ΑΠΟΤΙΜΗΣΗ
ΟΜΟΙΟΤΗΤΑΣ ΔΕΔΟΜΕΝΩΝ ΣΕ ΠΟΛΥΔΙΑΣΤΑΤΟΥΣ ΧΩΡΟΥΣ

Η
ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ ΕΞΕΙΔΙΚΕΥΣΗΣ

Υποβάλλεται στην

ορισθείσα από την Γενική Συνέλευση Ειδικής Σύνθεσης
του Τμήματος Πληροφορικής
Εξεταστική Επιτροπή

από τον

Ρογκάκο Γεώργιο

ως μέρος των Υποχρεώσεων

για τη λήψη

του

ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΔΙΠΛΩΜΑΤΟΣ ΣΤΗΝ ΠΛΗΡΟΦΟΡΙΚΗ

ΜΕ ΕΞΕΙΔΙΚΕΥΣΗ ΣΤΟ ΛΟΓΙΣΜΙΚΟ

Ιούλιος 2010

DEDICATION

This Thesis is dedicated to my family for their support since the beginning of my studies.

ACKNOWLEGMENTS

First of all would like to thank my supervisor Dr. Panos Vassiliadis for his support and the motivation that I received in order to fulfill my research.

I would also like to thank my colleagues for their help and support. Specifically, I am thankful to Eftychia Baikousi for helping me at the beginning of my research.

Finally, many thanks must be given to the people that spent some of their valuable time in order to answer to our user studies.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ΕΚΤΕΝΗΣ ΠΕΡΙΛΗΨΗ ΣΤΑ ΕΛΛΗΝΙΚΑ

Γεώργιος Ρογκάκος του Θωμά και της Ζωίτσας. MSc, Τμήμα Πληροφορικής, Πανεπιστήμιο Ιωαννίνων, Ιούλιος 2010. Αποτίμηση Ομοιότητας Δεδομένων σε Πολυδιάστατους Χώρους.
Επιβλέποντας: Παναγιώτης Βασιλειάδης.

Πόσο μοιάζουν δύο κύβοι δεδομένων; Με άλλα λόγια το ερώτημα που τίθεται είναι το εξής : Δοθέντων δύο συνόλων από σημεία ενός πολυδιάστατο χώρου με ιεραρχίες, ποια είναι η απόσταση ανάμεσα στα δύο σύνολα; Λόγω του μεγάλου πλήθους των δεδομένων που συναντάμε, είναι θεμελιώδες να παρέχουμε μέτρα ομοιότητας για σύνολα πολυδιάστατων δεδομένων.

Το συγκεκριμένο πρόβλημα είναι γενικό καθώς συναντάται σε αρκετές εφαρμογές στα πλαίσια της εξόρυξης πληροφορίας πολυμέσων, σε επιστημονικές βάσεις δεδομένων και σε ψηφιακές βιβλιοθήκες. Σε τέτοιες εφαρμογές, δημιουργείται η ανάγκη για αποθήκευση εξαιρετικά μεγάλου όγκου ετερογενών δεδομένων. Αυτό οδηγεί στην ανάγκη για αναζήτηση ομοιότητας σε δεδομένα τέτοιου τύπου. Για το λόγο αυτό, είναι χρήσιμο να βρούμε μέτρα ομοιότητας που να ικανοποιούν τις ανθρώπινες ανάγκες σε εφαρμογές που αφορούν αναζητήσεις σε υπολογιστικά συστήματα.

Στην παρούσα διατριβή μελετάμε ένα σύνολο συναρτήσεων απόστασης που μπορούν να χρησιμοποιηθούν για την αποτίμηση ομοιότητας δεδομένων σε πολυδιάστατους χώρους με ιεραρχίες διαστάσεων. Η κατηγοριοποίηση αυτού του συνόλου συναρτήσεων απόστασης οργανώνεται με βάση τις ιδιότητες των ιεραρχιών των διαστάσεων, των επιπέδων και των τιμών τους. Ειδικότερα, η κατηγοριοποίηση των συναρτήσεων οργανώνεται ως εξής: Πρώτον, περιγράφουμε τις συναρτήσεις

απόστασης που υπολογίζουν την απόσταση μεταξύ δύο τιμών της ίδιας διάστασης ενός πολυδιάστατου χώρου, δεύτερον περιγράφουμε συναρτήσεις απόστασης για τον υπολογισμό της απόστασης μεταξύ σημείων ενός πολυδιάστατου χώρου και τέλος περιγράφουμε συναρτήσεις που υπολογίζουν την απόσταση μεταξύ δύο συνόλων πολυδιάστατου χώρου.

