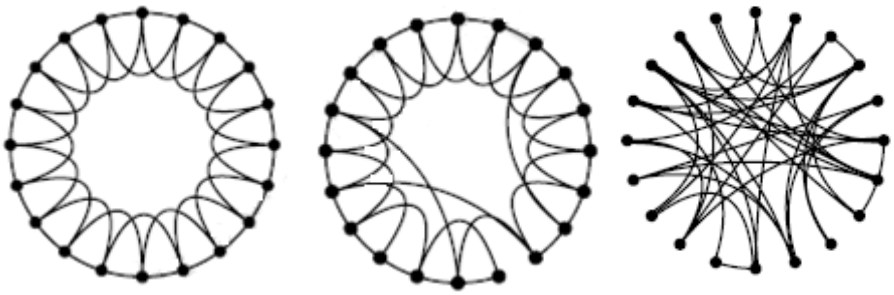
$$C = \frac{3(k-2)}{4(k-1)} \quad (1)$$

which tends to $\frac{3}{4}$ in the limit of large z. We can also build networks out of higher-dimensional topologies, such as square or cubic, and these also show the clustering property. The value of the clustering coefficient in general dimension d is

$$C = \frac{3(k-2d)}{4(k-d)} \quad (2)$$

which also tends to $\frac{3}{4}$ for k>>2d.

Therefore, for small values of d, regular networks do not possess small-world behavior, which increases only slowly with system size. For a regular network which has the shape of a square or cube of side L, it can be shown that $N=L^d$. That means that the average vertex–vertex distance increases as L; this does not show the small-world effect. But for large value of d, it will have the small-world effect, because $N^{1/d}$ becomes a slowly increasing function of N.



Regular Network *Small-world Network* *Random Network*

Figure 2: Small-world network compared to Regular and Random networks

In 1998, Watts (Watts and Strogatz 1998) proposed a model for small-worlds that can be built between regular network and random distributed networks. The researchers have proved that some parameters, such as size and average distance in the regular network, are bigger than in the random network. Their suggestion was to build a model which is, in essence, a low-dimensional regular lattice—say a one-dimensional lattice—but which has some degree of randomness in it, like a random graph, to produce the small-world effect. Watts assumes a small possibility p for regular network's edges. Moreover, selection of a new node for the connection is made absolutely at random. Therefore, we can disconnect each edge in the graph from

its end point and connect it to another random node. By this action, the graph diameter is reduced, because the farther node is being reconnected too. In only a few steps, the network can have a shorter diameter without reducing the clustering coefficient. In this model, each node in the graph is considered equally likely to be rewired. Both regular and random networks could be achieved by changing the value of p. If we set p=0, then we will produce a regular network. On the other side, when we select p=1, random network is built. Small world networks have been constructed by using $0<p<1$, having both high cluster parameters and small average distances (Figure 2).

This model can be justified in social terms in that most people are friends with their immediate neighbors—neighbors on the same street, people that they work with, people that their friends introduce to some people and are also friends with one or two people who are a long way away, in some social sense—people in other countries, people from other walks of life, acquaintances from previous eras of their lives, and so forth. The model of Watts and Strogatz represents these long-distance acquaintances by the long-range links. It is straightforward to show that the values of the clustering coefficient C for the Watts–Strogatz model, with small values of p, will be close to those for the perfectly regular network, which tend to ¾, for fixed small d and large k. Moreover, they showed that the average vertex–vertex distance ℓ is comparable with that of a true random graph, even for quite small values of p. By numerical simulation, they showed for a random graph with N = 1000 and k = 10, that the average distance was about $\ell = 3.2$ between two vertices chosen at random. In addition, the average distance was only slightly greater, at $\ell = 3.6$, when the rewiring probability $p = \frac{1}{4}$, compared with $\ell = 50$ for the graph with no rewired links at all. And even for $p = \frac{1}{64} = 0.0156$, they found $\ell = 7.4$, a little over twice the value for the random graph. Therefore, the model is able to have both the clustering and small-world properties simultaneously.

4.2. Newman-Watts Model

Newman and Watts (M. Newman and Watts 1999) studied the model of Watts and Strogatz using the techniques of statistical physics, and showed that it possesses a

continuous phase transition in the limit where the density of shortcuts tends to zero. They investigated this transition using a renormalization group (RG) method and calculated the scaling forms and the single critical exponent describing the behavior of the model in the critical region. They clarified that considering equal probability for each node in the Watts-Strogatz model has two problems. First, shortcuts do not distribute totally uniform and all choices of positions are not equally appropriate for rewiring. Secondly, this model poorly defines the average distance between pairs of vertices on the graph because there is a probability of detaching some parts of the graph through the process of rewiring. They kept the original links without variation, while adding shortcuts between pairs of vertices that were chosen uniformly at random. For sufficiently small p and large L, this makes no difference to the mean separation between vertices of the network for $k \geq 2$. For $k = 1$ it does make a difference, since the original small-world model is poorly defined in this case—there is a finite probability of a part of the lattice becoming disconnected from the rest and therefore making an infinite contribution to the average distance between vertices, and this makes the distance averaged over all networks for a given value of p also infinite. Therefore, they considered more than one link between any vertice connected to it.

4.3. Small World Index Model (SWIM)

Two special issues are integrated in SWIM (Androutsos, Androutsos, and Venetsanopoulos 2006): relatively unrelated and highly "connected" concepts. The first expresses a comprehension upon which the world's population is placing increasing demands on a daily basis. For most of people, Internet is considered as the main source of communication with friends, clients, and peers, as well as their first choice for up-to-the-minute, customized, and interconnected news, information, and knowledge. On the other hand, the highly connected nature of the Internet's World Wide Web (WWW) is truly astronomical in size, but it all manages to work together, grow, and evolve in a very natural way.

The model's distributed nature comes from the fact that it tries to exploit the small world phenomenon, where all members of a social network retain information about their peers and apply it to the region of media indexing. Three important actions are done by each SWIM node:

■ store information about other nodes

■ actively search for similar peers

■ interact with other nodes in the SWIM network to assist them in performing the same tasks.

The SWIM is a model that is very close to Newman and Watts's model, but has some different fundamental assumptions and modifications. At first, in contrast to the models in (Watts and Strogatz 1998) and (M. Newman and Watts 1999), which exclusively use undirected edges, all connections within the SWIM network are directed, similarly to the structure of the WWW. Specifically, in the SWIM system, an underlying directed *k*-lattice is embedded together with a directed pseudorandom graph in an *N*-node network (see Figure 3). It should be noted that the SWIM network does not establish random connections between nodes according to a set probability value (unlike the Newman-Watts model). Instead, connections are built in accordance to a particular external distance that computes by data stored locally of each node. There is a one descriptor in all nodes that is used to calculate the distance. As mentioned before, the similarity graph is related to a specific descriptor and distance measure. Therefore, by using a different descriptor for describing the produces data we will have a different similarity graph. Furthermore, the use of a different distance measure also results in a different graph. Given that two vertices N_1 and N_2 which have the vectors $\overrightarrow{d_1}$ and $\overrightarrow{d_2}$ to describe them, the distance between them is denoted as $D_L(\overrightarrow{d_1}, \overrightarrow{d_2})$. There are many different descriptions can be used for ring topology but the most straightforward is a descriptor that generates the x− y coordinates of each vertex, a two-dimensional (2-D) description vector. To calculate the smallest distances between them, all the Euclidean distances between all descriptions are computed. Then the smallest one is selected.

It is important to note that the k-lattice (called weak network) in SWIM is not dependent on the distance measure. Instead, it is only dependent on the order in which nodes are introduced into the network.