Για το σκοπό του προσδιορισμού των συναρτήσεων που ικανοποιούν καλύτερα τις ανάγκες των χρηστών, οργανώσαμε δύο πειράματα με χρήστες. Το πρώτο πείραμα αφορά την πιο προτιμητέα συνάρτηση απόστασης από τη κατηγορία των συναρτήσεων απόστασης μεταξύ δύο τιμών της ίδιας διάστασης ενός πολυδιάστατου ιεραρχικού χώρου δεδομένων (πιθανά όμως, σε διαφορετικά επίπεδα της ιεραρχίας της διάστασης) δεδομένων. Το βασικό συμπέρασμα αυτού του πειράματος ήταν ότι η πιο προτιμητέα συνάρτηση απόστασης μεταξύ δύο τιμών μιας διάστασης, είναι εκείνη που χρησιμοποιεί το ελάχιστο μονοπάτι που συνδέει τις δύο τιμές και τον κοινό τους πρόγονο στην ιεραρχία της διάστασης.

Λαμβάνοντας υπόψη τα συμπεράσματα του πρώτου πειράματος χρηστών οργανώσαμε το νέο πείραμα με χρήστες. Το δεύτερο πείραμα είχε σκοπό την ανακάλυψη της πιο προτιμητέας συνάρτησης απόστασης μεταξύ των συναρτήσεων *Κοντινότερου Συνδεδεμένου* (η οποία αποτιμά την απόσταση δύο κύβων σαν ένα ζυγισμένο άθροισμα των επιμέρους ελαχίστων αποστάσεων των κελιών τους) και *Hausdorff* (η οποία αποτιμά την απόσταση δύο κύβων σαν τη μέγιστη των ελαχίστων, των αποστάσεων των κελιών τους) από την κατηγορία των συναρτήσεων απόστασης μεταξύ δύο κύβων δεδομένων. Τελικά, το συμπέρασμα από το δεύτερο πείραμα ήταν ότι η συνάρτηση *Κοντινότερου Συνδεδεμένου* έχει ένα σχετικό, αλλά όχι απόλυτο προβάδισμα σε σχέση με τη συνάρτηση *Hausdorff*.

ABSTRACT

Georgios Rogkakos, MSc, Computer Science Department, University of Ioannina, Greece. July, 2010. Similarity Measures For Multidimensional Data.
Thesis Supervisor: Panos Vassiliadis.

How similar are two data-cubes? In other words, the question under consideration is: given two sets of points in a multidimensional hierarchical space, what is the distance value between them? Due to the great amount of data stored nowadays, it is fundamental to provide similarity measures within sets of multidimensional data. This problem is generic since it can be found within a number of applications in fields such as multimedia information retrieval, scientific databases and digital libraries. In the context of such applications a huge amount of heterogeneous data is stored. This leads to the necessity of similarity search among this type of data. Therefore, there is a need for similarity measures that can capture human demands of search computing.

In this thesis we explore various distance functions that can be used over multidimensional hierarchical spaces. We organize the discussed functions with respect to the properties of the dimension hierarchies, levels and values. Especially, the taxonomy of distance functions we provide is as follows: Firstly, we describe distance functions that compute the distance between two values of a dimension of a multidimensional space, secondly we describe distance function that compute the distance between two points of a multidimensional space and finally we describe distance functions that compute the distance of two sets of points of a multidimensional space.

In order to discover which distance functions are more suitable and meaningful to the users, we conducted two user study analysis. The first user study analysis concerns

the most preferred distance function from the category of distance functions between two values of a dimension. The findings of this user study indicate that the most preferred distance function was the length of the path between the two values and their common ancestor in the dimension's hierarchy.

Taking into consideration the findings of the first user study we conducted a second user study. The second user study aimed in discovering which distance function, between the *closest relative* and the *Hausdorff*, from the category of distance functions between two data cubes, users prefer. The results of the second user study indicate that the *closest relative* distance function was rather preferred by users in contrast to the *Hausdorff* function.

CHAPTER 1. INTRODUCTION

How similar are two data-cubes? To put the question a little more precisely, given two sets of points in a multidimensional hierarchical space, what is the distance between these two collections? The above research problem is generic and has several applications in domains such as multimedia information retrieval, statistical data analysis, scientific databases and digital libraries [ZADB06]. In such applications, where contemporary data lead to huge repositories of heterogeneous data stored in data warehouses, there is a need of similarity search that complements the traditional exact match search. For example, one might easily envision a context where a user of an OLAP tool is proactively informed on reports that are similar to the one she is currently browsing.

In this thesis, we address the problem by (a) organizing alternative distance functions in a taxonomy of functions and (b) experimentally assessing the effectiveness of each distance function via a user study.

So far, related work has dealt with similar problems in different ways; however, this particular problem has not been dealt per se. Specifically, Sarawagi in [Sara99] and [Sara00] has dealt with the problem of discovering interesting patterns and differences within two instances of an OLAP cube. The DIFF and RELAX operators summarize the difference between two sub-cubes in order to discover the reason of abnormalities within the measures of two given cells. The only common factor of this work with ours is the usage of the Manhattan distance function in the procedure of discovering abnormalities. Our work addresses the problem of finding the appropriate distance function among a great variety of functions in order to compute the similarity between two given OLAP cubes. Giacometti et. al. [GMNS09] propose a recommendation system for OLAP queries by evaluating distances between multidimensional queries.

This work involves the distance between queries whereas our work involves distance functions between the data of multidimensional queries. Li et.al. in [LiBM03] describe the semantic similarity between ontologies. In contrast to our work, they consider a limited set of functions whereas we have a wider range of distance functions and our work focuses on distances between data in the multidimensional space.

The main findings of our approach are due to two user studies that we have conducted to assess which distance functions appear to work better for the users (Section 4). The first experiment involved 15 users of various backgrounds and the *Adult* real dataset [FuWY05]. Each user was given 14 scenarios that contained a reference cube as well as a set o variant cubes, each associated with a distance function. The task of the user was to select a cube from the set of variant cubes that seemed more similar to the reference cube. The diversity of users and data types contained in the experiment was taken into consideration in order to discover which distance function between two values of a dimension is preferred depending on the user group or the type of data. The first user study showed that all distance functions under test were used at least once, but there were a couple of distance functions that were most preferred among the others. In particular, the users seemed to prefer distance functions that express the similarity between two cubes based on the hierarchical shortest path or in regards to ancestor values.

The second user study involved 39 users and the results of the first user study were taken into account. Each user was given 14 scenarios that contained a reference cube and three variant cubes. The purpose of this second user study concerns the most preferred distance function between two data cubes.

Our approach is structured as follows: We start (Chapter 2) with the a description of the related work then (Chapter 3) we provide some formal foundations of modeling multidimensional spaces and cubes based on an existing model in the related literature [VaSk00]. and we also provide a taxonomy of distance functions for cubes based on a detailed study of the characteristics of dimension hierarchies, levels and members.

At first, we organize our families of functions as follows: Initially we describe functions that can be applied between two specific values that belong in the same level of hierarchy within a given dimension. Following, we describe distance functions that are applied between two cells of a cube and then distance functions between two OLAP cubes.

Finally, in chapter 4 the implementation issues of this thesis are presented and also the user study experiments along with the results of the most preferred functions.

CHAPTER 2. RELATED WORK

In the related literature there are a number of papers that have pointed out the necessity of having appropriate similarity measures in order to discover objects that are similar to each other and measure in a quantitative way the distance among them. Most of them examine similarity measures used between objects that are described from a number of various features such as in image retrieval or data that are stored in a hierarchical taxonomy. In addition, there are a few papers that describe how similarity measures used by human perception and computer science follow different properties. Not only computer scientists, but also scientists from other areas need similarity measures for the purpose of comparing data and objects of their expertise. In the area of Biology, a well-known example is the need of comparing genes. Another area that has dealt with the problem of introducing similarity measures is that of mathematics. Computer scientists in the areas of data mining and information retrieval have also considered the problem of introducing appropriate similarity measures. Few papers have associated the areas of mathematics and computer science and have introduced similarity measures for the concept of lattices by mapping them with semantic hierarchies.

In the following subsections we will present the related work. More precisely subsection 2.1 describes some fundamental concepts about distance functions, subsection 2.2 presents some distance functions that can be applied on graphs and

lattices, subsection 2.3 presents distances for structured data and finally subsection 2.4 describes a work about integrating texts and databases.

2.1. Fundamentals

In this subsection, we start with the presentation of some fundamental distance functions and their properties that were used in this MSc thesis. Specifically, this subsection is structured as follows: in section 2.1.1 we start the analysis of several distance measures that are categorized according to the types of variables that are applied on, in section 2.1.2 the Hausdorff distance is presented and in section 2.1.3 we discuss a work that introduces a similarity measure and demurs at the classic metric axioms.

2.1.1. Distance Measures

In this section, we follow the presentation of fundamental concepts around some common distance measures made by Han and Kamber in [JK00]. Generally, a distance measure is called a *metric* when it satisfies the following criteria:

$d(i,j) \geq 0$

$d(i,j) = d(j,i)$

$d(i,i) = 0$

$d(i,j) \leq d(i,k)+d(j,k)$

The distance measures are categorized according to the type of variables that they are applied on, in order to describe their dissimilarity. The different types of variables are the interval-scaled variables, the binary variables, the categorical variables and, finally, variables of mixed types.