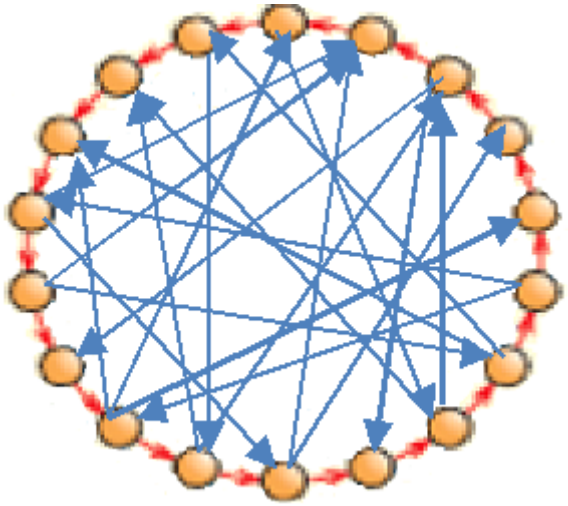


Figure 3: The SWIM graph made by combining similar graph and K-lattice

Using conventional graph notation, a similarity graph G_p can be expressed with a vertex and edge set pair as

$$G_p = \left(V_N, E_p(P, d, D_L)\right) \quad \textbf{(3)}$$

where V_N denotes the set of vertices $\{v_1, v_2, ..., v_N\}$ and the edge set Ep is dependent on the number of small world peers P, the particular descriptor d, and the distance measure D_L employed. In contrast to Gp, the directed k-lattice ring component of the SWIM model creates connections by joining nodes to their k successors with a periodic boundary condition. Since these directed connections are established independently of distance measure and descriptor, they create a weak graph (i.e., not based on similarity), which is denoted by Ge that is build according to entrance order and is defined using (4) and (5).

$$G_e = (V_N, E_e(K)) \quad \textbf{(4)}$$

$$G_e(K) = \left\{ \forall(v_i, v_j) \middle| \begin{matrix} if\ i < N - l\text{:}\ \ j = i + l \\ else\text{:}\quad\ \ j = i + l - N \end{matrix} \right. \quad \textbf{(5)}$$

Where i, j= 0, 1,..., N and ℓ = 1, 2,..., k. Figure 3 shows a typical SWIM network's similarity and weak graphs, for a given distance measure and descriptor, where N = 16, k = 1, and P = 2. When viewing these graphs, the highly regular structure of G_e is immediately apparent, while the connections contained in G_p seem almost random (which is why the term pseudorandom is employed). A different similarity graph can

be produced by using a different descriptor for detailing the data represented by the network nodes. Moreover, a different G_p will result, if a different distance measure D is applied.

4.4. Other models of the small world

In this section, we describe briefly three other models that have been proposed. One alternative to the view put forward by Watts and Strogatz is that the small-world phenomenon arises because there are a few nodes in the network which have unusually high coordination numbers or, which are linked to a widely distributed set of neighbors, and not because there are a few long-range connections in the otherwise short-range structure of a social network. Maybe the six degree of separation effect is because of a few people who are especially well connected. In (Kasturirangan 1999) it was mentioned that a septuagenarian salon proprietor in Chicago named Lois Weisberg is an example of precisely such a person. In a simple model of this kind of network, we start again with a one-dimensional lattice, but instead of adding extra links between pairs of sites, we add a number of extra vertices in the middle, connected to a large number of sites on the main lattice, and chosen at random. This model is similar to the Watts–Strogatz model in that the addition of the extra sites effectively introduces shortcuts between randomly chosen positions on the lattice, so it should not be surprising to learn that this model does display the small-world effect (see Figure 4). In fact, even in the case where only one extra site is added, the model shows the small-world effect if that site is sufficiently highly connected.

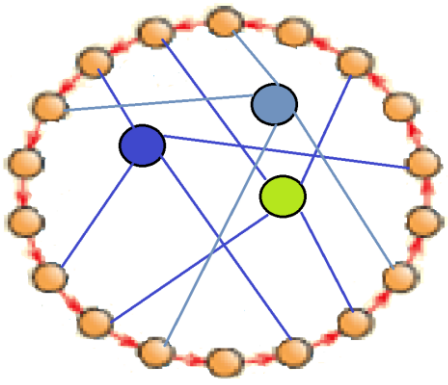


Figure 4: An alternative model of a small world, in which there are a small number of individuals who are connected to many widely-distributed acquaintances.

Albert *et al.* (Reka, Jeong, and Barabasi 1999) have proposed another alternative model of the small world. In this model, the Web is dominated by a small number of very highly connected sites, as described above, but they also found that the distribution of the coordination numbers of sites (the number of "hyperlinks" pointing to or from a site) was a power-law, rather than a bimodal one, in the previous model.

A third suggestion has been put forward by Kleinberg (Kleinberg 1999), who argue that a model such as that of Watts and Strogatz, in which shortcuts connect vertices arbitrarily far apart with uniform probability, is a poor representation of at least some real-world situations. He has stated that no algorithm exists, which is capable of finding such paths on networks of the Watts–Strogatz type, again given only local information. Kleinberg has proposed a generalization of the Watts–Strogatz model in which the typical distance traversed by the shortcuts can be tuned.

4.5. Communities

In biology, a group of organisms that are sharing the common environment is termed a community. The items that may be present and common in human communities are intent, belief, resources, etc. Indeed, in a community, helping each other plays an important role. Moreover, affecting the identity of the members and their degree of cohesiveness can be considered. Community in the network is a result of the action that divides the network nodes into groups within which the network connections are dense. By analyzing such groups, we are able to understand and visualize the structure of networks (Gladwell 1999). Community structure is a common property that can be seen in many networks. Newman has shown that the community has a high density within edges and a lower density elsewhere (M. Newman and Girvan 2004).

The study of community structure in networks has a long history. It is closely related to the ideas of graph partitioning in graph theory and computer science, and hierarchical clustering in sociology (Scott 2000). Graph partitioning is a problem that arises in, for example, parallel computing. Suppose we have a number n of intercommunicating computer processes, which we wish to distribute over a number g of computer processors. Processes do not necessarily need to communicate with all others, and the pattern of required communications can be represented by a graph or network in which the vertices represent processes and edges join process pairs that

need to communicate. The problem is to allocate the processes to processors in such a way, as roughly to balance the load on each processor, while at the same time minimizing the number of edges that run between processors, so that the amount of inter processor communication (which is normally slow) is minimized. In general, finding an exact solution to a partitioning task of this kind is believed to be an NP-complete problem, making it prohibitively difficult to solve for large graphs, but a wide variety of heuristic algorithms have been developed that give acceptably good solutions in many cases, the best known being perhaps the Kernighan–Lin algorithm (Kernighan and Lin 1970), which runs in time O(n3) on sparse graphs. A solution to the graph partitioning problem is how- ever not particularly helpful for analyzing and understanding networks in general. If we merely want to find if and how a given network breaks down into communities, we probably don't know how many such communities there are going to be, and there is no reason why they should be roughly the same size. Furthermore, the number of inter-community edges needn't be strictly minimized either, since more such edges are admissible between large communities than between small ones.

As far as our goals in this work are concerned, a more useful approach is that taken by social network analysis with the set of techniques known as hierarchical clustering. These techniques are aimed at discovering natural divisions of (social) networks into groups, based on various metrics of similarity or strength of connection between vertices.

There are two broad classes to build a community: an agglomerative technique and a divisive method. The communities are constructed based on various criteria such as similarity or strength of connections between members. At first, an agglomerative technique starts with an empty network. Then similarities or other metrics are computed, and edges are to connect the pairs of members with highest similarity. In this technique, the procedure can be halted at any point. The Concor algorithm of Breiger et al (Breiger, Boorman, and Arabie 1974) is a well-known example of an agglomerative clustering method.

In contrast to agglomerative methods, a divisive method uses removal of edges. It starts with the network and tries to find the least similar one with connected pairs.

Then edges are removed, providing smaller components in network communities (M. Newman and Girvan 2004).

References

Androutsos, Panagiotis, Dimitrios Androutsos, and Anastasios N Venetsanopoulos. 2006. “Small World Distributed Access of Multimedia Data.” *IEEE Signal Processing Magazine*: 142–153.

Bollobás, B. 1998. *Random Graphs*. http://link.springer.com/chapter/10.1007/978-1-4612-0619-4_7.