As for the *interval-scaled variables* the presented distances are the *Euclidean*, the *Manhattan* and the *Minkowski* distances. For two points $p_1(x_1, x_2, \ldots, x_n)$ and $p_2(y_1,y_2,\ldots, y_n)$ in the n dimensional space, the formulas for the above distances are expressed as:

Manhattan: $dist(p_1,p_2)=|x_1 - y_1| + |x_2 - y_2| + \ldots + |x_n - y_n|$

Euclidean: $dist(p_1,p_2)=\sqrt{|x_1-y_1|^2+|x_2-y_2|^2+\ldots+|x_n-y_n|^2}$

Minkowski (*p-norm*): $dist(p_1,p_2)=\sqrt[p]{|x_1-y_1|^p+|x_2-y_2|^p+\ldots+|x_n-y_n|^p}$

Binary variables. The Jaccard distance is defined for pairs of sets comprised of members that are treated as binary variables (i.e., we can only check them for identity or not). For two objects *A* and *B* the jaccard distance is $J(A,B)=\frac{|A\cap B|}{|A\cup B|}$. Viewed from another point of view, we need to define two categories of binary variables before defining the Jaccard similarity. The first category is the symmetric binary variables and the second the asymmetric binary variables. The difference between asymmetric and symmetric binary variables is that when considering of symmetric variables, both of its states are equally valuable. For example, the agreement of two 1s (positive match) is considered the same as the agreement of two 0s (negative match). So, for the symmetric binary objects *i*,*i* we can use the equation $d(i,j)=\frac{r+s}{q+r+s+t}$ where *q* is the number of variables that equal 1 for both *i* and *j*, r is the number of variables that equal to 1 for object *i* but that are 0 for object *j*, *s* is the number of variables that equal 0 for *i* but equal 1 for *j* and *q* is the number of variables that equal 0 for both *i* and *j*. For the asymmetric binary dissimilarity between two objects *i* and *j* the previous equation becomes $d(i,j)=\frac{r+s}{q+r+s}$ because negative matches considered unimportant and so *t* is ignored. Based on the notion of similarity between *i* and *j* the equation of similarity is $sim(i,j)=\frac{q}{q+r+s}=1-d(i,j)$. Then, *sim*(*i*,*j*) is called Jaccard coefficient.

A *categorical variable* is a generalization of the binary variable because it can take more than two states. So, the dissimilarity for two categorical objects *i*,*j* is computed by the equation $d(i,j)=\frac{p-m}{p}$ where *m* is the number of matches and *p* is the total number of variables.

2.1.2. Hausdorff Distance

In [ZADB06] the authors describe the Hausdorff distance. For two sets of features $A(x_1,x_2,\ldots,x_n)$ and $B(y_1,y_2,\ldots,y_m)$ the Hausdorff distance is defined as: $d(A,B) = max\{d_s(A,B),d_s(B,A)\}$. In the above formula $d_s(A,B) = \sup_{x\in A} d_p(x_i,B)$ and $d_s(B,A)= \sup_{y\in B} d_p(A,y_j)$ where sup is the supremum of all the distances d_p. The $d_p(x_i,B)$ and $d_p(A,y_j)$ are denoted by the following formulas : $d_p(x,B)= \inf_{y\in B} d_e(x,y_j)$ and $d_p(A,y)= \inf_{x\in A} d_e(x_i,y)$ where inf is the infimum of all the distances d_e. Finally, d_e can be an arbitrary distance measure, e.g. the Euclidean distance.

For example, in the figure 2.1 there are two sets of points, the set *A* containing $\{a_1, a_2, a_3\}$ and the set *B containing* $\{b_1,b_2,b_3\}$**.** We assume, without loss of generality, that d_e denotes the Euclidean distance. In this example the notions of inf and sup coincide in being the min and max respectively. So $d_p(a_1,B)= \inf_{y\in B} d_e(a_1,y_j)=d_e(a_1, b_2)$ and similarly $d_p(a_2,B)= d_e(a_2, b_2)$, $d_p(a_3,B)=d_e(a_3, b_2)$, $d_p(A,b_1)=d_e(a_2, b_1)$, $d_p(A,b_2)=d_e(a_2, b_2)$ and $d_p(A,b_3)=d_e(a_2,b_3)$. From the above, we have that $d_s(A,B)= \sup_{x\in A} d_p(x_i,B)= d_e(a_1, b_2)$ and also $d_s(B,A)= \sup_{y\in B} d_p(A,y_j)= d_p(A,b_3)=d_e(a_2, B_3)$. Finally, $d(A,B) = max\{d_s(A,B),d_s(B,A)\}=max\{ d_e(a_1, b_2), d_e(a_2, b_3)\}$.

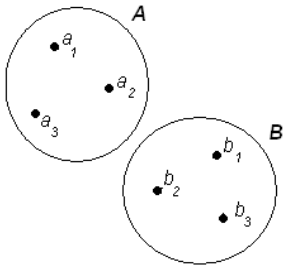


Figure 2.1 Two sets of points

2.1.3. Controversy on Metric Axioms

In [SJ95] and [SJ99] the authors introduce a similarity measure as an extension of Tversky's Feature Contrast. This extension is based on Fuzzy Logic and it is called Fuzzy Feature Contrast (FFC). Especially in the area of image and texture comparison the authors suggest that similarity measures must be close enough to human's similarity judgment introduced by psychologists. The authors were driven to use Fuzzy Logic because in a variety of works there is a disagreement on the correspondence of the metric axioms to the behavior of the real users in practice. Specifically, they provide a collection of references where the metric axioms have been refuted.