Breiger, Ronal L., Scott A. Boorman, and Phipps Arabie. 1974. “AN ALGORITHM FOR BLOCKING RELATIONAL DATA, WITH APPLICATIONS TO SOCIAL NETWORK ANALYSIS AND COMPARISON WITH MULTIDIMENSIONAL SCALING.” *Psychology and Education Series*.

Gladwell, M. 1999. “Six Degrees of Lois Weisberg.” *The New Yorker* 41: 52–64. http://croker.harpethhall.org/Must Know/Psychology/WeisbergGladwell.pdf.

Guare, J. 1992. *Six Degrees of Separation*. http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Six+Degrees+of+Separation#0.

Kasturirangan, Rajesh. 1999. “Multiple Scales in Small-world Networks.” http://18.7.29.232/handle/1721.1/5930.

Kernighan, B. W., and S. Lin. 1970. “A Efficient Heuristic Procedure for Partitoning Graphs.” *Bell System Technical* 49: 291–307.

Kleinberg, Jon. 1999. “The Small-world Phenomenon: An Algorithm Perspective.” *Proceedings of the Thirty-second Annual ACM*. http://dl.acm.org/citation.cfm?id=335325.

L. A. N. Amaral, A. Scala, M. Barth´el´emy, H. E., and Stanley. 2000. “Classes of Small-world Networks.” In *Natl. Acad. Sci.*

Marchiori, Massimo, and Vito Latora. 2000. “Harmony in the Small-world.” *Physica A: Statistical Mechanics and Its Applications*: 539–546. http://www.sciencedirect.com/science/article/pii/S0378437100003113.

Milgram, Stanley. 1967. “The Small-World Problem.” *Federal Legal Communications* 1 (1): 60–67.

Newman, M E J. 2000. “Models of the Small World.” *Statistical Physics* 101 (3-4): 819–841.

Newman, M E, C Moore, and D J Watts. 2000. “Mean-field Solution of the Small-world Network Model.” *Physical Review Letters* 84 (14) (April 3): 3201–4. http://www.ncbi.nlm.nih.gov/pubmed/11019047.

Newman, MEJ, and M Girvan. 2004. “Finding and Evaluating Community Structure in Networks.” *Physical Review*: 1–16. http://pre.aps.org/abstract/PRE/v69/i2/e026113.

Newman, MEJ, and DJ Watts. 1999. “Scaling and Percolation in the Small-world Network Model.” *Physical Review E* 87501. http://pre.aps.org/abstract/PRE/v60/i6/p7332_1.

Reka, Albert, Hawoong Jeong, and AL Barabasi. 1999. “Diameter of the World Wide Web.” *Nature* 401 (September): 398–399. http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Diameter+of+the+World-Wide+Web#0.

Scott, J. 2000. *Social Network Analysis: A Handbook*.

Watts, D J, and S H Strogatz. 1998. “Collective Dynamics of ‘Small-world’ Networks.” *Nature* 393 (6684) (June 4): 440–2. doi:10.1038/30918. http://www.ncbi.nlm.nih.gov/pubmed/9623998.

Chapter 7

The Proposed Architecture

1. Introduction

Representations and access mechanisms in multimedia content are more complicated than in text-based data. To represent low-level features such as colors, textures, objects, typically we use multidimensional vectors. On the other hand, semantic information in the form of high-level features is quite fuzzy. Thus, any data management framework for multimedia data should be supported both by multidimensional representations and by semantic information.

Another factor that is important for the development of a distributed multimedia content system is efficiency; it aims at minimizing resource consumption while providing the most relevant results to a user's query in the shortest time. In this framework, efficient content indexing is a key issue to be considered. This includes whether indexing is based on a fixed, or on a variable set of algorithms, or whether algorithms are executed over the entire data set, or only over a selected sub-collection of it. Selecting a distributed, or a centralized, indexing approach is another factor that should be considered. Finally, where indexing is executed also is another important issue.

Totally, to design a distributed content search framework, attention should be paid to the following key issues:

- Use a fixed or a variable set of indexing algorithms;
- Algorithms execute at acquisition time, or at users' query;
- Use distributed or centralized indexing execution;
- Indexing is done in the same location with content, or in a special indexation server where content is transferred;
- Filtering content is accomplished before, or after, indexing;
- Selecting algorithms, or not, according to user's query;
- Selecting the relevant remote servers, or not, according to the user's query.

2. Related Works

There are a lot of methods which have been proposed for creating distributed frameworks and platforms for multimedia systems (Siqueira, 1998)(Müller, Müller, & Squire, 2000). Amoretti (Amoretti, Bianchi, & Conte, 2004) has provided an ontology-based Grid Service for searching multimedia content. In this work, too many local search services are connected to a global search service. Brut (Brut, Codreanu,

& Dumitrescu, 2011) has designed a framework that uses a central server and many remote ones. In this approach, a query is executed only on a set of relevant servers based on user queries, using semantic processing and available knowledge about the distributed servers, the multimedia content and the indexing algorithms. In (Chatterjee, Sadjadi, & Chen, 2010) two components are used to perform distributed query processing; a multimedia application interface, that is a global query processing interface, and a distributed query content-retrieval engine. Laborie (Sebastien Laborie, Manzat, & Se, 2009) has designed a framework for distributed multimedia that also uses a central server and many remote servers. It transfers only a concise version of the distributed metadata to the central server. When a set of servers has been selected by the central server as one that includes data likely to match a particular user query, the query can be processed locally on these servers.

In all these approaches, a centralized knowledge repository is used to feed the algorithms and perform content filtering. As was above mentioned, decentralization as well as efficiency are important and desired factors. Subsequently, some researchers have proposed approaches based on the small world network principle (Milgram, 1967)(Guare, 1992) due to which a network has a short distance between nodes and a high decentralization ability. In general, small world networks mimic social networks. In these networks, community plays an important role. Androutsos (Androutsos, Androutsos, & Venetsanopoulos, 2006) has applied the small network concept to his framework, but he has not paid the necessary attention to the community.

In the following, we propose a distributed framework for content management and search using small-world networks. It is based on the community concept, also using "fuzzy similarity" to build cliques of data nodes, thus, locally performing most of the tasks.

3. The Proposed Architecture

In the following, we describe an architecture that fulfills the requirements of improved performance of the search and retrieval process and of the indexing framework. This architecture has a two-layered structure. The lower layer provides the data resources, as well as specific services related to the storage system and the metadata repository. The upper layer includes high-level components, such as communities and small network links. Small-words are constructed by grouping

nodes according to their fuzzy ontological similarity. In particular, the small-world layer organizes multimedia nodes in such communities, allowing nodes to efficiently locate areas of ontology similarity. The communities facilitate creation of the ontological small worlds. Accordingly, queries can be executed on the nodes that are likely to contain the relevant resource; as a consequence, a lower network load and a better search performance is achieved.

To make a community, firstly, we compute the community coefficient (CC), defined as follows:

$$CC(\text{Head, New Node}) = \begin{cases} 1 & \text{Sim(Head, New node)} >= th \\ 0 & \text{Otherwise} \end{cases} \quad (1)$$

The threshold (th) is determined according to the amount of similarity between nodes of the network. The amount of threshold plays an important role in framework structure because a small value of it causes creation of a big community. In the opposite, a big value of it results in a small community. After CC coefficient computing, if the CC value is equal to one, the node is inserted to the current community. Otherwise, it move to the next community head.

Figure 1 shows the proposed two-layer structure, including the physical layer and the small-world layer.