After rejecting the geometrical distance axioms such as symmetry and triangular inequality, the authors present the extension of Tversky's Feature Contrast by making use of Fuzzy Logic. The trivial procedure of measuring the similarity of two images is by expressing it as a combination (e.g., average, weighted summation) of a number of individual similarity measures between the various features that describe an image. In this paper, the authors introduce a similarity measure based on Fuzzy Logic. This way, the authors manage to express similarity between two images that are described by a number of features by taking into consideration the relationship and degree of association among the object's features. The idea of expressing a similarity measure through a Fuzzy Logic model was mainly motivated by the need of expressing a measure that can capture the human judgment. Also, the authors conducted a number of experiments trying to find similarities between images of faces and textures. Their main goal was to introduce a measure between features that captures the human perception as close as possible. Therefore, in their experiments they compared FFC and a couple of other measures (e.g., Euclidean distance) with human perception. Specifically, human subjects provided a ranking of images (faces, textures), which were compared with the equivalent rankings that occurred from the FFC and the other measures.

2.2. Distances on Graphs and Lattices

In this section we present distances that are applied on Graphs and Lattices. In section 2.2.1 the basic ideas of highway hierarchies and distances in semantic hierarchies are presented. Following, in section 2.2.2 the distances on lattices and semantic hierarchies are presented. Finally, in section 2.2.3 the similarity of words in semantic hierarchies is discussed.

2.2.1. Highway Hierarchies

In [SS05] the authors introduce a technique for the faster computation of shortest paths between two nodes of a graph. This technique borrows the idea of the highway roads in the road networks and also the Dijkstra's algorithm idea. The technique is based on the observation that the shortest paths among two points in a road network, usually consists of small roads locally and a highway road. So, the distance between two nodes in a road network is calculated by finding the shortest path of each node from a highway road and then by making use of the highway road. Based on the previous idea, a highway hierarchy is constructed. Specifically, the highway hierarchy consists of highway edges with attached sub trees of locally computable shortest paths of nodes from the highway network. An edge of the complete graph belongs in the set of highway edges if it represents an important road according to the information that it carries.

The approach of [SaSc05] was motivated by the great amount of time needed to compute distances of shortest paths in large road networks when using Dijkstra's Algorithm. The authors proposed an approach that uses the highway hierarchies in order to compute distance matrices. The basic algorithm for fast computation of distance tables is introduced based on the basic concepts and definitions of highway hierarchies. This algorithm is making use of the Highway Hierarchies query algorithm and two specific operations, namely the operations Highway Hierarchy Forward Search Space and Highway Hierarchy Backward Search Space. Highway Forward Search Space finds the nodes that belong in the shortest path originating from a source node in a graph G. Backward Search on the other hand finds the set of nodes that belong in the shortest path originating from a target node in the converse graph of G.

Finally, some optimizations on this algorithm bring further improvement on the computational time of the distance tables. In their experiments, the authors compared Dijkstra's Algorithm with the Highway Hierarchies method for the computation of distance matrices. The first experiment included 100 random nodes on the street network of Germany and the second included 173 nodes on the street network of four European countries. The experiments showed that the proposed approach for the computation of distance matrices outperforms Dijkstra's algorithm.

2.2.2. Lattices and Semantic Hierarchies

In [JO04], the author describes some fundamental ideas about treating large posets as data objects. Specifically, he refers to the notions of distance and level in such structures as an interval-valued property. A partially order set (poset) is a directed graph with no cycles and it is more general than a tree or a lattice and a node can have multiple parents. The main idea that gave feed to this work was the POSet Ontology Categorizer (POSOC), which was motivated by the needs of biologists to use algorithmic tools to navigate the Gene Ontology (GO). After reviewing POSOC's foundations, including some elementary theory about partially ordered set (poset) and in general semantic hierarchies, the author introduces two basic distance metrics in the overall structure of object under the poset notion. Namely, these metrics are (a) the interval valued poset rank and (b) the vector-valued poset distance. The first metric describes a rank as a measure of the vertical "level" of a node within a poset. The second metric describes a distance measure among nodes by taking into consideration their horizontal relationship as well. Finally, the author provides a discussion of how the two proposed metrics could work in concept lattices. This discussion is based on the trivial observation that lattices are special cases of posets.

In [JB05] paper the authors introduce link weights and weighted normalized pseudo-distances among comparable nodes in a poset. Taking into consideration some fundamental elements on DAGs, Posets and Covers, the authors continue by reintroducing the pseudo-distances implemented in Posoc. Posoc is a Categorizer for a gene ontology poset which is called a POSet Ontology (POSO) [JMFH04]. These pseudo-distances briefly are (a) the minimum chain length, (b) the maximum chain

length, (c) the average of extreme chain length and (d) the average of all chain lengths. A collection A of nodes in a poset is called chain if $\forall a,b \in A,\ a \leq b$ or $a \geq b$ In addition there is a quick review on the basic operations of probabilities on posets.

2.2.3. Semantic Similarity between Words

In [YZM03] the authors introduce a similarity measure in the field of semantic similarity between words. The propose measure combines different, already known measures such us the path length between two words in a semantic hierarchy, the depth of the subsumer concept node of these words in the hierarchy and the information content that makes use of the probability of encountering an instance of a concept in a corpus. The proposed measure and other measures were tested through an extensive experimental analysis in order to discover which measure captures better the human perception. For the needs of their experiments, the authors used two databases, the WordNet [MI95] and the Brown Corpus [7]. To evaluate their method against the state of the art methods, they applied word similarity on a word set with human ratings. The word set consisted of two subsets. The first word set included 30 pairs of words and the second included 37 pairs. All pairs were rated for similarity in meaning. The authors used the second word set in order to design their method. The first word set was used in order to test their proposed method. The authors tested 10 variations of different measures where each one occurred as a combination of the above similarity measures (i.e., the one proposed by the authors and the already known measures) and by altering the values of different parameters. The findings of [YZM03] show that the best similarity measure among the 10 measures that were tested was the similarity measure, that combined the shortest path length and the depth of the subsumer in a nonlinearly type of combination. Moreover, this new measure outperforms all previous published methods.

2.3. Distances for Collections of Structured Data

This category includes works where the distance between collections of data is measured.

In [Sar99] the author introduces a new operator for Online Analytical Processing (OLAP) products. This idea was motivated by the needs for data analysts to perform data mining tasks faster. Current OLAP products provide operators for aggregations such as Sum and Average and also provide navigational operators like Roll-up and Drill-down. The analysts use these operators for exploring the data but as the size and dimensionality increases, ad hoc exploration gets difficult and error prone. The introduced operator, called DIFF, saves time and effort for the analysts by eliminating the manual exploration for detecting reasons of fluctuations observed at an aggregated level. More precisely, the DIFF summarizes the reasons for which a cell has a bigger or a smaller aggregated quantity compared with another and completes the above operation in one step. Without the DIFF operator, the analysts should make use of a combination of several Roll-up and Drill-down operations in order to achieve the same result and with a possibility of containing errors.

The use of the DIFF operator is simple. The analyst highlights two aggregated cells on a report and then invokes the DIFF operator. The operator then will return the top rows that contain aggregated data over lower levels. These top rows are the ones that mostly affect the variance of the two cells. The number of the rows that will be returned is configurable by the user.

In general, given the two aggregated cells, the operator firstly finds the rows at the detailed level that have the biggest changes among them and secondly, it summarizes some or all of them that have similar changes. For this reason, the returned rows include also a ratio and an error field. In this part of the procedure a problem that arises concerns whether the changes of a larger magnitude are more important than the summarization of rows with similar changes.

To handle this problem the author developed an information theoretic model for cleanly capturing these tradeoffs and also suggests an algorithm that is making use of dynamic programming. The author firstly presents the way the algorithm works for a single dimension with no hierarchies. Then, this method is generalized for a single dimension with hierarchies and, finally, for multiple dimensions.

Concerning the implementation of the proposed work, the author developed the DIFF operator as a stored procedure that resides on the server's side. The stored procedure is a light-weight addition to the server because the indexing and query processing capability of the server is used to do the heavy-weight processing. Moreover, the amount of memory used by the stored procedure is independent of the number of rows.

Finally, for the experiments the author used two datasets. The first dataset was the OLAP Council Benchmark [Cou] and the other was the demo dataset Grocery Sales data, which was obtained from the Microsoft DSS product [Mic98a]. The results of the experiments showed that even for a huge number of tuples included in the DIFF query, the processing time was maximum 1 minute. Also, the scalability of the algorithm was tested over increasing number for the database tuples, the number of levels of the hierarchy and the answer size.

In [SS01] the authors propose a new operator to make the exploration of large multidimensional databases easier. This new operator called RELAX is very similar to the DIFF [Sar00] operator with the main difference that it acts the opposite way. Specifically, this new operator generalizes a drop or an increase between two cells in the detailed level. That means that the operator tries to generalize the observed drop/increase on a higher lever in some of the dimension's hierarchies. Without RELAX the analyst should use multiple Roll-ups and pivots followed by multiple drill-downs and so on. This operation might be tedious and imprecise especially for large datasets.

The use of the Relax is simple. The analyst specifies a tuple T_s and a property of T_s that he wants to generalize. An example of a property is that the sales in current year are less than sales in previous years. Then a function R measures how closely another tuple T conforms to the generalization property. Function R is called the generalization error and is zero when T is very close to T_s and increases as T departs from the generalization property. There is also a penalty function S that is close to 0

when the difference between T and T_s increases and large when T is close to T_s. A generalization is approved when the sum of $S(T)$ is greater than the sum of $R(T)$. In every generalization there might be exceptions that also appear in the results.