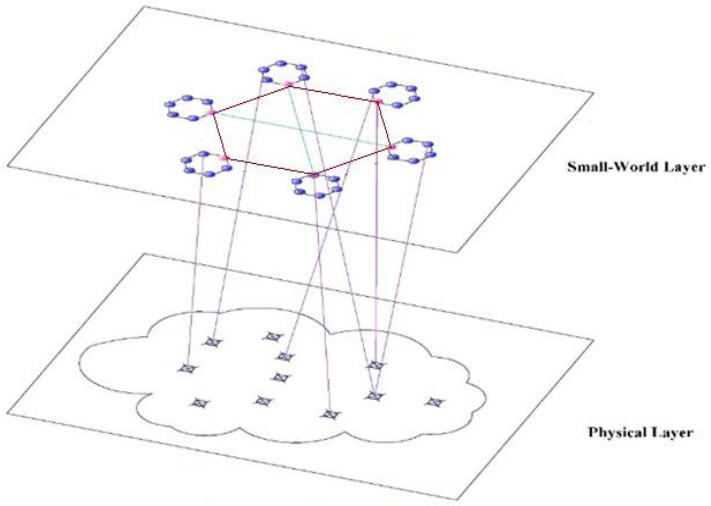


Figure 1: The two-tier structure

3.1. The Physical Layer

The physical layer comprises multimedia as well as metadata collections. These collections can be organized in many ways. In large scale distributed indexation of multimedia Objects (LINDO) system (Brut et al., 2011), there are some remote servers that store and index all acquired multimedia information. Some modules are used to manage all needed tasks such as the *Storage Manager, t*he *Access Manager, t*he *Feature Extractor Manager, and t*he *Metadata Engine.* Laborie (Sebastien Laborie et al., 2009) proposed a scheme in which parts of every server store multimedia information. Moreover, a multimedia collection is stored on a server dedicated to acquire remote site information. A set of extractors is applied to a given piece of multimedia content returning a set of content metadata. The latter contain information about the media characteristics, while a metadata collection contains all content metadata, describing objects of the multimedia collection. Chatterjee (Chatterjee et al., 2010) proposed an architecture that includes a set of data nodes. Each data node has a multimedia database management system embedded in it. It also has a GridFTP server that takes care of the physical transfer of multimedia objects from one node to another. The data is basically stored in a data server. The multimedia database framework is divided into four major components: a multimedia interface, a core DBMS engine, a content-retrieval engine and a high-level relationship manager. These four components interact with each another to achieve the major functionalities, including queries, and updates.

In our framework, the physical layer can include different storing policies for each node, as shown in Figure 2.

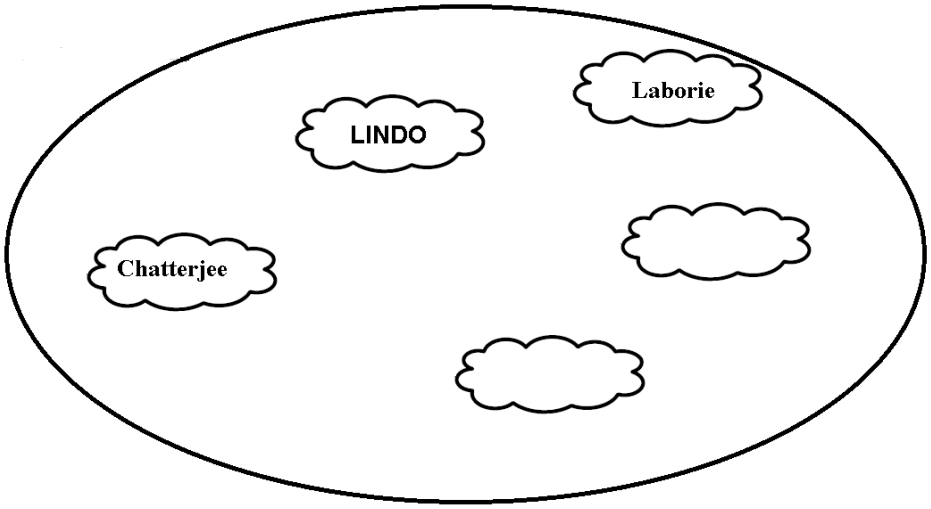


Figure 2: Physical Layer.

It is, however, necessary to provide a mapping to meet the requirements of a small-network layer.

3.2. The Small Network Layer

With the help of this layer, nodes are connected according to their ontological similarity, forming small-world network communities. The topology of each community overlay can be flexible: if the community is large, nodes will be further broken up into small subcommunities to distribute multimedia nodes efficiently into the community, according to their fuzzy ontology similarity. Each community has a node, as community index, that is used to insert and retrieve nodes.

The layer's distributed nature stems from the fact that it attempts to exploit the small world phenomenon; i.e., all members of a social network retain information about respective small groups of peers and apply it to perform media indexing. As can be seen in Figure 1, we have used the small-world network properties of the layer. The nodes are located in this layer according to the amount of similarity they have with the head of each community. Clustering based on this criterion will result in improved system performance. Every community has a head that is used to enter and search content in the community. The properties of small worlds are set by the heads. The latter locally store descriptions of themselves and their peers. The peers can be selected using a distance measure function, based on multiple criteria of similarity. In addition, this layer attempts to make each community behave as an independent entity in the network in which it participates. Since a small-world network allows only few step searching between nodes (6 according to Milgram), search time is

significantly reduced. To understand better the structure, we explain the insert and search methods in the following.

3.3. Small World Layer Construction

The first node is considered as head of each community. When adding a new node, we compare it to the head. The following situations may be happen:

- Similarity between head and new node is greater than a threshold:

  It will be added to the community. Based on the similarity degree of nodes within the community, nodes can be sorted in descending order.

- Similarity between head and new node is less than a threshold:
  Relevant heads are compared to the new node. The following situations may occur:
  - There are communities, in which similarity degrees are greater than a threshold. In this case, the new node is added to the community with the highest similarity.
  - There is no community, in which a similarity degree is greater than a threshold. In this case, a new community, with this node as head, is created. Then, all other communities are sorted based on their degree of similarity. According to the number of peers, the unique number associated with the new community can compute its peers. These steps are shown in detail in Figure 3.

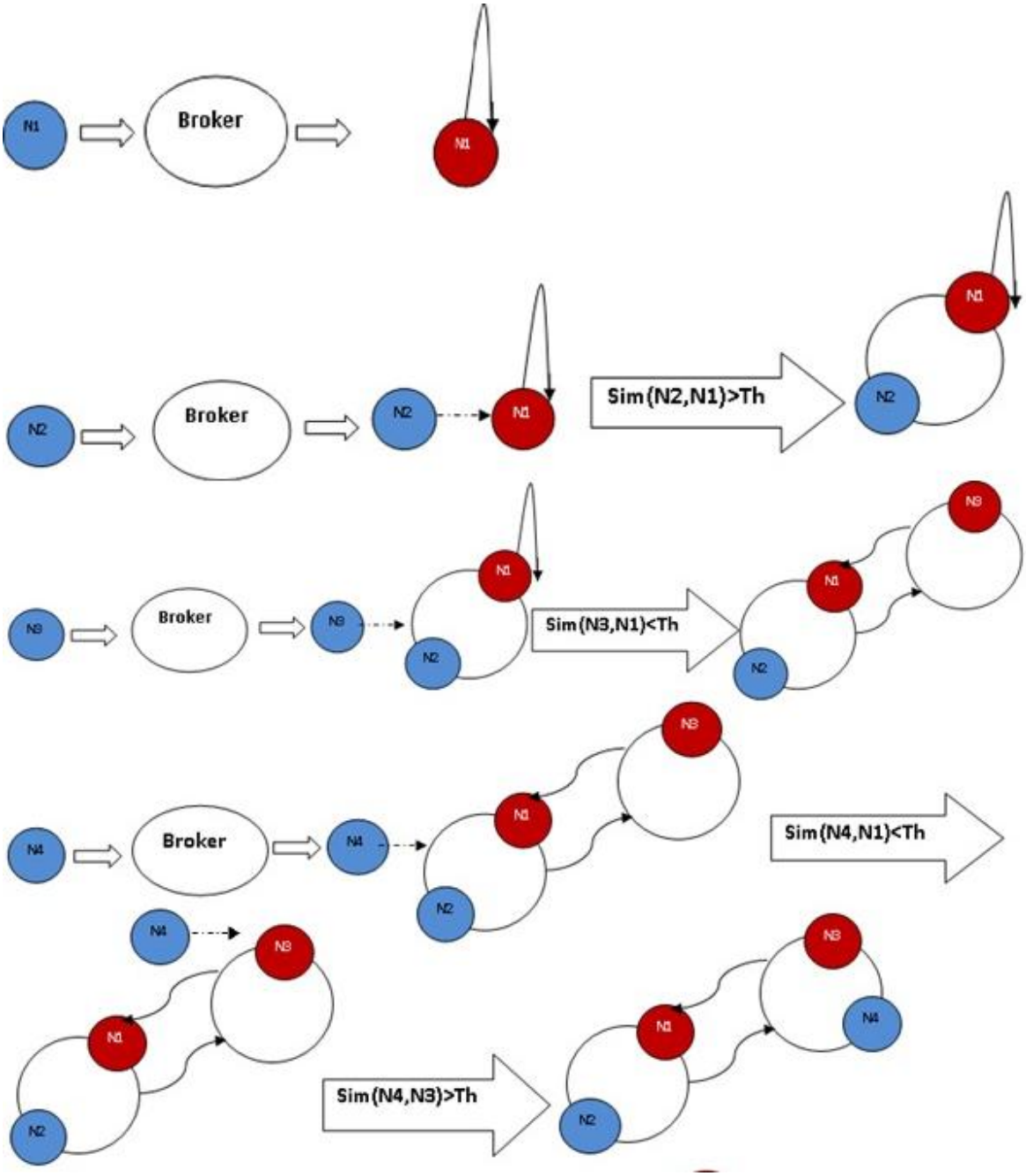


Figure3: The Small World Layer Making

References

Amoretti, M., Bianchi, D., & Conte, G. (2004). An Ontology-based Grid Service for Multimedia Search. *software GRID*, 1–11. Retrieved from http://dsg.ce.unipr.it/userfiles/file/publications/2004/amorettiOntologyGS.pdf

Androutsos, P., Androutsos, D., & Venetsanopoulos, A. N. (2006). Small World Distributed Access of Multimedia Data. *IEEE Signal Processing Magazine*, 142–153.

Brut, M., Codreanu, D., & Dumitrescu, S. (2011). A distributed architecture for flexible multimedia management and retrieval. *Database and Expert*, 249–263. Retrieved from http://www.springerlink.com/index/K503V7L931571175.pdf

Chatterjee, K., Sadjadi, S., & Chen, S. (2010). A Distributed Multimedia Data Management over the Grid. *Multimedia Services in Intelligent*. Retrieved from http://www.springerlink.com/index/H568401887Q715UH.pdf

Guare, J. (1992). *Six degrees of separation*. Retrieved from http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Six+Degrees+of+Separation#0

Milgram, S. (1967). The Small-World Problem. *Federal Legal Communications*, *1*(1), 60–67.

Müller, H., Müller, W., & Squire, D. (2000). An open framework for distributed multimedia retrieval. *Recherche d'Informations Assistée par Ordinateur (RIAO) Computer-AssistedInformation Retrieval*. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.25.7310

Sebastien Laborie, Manzat, A., & Se, F. (2009). Managing and Querying Distributed Multimedia Metadata. *IEEE MultiMedia*, 12–21.

Siqueira, F. (1998). A Framework for Distributed Multimedia Applications based on CORBA and Integrated Services Networks. *Distributed Systems Group, Department of Computer*, 1–34. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.55.4533&rep=rep1&type=pdf

Chapter 8

Experimental study and Evaluation

1. Introduction

A graph is used to represent the topology of a distributed system where nodes are as the processes, and links are considered as communication channels. Accordingly, various graph theoretic problems that have numerous applications in communication and networking are posed in distributed systems. In the following, some important problems considered.

The first relates to routing in a communication network. When a message is sent from node i to a nonneighboring node j, the message is routed by the intermediate nodes according to the information stored in the local table. This is termed hop-by-hop or destination-based routing. For a messages to reach its destination in the smallest number of hops or with minimum delay is an important problem. In addition, the minimum hop routing is equivalent to computing the shortest path between a pair of nodes using locally available information.

The second problem is related to the amount of space required to store a routing table in a network. Without any optimization, O (N) is the space requirement for N nodes of network, which is a matter of concern with regard to the explosive growth of the Internet.

Broadcasting in a network whose topology is represented by a connected graph is the third problem. Transmission of messages in an uncontrolled way results in a flooding, which wastes communication bandwidth.

Routing is a fundamental problem in networks. The major issue is to discover and maintain an acyclic path from the source of a message to its destination. Each node has a routing table that determines how to route a packet so that it reaches its destination. The routing table is updated when the topology changes. A path can have many attributes: these include the number of hops or the end-to-end delay. For efficient routing, a simple goal is to route a message using minimum number of hops. An argument against the use of the number of messages as a measure of time complexity could be that in a purely asynchronous message-passing model with arbitrarily large message propagation delays, absolute time plays no role. However, in models with bounded channel delays and approximately synchronized clocks, a useful alternative metric is the total time required to execute an instance of the algorithm. One can separately estimate the average and the worst-case complexities.

2. Experiment 1

2.1. Evaluation environment

To evaluate the proposed framework, we have used the image database of Lotus Hill Institute (LHI). The database provides large scale annotated ground truth data including extraction of edges, contours, contour attributes, segmentation, grouping, occluded contour completion, text, and object category recognition, 3D frames, UAV images, Google Earth images, video and cartoons. Part of the ground truth data are packed for various vision tasks and released one-by-one in xml format, together with respective Matlab code for reading and visualizing the annotation.

2.2. Evaluation method and results

We used the images of the database to test the proposed method. To build the community, we have used the fuzzy XML similarity algorithm we presented in (Javadi-Moghaddam & Kollias, 2014).

As was mentioned before, one of the most important factors in evaluating the performance of the distributed system is the number of hops to locate an object; a hop corresponds to a move from one community, or cluster, to another. Reducing the number of hops is equivalent to improving the system's performance.

On the other hand, each node in the framework has some peers that are selected based on specific criteria. First, we have compared the number of hops when we select peers of maximum, or, of minimum similarity. As can be seen in Figure 1, when using the maximum factor, provides a lower number of hops than when using the minimum one.