The authors used two datasets for their experiments, the OLAP Council Benchmark [Cou] and the Food dataset. The findings of the experiments showed that their algorithm for finding exceptions is optimal for the case of single hierarchies and finite-domained functions. Also the algorithm assigns the heavy-weight processing to the DBMS and the amount of needed memory is independent of the number of tuples.

In [MUFL06] the authors try to describe the distance between two relational databases under the same schema. One example of such databases is in the presence of replicas of a given database that might have different modifications. The motivation on the way the authors compute the distance stems from the common way that the distance between two strings is computed. More precisely, the authors define the distance of a relational database *A* from another relational database *B*, as the number of updates that must be performed to *A*, in order to become identical to *B*. By referring to updates, the authors refer to sql-like insertions, deletions and updates. Without loss of generality, they don't use insertions and deletions on their algorithms. There might be several update sequences that can bring the desired result. The sequence with the fewer updates is considered the optimal. As they present, when an update is performed it might cause more conflicts between the two relational databases than before the update but it might ease the next updates in order to achieve less number of updates.

2.4. Integrating Texts and Databases

In [XDH++08] the authors integrate traditional OLAP cubes with text data and introduce Informational Retrieval (IR) techniques on these text data. The result is what they call a *Text Cube*. The contributions of this work are (a) the introduction of a new semantic hierarchy over the terms of text collections, (b) the ability of making use of IR measures over aggregated text data and (c) the partial materialization of some previously computed cubes in order to compute more efficiently the complete aggregated cube. In the *Text Cube* two kinds of hierarchies coexist, the traditional

OLAP dimension hierarchy and the proposed *term hierarchy*. The *term hierarchy* is a semantic hierarchy that helps the navigation in the text data. Its structure is similar with the traditional OLAP hierarchies which are based on levels. In addition, the term hierarchy is related with two operations that are called *pull-up* and *push-down*. In the detailed Text Cube, for a specific assignment of the values in the cube's dimensions, a document collection is attached. In this model, if an aggregation is performed on the text data, then two IR measures, *term frequency* and *inverted index*, are materialized. Consequently, IR queries on the aggregated text data can be efficiently answered. Moreover, the authors introduce algorithms for the optimal processing of OLAP queries. Taking into consideration that the materialization of the full text cube is prohibitive, the authors materialized the cube partially. In addition, the authors propose an optimization on the partially materialized cube by bounding the query processing cost.

CHAPTER 3. FAMILIES FOR SIMILARITY MEASURES

In this section, we organize the distance functions that can be used to measure the distance between two cubes. We begin with a presentation of the OLAP model that was used in this thesis. Then we build our taxonomy of distances progressively: In section 3.2 we describe the distance functions that can be applied between two values for a given dimension. In section 3.3 we provide a taxonomy for distance functions between two cells of cubes and in 3.4 a taxonomy for distance functions between two OLAP cubes. Throughout all our deliberations we will refer to two reference dimensions, *Time* and *Location*. The hierarchies of these dimensions are shown in figure 1(a). In more detail, the *Time* dimension hierarchy consists of 5 levels. The levels of *Time* are *Day* (L_1), *Week* (L_2) and *Month* (L_2), *Year* (L_3) and *All* (L_4). The dimension *Location* consists of four levels of hierarchy which are *City* (L_1), *Country* (L_2), *Continent* (L_3) and *All* (L_4). In figure 1(b) we illustrate the lattice of the dimension *Location* at the instance level.

3.1. OLAP Fundamentals

Our model consists of data that are stored under a structured form making use of OLAP technologies. We model a collection of data in the form of a multi-dimensional

array called Cube. Each cell of the cube contains data and the cell is uniquely defined by its coordinates as values of the dimensions of the cube.

Definition 1 (level). A level $L=(\lambda_i, i\geq 1)$ is a set of finite names where λ_i is a name.

Definition 2 (dimension) [VS00]. A dimension D is a lattice $(\mathcal{L}, \prec)$ such that: $\mathcal{L}=(L_1, ..., L_n, ALL)$ is a finite subset of levels and $\prec$ is a partial order defined among the levels of $\mathcal{L}$, such that $L_1 \prec L_i \prec ALL$ for every $1<i\leq n$. We require that the upper bound of the lattice is always the level *ALL*, so that we can group all the values of the dimension into the single value 'all'. The lower bound of the lattice is called the detailed level of the dimension.

Each dimension has an associated hierarchy of levels of aggregated data. In addition, for every level L_i there is a domain of values denoted as $dom(L_i)$. Therefore, for every dimension D_i the domain is denoted as $DOM(D_i)=\bigcup_{j=1}^{m} dom(L_j)$ which states that it is the union of the domains of every level of hierarchy of the specific dimension.

Definition 3 (hierarchy). A hierarchy $\mathcal{H}=(h_1, h_2, \ldots, h_n)$ is a preordered set of levels.