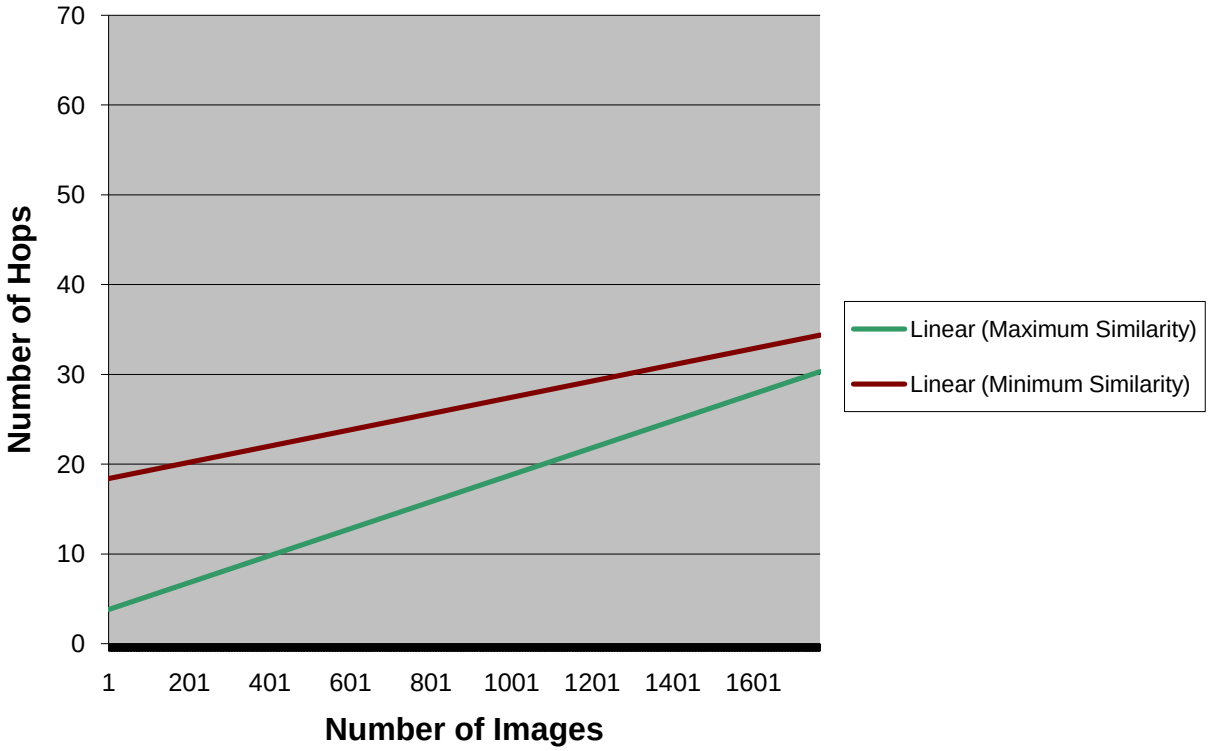


Figure 1: Comparison of performance when peers are selected based on maximum (or minimum) similarity.

Secondly, we have measured the effect of an increased number of peers to the required number of hops. The result is shown in Figures 2 and 3. It can be seen that performance is improved when peers are added.

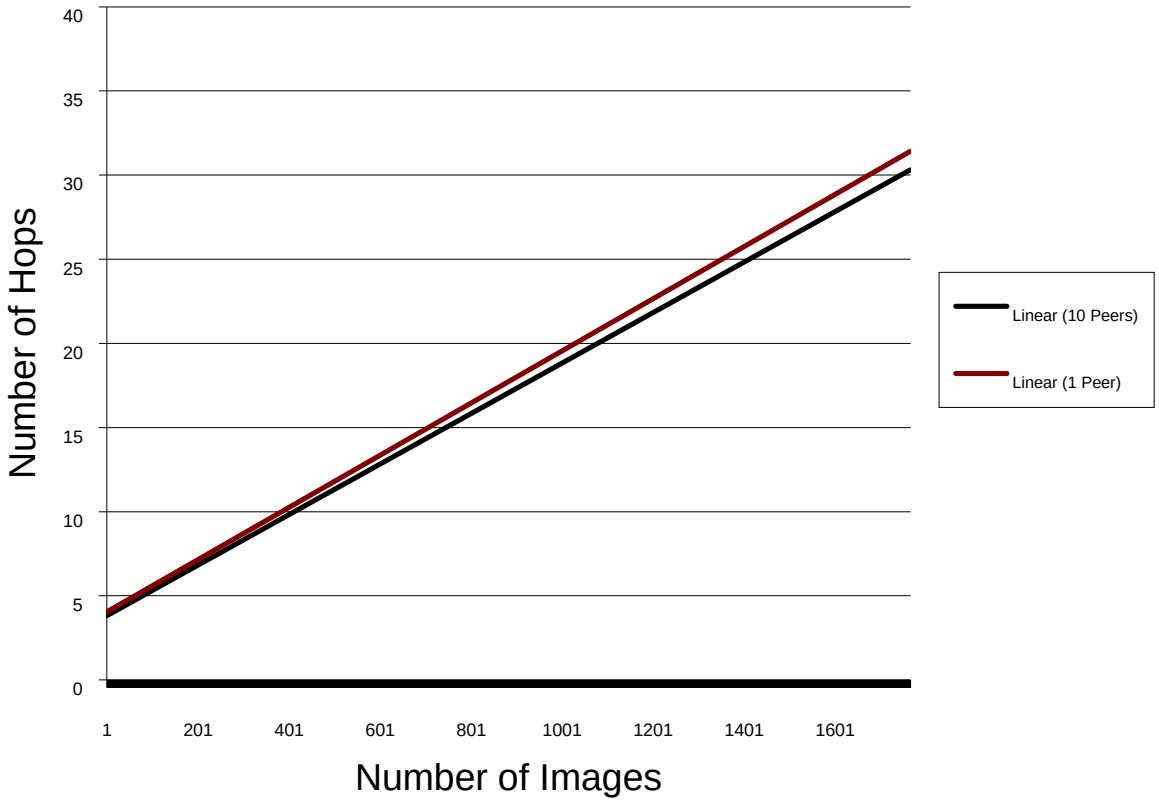


Figure 2: The effect of increasing number of peers on the required number of hops.

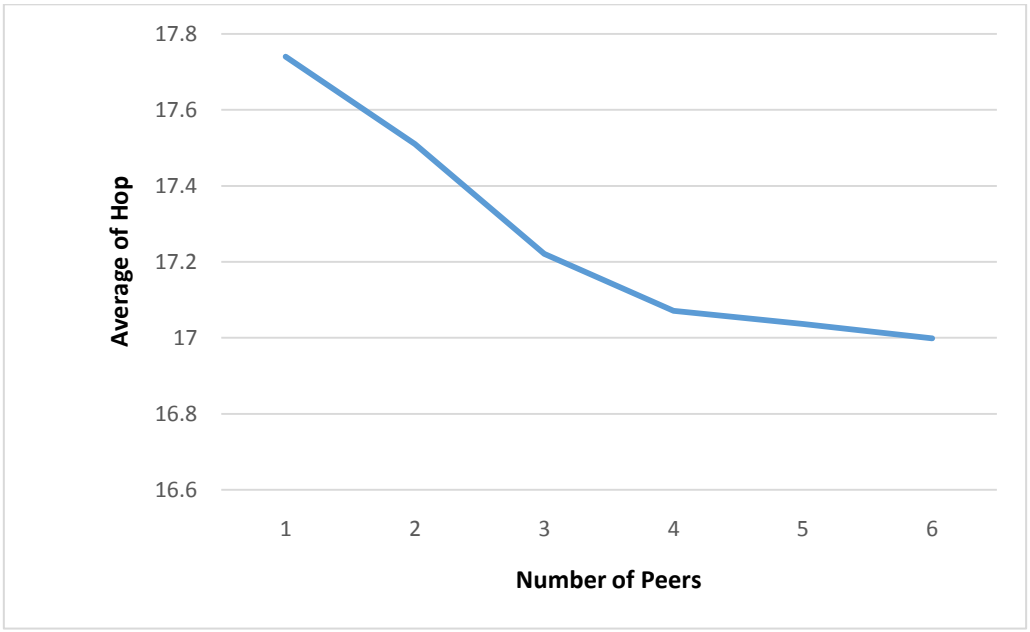


Figure 3: The average number of hops when increasing the number of peers.

Finally, we have computed the required number of hops when using a normal clustering framework compared to the proposed framework. It can be seen that the

proposed method achieves a better performance, which gets higher, as the number of image increases (see Figure 4).

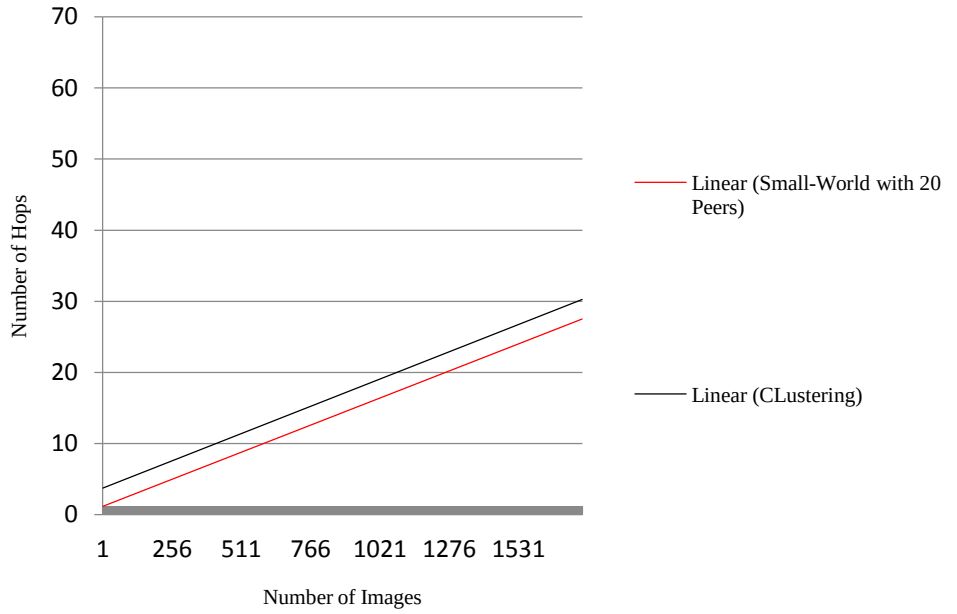


Figure 4: The required number of hops when using a normal clustering framework and the proposed framework.

3. Experiment 2

3.1. Evaluation environment

The dataset of experiment 2 was a set of 10000 homogeneous XML documents taken from the dblp DTD. The DBLP computer science bibliography contains metadata of over 1.8 million publications, written by over 1 million authors in several thousands of journals or conference proceedings series. For computer science researchers the DBLP web site is a popular tool to trace the work of colleagues and to retrieve bibliographic details when composing the lists of references for new papers. Ranking and profiling of persons, institutions, journals, or conferences are the other controversial uses of DBLP. The bibliographic records are contained in a huge XML file. Many researchers simply need non-toy files to test and evaluate their algorithms. It is easy to derive several graphs like the bipartite person publication graph, the person-journal or person-conference graphs, or the coauthor graph, which are examples of a social network. Methods for analysis and visualization of these medium

sized graphs are reported in many papers. To evaluate the proposed approach, we use the protocol develop by(Alguliev, Aliguliyev, & Alekperova, 2014).

The DBLP data set is available in http://dblp.uni-trier.de/xml/. The file dblp.xml contains all bibliographic records which make DBLP. It is accompanied by the data type definition file dblp.dtd. dblp.xml has a simple layout:

<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE dblp SYSTEM "dblp.dtd">

<dblp>

record 1

...

record n

</dblp>

The header line specifies ISO-8859-1 (\Latin-1") as the encoding, but in fact the file only contains characters <128, i.e. pure ASCII. All non-ASCII characters are represented by symbolic or numeric entities. The symbolic entities like é for the character '_e' are declared in the DTD. Numeric entities like é should be understood by any XML parser without declaration. In practice there are some obstacles in parsing the large XML file which cost us a lot of time: The SAX parser contained in the Java standard distribution has a limit for handling symbolic entities. When starting the Java virtual machine the option 'DentityExpansionLimit' has to be set to a large number. The Xerces-J from the Apache XML project reads dblp.xml without any problem. The DBLP FAQ3 reports more details. The XML root element <dblp> contains a long sequence of bibliographic records. The DTD lists several elements to be used as a bibliographic record:

<!ELEMENT dblp (articlejinproceedingsj
proceedingsjbookjincollectionj
phdthesisjmastersthesisjwww)>*

<article key="journals/cacm/Szalay08"
mdate="2008-11-03">
<author>Alexander S. Szalay</author>
<title>Jim Gray, astronomer.</title>
<pages>58-65</pages>

<year>2008</year>

<volume>51</volume>

<journal>Commun. ACM</journal>

<number>11</number>

<ee>http://doi.acm.org/10.1145/

1400214.1400231</ee>

<url>db/journals/cacm/

cacm51.html#Szalay08</url>

</article>

3.2. Evaluation method and results

The experiment aims at providing an efficient way to search for xml files in the dblp dataset. The performance evaluation is done on a computer with Intel 3Ghz CPU and 4GB RAM.

Clustering of the files has been achieved using the concept of fuzzy similarity, which was above described. In the presented results, shortcuts were built after clustering execution. Three parameters have been considered to evaluate the proposed system: the number of hops (similarly to Experiment 1), the number of shortcuts, and the number of files. In this experiment, at first we varied the data size from 500 to 10,000 xml files. All files were introduced in the clusters and then there was a search for them. Figure 5 shows the evolution of the number of total hops with respect to the data set size at different shortcuts (peers).

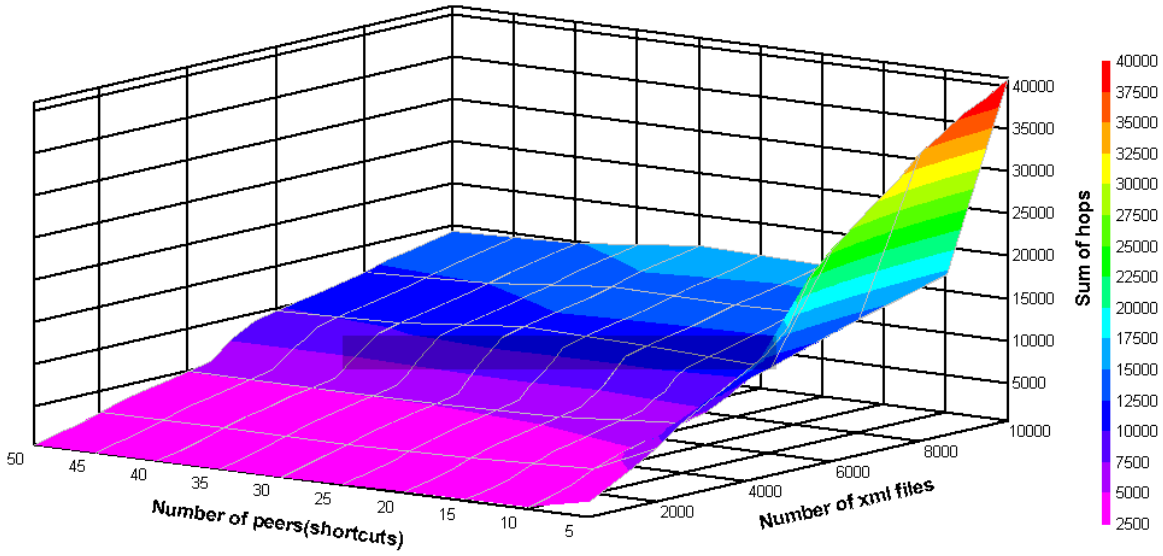


Figure 5: Dependence of the hops on number of peers and dataset size.

It is clear from Figure 5 that an increase of the data set size leads to an increase in the number of hops. This finding is in agreement with previous researches (Alguliev et al., 2014)(Chatterjee, Sadjadi, & Chen, 2010)(Crestani & Markov, 2013). Moreover, increasing the number of peers from 5 to 10 decreases the respective number of hops when we deal with a small dataset; if, however, we increase the dataset size, this effect is reduced. On the other hand, the effect when changing the number of peers from 10 to 50 is not significant when the dataset size is small. On the contrary, it gets significant in larger datasets, e.g., when 10000 files are considered. A regression equation that describes the chart can be computed as follows:

$$Z = 0.0043 + 2.152X - 197.122Y \qquad R^2 = .7 \qquad (1)$$

where X, Y, and Z are number of files, number of peers, and sum of hops respectively.

Finally, the required number of hops when using a normal clustering framework has been investigated and compared to the proposed framework. It can be seen that the

proposed method shows a much better performance, which requires less hops as the number of xml files increases (see Figure 6).

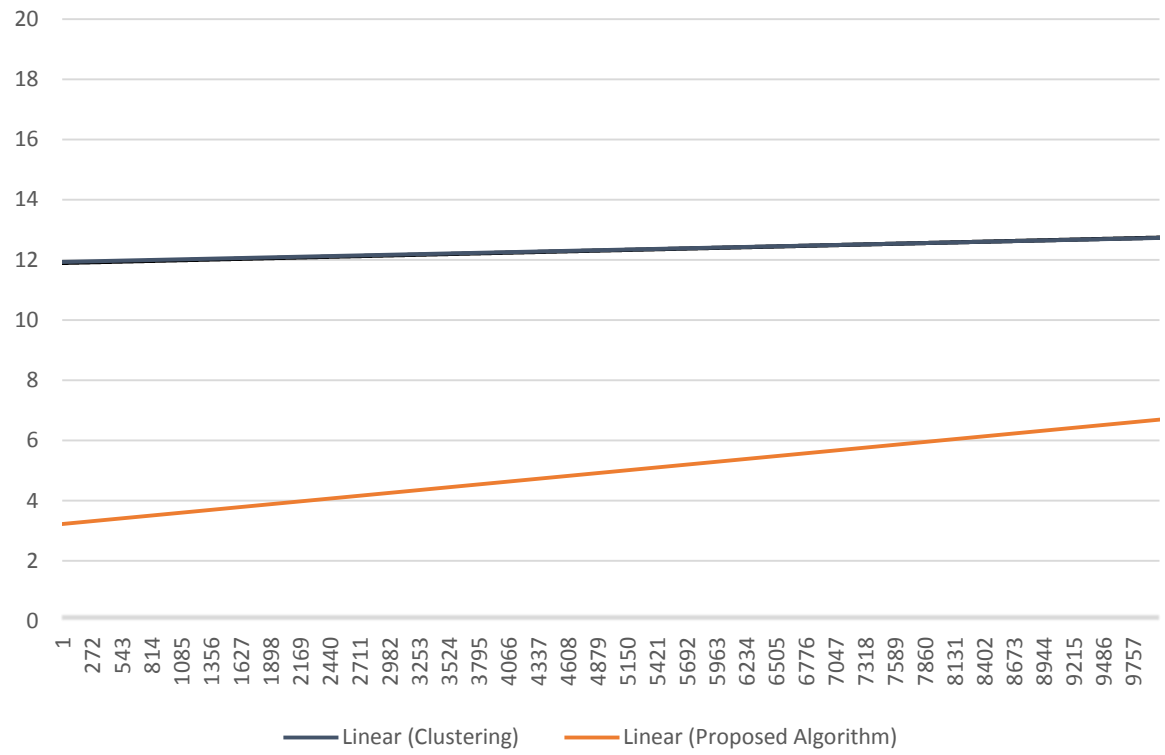


Figure 6: The required number of hops when using the proposed method compared to a normal clustering framework.

References

Alguliev, R. M., Aliguliyev, R. M., & Alekperova, I. Y. (2014). Cluster approach to the efficient use of multimedia resources in information warfare in wikimedia. *Automatic Control and Computer Sciences*, *48*(2), 97–108.

Chatterjee, K., Sadjadi, S., & Chen, S. (2010). A Distributed Multimedia Data Management over the Grid. *Multimedia Services in Intelligent*. Retrieved from http://www.springerlink.com/index/H568401887Q715UH.pdf

Crestani, F., & Markov, I. (2013). Distributed information retrieval and applications. *Advances in Information Retrieval, 7814*, 865–868.

Javadi-Moghaddam, S.-M., & Kollias, S. (2014). A Fuzzy Similarity Measure for XML Documents. *International Journal of Information Technology & Computer Science (IJITCS)*, *13*(2), 9–17. Retrieved from http://ijitcs.com/volume 13_No_2/Seyyed+Mohammad.pdf

Worring, M., & Snoek, C. (2006). Semantic Indexing and Retrieval of Video. *ACM Multimedia*, *13*.

Chapter 9

Conclusion & Future Work

We have presented a distributed content search framework based on a two-tier architecture, consisting of a physical and a small-world layer. We have shown that this scheme provides efficient content search. In our experiments, we compared the performance of the proposed small-world framework to a clustering system. We have shown that the small-world network requires a much smaller number of hops and subsequently results in lower content search time.

1. Conclusions

The proposed method can be of great importance for a variety of image and video retrieval and indexing tasks. A great diversity of methodologies can gain from adopting the presented framework. In particular, the focus of (Urruty, Belkouch, & Djeraba, 2005) has been on reducing the data space in the considered clusters. However, in this approach, no relation between the clusters is taken into account, while the advantage of our framework is the small-world network which is built between the clusters. Indexing approaches for image or video files have been presented in (Idris & S. Panchanathan, 1997)(Worring & Snoek, 2006). However, in our approach we go one step forward, using an ontology-based fuzzy similarity, based on both semantic and structural issues. The centralization used in these techniques (Urruty et al., 2005)(Kulkarni & Callan, 2010) for dealing with a distributed framework results in a reduction of the search efficiency. Obviously, gathering information within the distributed system causes a computational overload. This does not occur in our approach, since, in each node of the distributed framework, we can calculate resource selection without gathering information from other nodes - we build it with regard to the shortcut list locally.

Other works (e.g., (Frank Hopfgartner & Jose, 2010)) adopt (probably semantic based) user modeling techniques to capture users' evolving information needs. Such techniques can be considered as particular cases of our approach, if we focus on semantic similarity based on the interests of users.

One of the features of the proposed system is its ability to take into account semantic similarity between resources, possibly expressed in ontological form. There is a great number of publications related to knowledge technologies and multimedia content service provision (Stamou & Kollias, 2005). Recent advances include creation of the

"Billion Triple Challenge" ("Billion Triple Challenge," 2012) aiming at investigating the scalability of applications as well as the capability to deal with the specifics of data that has been crawled from the public web. Extending the proposed approach in this framework constitutes a topic of future work.

Application of the proposed method in a real cultural heritage environment ("Europeana Fashion project CIP-ICT-PSP," 2015) can also be a topic of future work.

2. Future Work

Development of a principled framework to support multimedia indexing and retrieval on large-scale databases is challenging. While we have presented in this thesis some promising solutions, there remain many open issues and exciting opportunities for further advancement.

The core functionality works, and has been tested in some small case studies. First, we need to evaluate our framework on a large-scale environment such as, a Grid system. As for future work, we plan to continue to evaluate this proposed framework with metadata based on RDFS and OWL, and more expressive query languages.

On the other hand, this study focuses on an approach which describes how the cluster is build. But it does not going deep inside the clusters. We are going to continue the study with implementation of different search algorithm within the clusters, such as B^+-Tree, B-Tree, Hash, and others. Implementation and comparison of these algorithms based on similarity are important issues for future research.

References

Billion Triple Challenge. (2012). http://doi.org/http://challenge.semanticweb.org

Europeana Fashion project CIP-ICT-PSP. (2015). http://doi.org/http://www.europeanafashion.eu

Frank Hopfgartner, & Jose, J. M. (2010). Semantic User Modelling for Personal News Video Retrieval. *Advances in Multimedia Modeling, 5916*, 336–346.

Idris, F., & S. Panchanathan. (1997). Review of Image and Video Indexing Techniques. *Journal of Visual Communication and Image Representation*, *8*(2), 146–166.

Kulkarni, A., & Callan, J. (2010). Document Allocation Policies for Selective Searching of Distributed Indexes. In *ACM international conference on Information and knowledge management (CIKM)* (pp. 449–458).

Stamou, G., & Kollias, S. (2005). *Multimedia Content and the Semantic Web*. Wiley, UK.

Urruty, T., Belkouch, F., & Djeraba, C. (2005). KPYR: An Efficient Indexing Method. In *IEEE International Conference on Multimedia and Expo (ICME)* (pp. 1448 – 1451).

Publications

Javadi-Moghaddam, S.-M., & Kollias, S. (2012). The Important Role of Validation in Knowledge Intensive. In UKCBR.

Javadi-Moghaddam, S.-M., & Kollias, S. (2014). A Fuzzy Similarity Measure for XML Documents. International Journal of Information Technology & Computer Science (IJITCS), 13(2), 9 – 17.

Javadi-Moghaddam, S.-M., & Kollias, S. (2016), A Distributed Framework for Content Search Using Small World Communities. International Journal of advanced Computer Science and Application (IJACSA), 7(1), 517-525.