Definition 4 (Cube) [VS00]. A cube c over the schema $[L_1, \ldots L_n, M_1, \ldots, M_m]$, is an expression of the form: $c=(DS^0, \varphi, [L_1, \ldots L_n, M_1, \ldots M_m], [agg_1(M_1^0, \ldots, agg_m(M_m^0)])$, where DS^0 is a detailed data set over the schema $S=[L_1^0, \ldots L_n^0, M_1^0, \ldots M_m^0]$, $m\leq k$, φ is a detailed selection condition, $M_1^0, \ldots M_m^0$ are detailed measures, $M_1, \ldots, M_m$ are aggregated measures, L_i^0 and L_i are levels such that $L_i^0 \prec L_i$, $1<i\leq n$ and agg_i, $1<i\leq m$ are aggregated functions from the set {*sum*, *min*, *max*, *count*}.

A strict hierarchy is defined as a one-to-many relationship between the values of the different levels in a dimension. In other words, assume that $L_i \prec L_{i+1}$ are two levels of hierarchy in a dimension. This hierarchy is characterized as strict when each value from L_i is related to only one value from L_{i+1} and a value from L_{i+1} may be related to many values from the level L_i. Therefore, the relationship between values of different

levels of hierarchy can be achieved through the use of a set of functions: $anc_{L_i}^{L_j}$ is a function that assigns a value from the domain of L_i to a value from the domain of L_j , where $L_i \prec L_j$.

Thus, for the set of functions $anc_{L_i}^{L_j}$ the following conditions hold:

For each pair of levels L_1 and L_2 such that $L_1 \prec L_2$ the function $anc_{L_1}^{L_2}$ maps each element of *dom(L_1)* to an element of *dom(L_2)*.

Given levels L_1, L_2 and L_3 such that $L_1 \prec L_2 \prec L_3$, the function $anc_{L_1}^{L_3}$ equals to the composition $anc_{L_1}^{L_2} \circ anc_{L_2}^{L_3}$.

For each pair of levels L_1 and L_2 such that $L_1 \prec L_2$ the function $anc_{L_1}^{L_2}$ is monotone i.e.,

$$\forall x, y \in dom(L_1) : x < y \Rightarrow anc_{L_1}^{L_2}(x) \le anc_{L_1}^{L_2}(y)$$

For each pair of levels L_1 and L_2 such that $L_1 \prec L_2$ the function $anc_{L_1}^{L_2}$ determines a set of finite equivalence classes X_i such that:

$\forall x, y \in dom(L_1), L_1 \prec L_2 : anc_{L_1}^{L_2}(x) = anc_{L_1}^{L_2}(y) \Rightarrow x, y$ belongs to the same X_i.

The relationship $desc_{L_1}^{L_2}$ is the inverse of the $anc_{L_1}^{L_2}$ function i.e.,

$$desc_{L_1}^{L_2}(1) = \{x \in dom(L) : anc_{L_1}^{L_2}(x) = 1\}$$

According to the type of values that a dimension level may have we can classify the distance functions that can be applied. Thus, we categorize the dimension levels according to the values of their domain as following.

A dimension's level domain is *Nominal* when its values hold the distinctness property. In other words, the values in such a dimension can be explicitly distinguished. For example in a dimension *Location* the level *City* can take distinct values such as *London*, *New York* etc.

A dimension's level domain is *Ordinal* when its values hold the distinctness property as well as the order property. The order property implies that the values of such a dimension abide by an order. For example in a dimension *Size* a level can take distinct and ordered values such as *small*, *medium*, *large*.

A dimension level is *Interval* when its values apart from the distinctness and order property also have the addition property. The addition property states that a unit of measurement exists. The difference between two values has a meaning, indicating how many values intermediate between them.

A dimension level is *Ratio* when its values apart from the distinctness, order and addition property also satisfy the multiplication property. The multiplication property states that differences and ratios between values have a meaning. In other words, the ratio between two values indicates their analogy difference expressed in a percentage scale.

3.2. Distance Functions between two Values

In this section we specify the distance functions that can be applied over two specific values of a dimension. In order to clarify things distance functions described in this section apply only between two dimension values and not between measure values of a cube.

Assume a specific dimension *D*, its lattice of level hierarchies $L_1 \prec L_2 \prec \ldots \prec ALL$, and two specific values *x* and *y* from levels of hierarchy L_x and L_y respectively. We classify the distance functions in the following categories: (a) *locally computable* and (b) *hierarchical computable* distance functions.

3.2.1. Locally Computable Distance Function.

The first category of locally computable distance functions can be divided into three subcategories: (a) Distance functions with explicit assignment of values, (b) Distance functions based on attribute values and (c) Distance functions based on the values of *x* and *y*.

Distance Functions with Explicit Assignment of Values. The functions of this category explicitly define n^2 distances for the n values of the *dom* (L_i) (the compared values must belong in the same level of the hierarchy). This requires *dom* (L_i) is a finite set. For example, assume a case where the distance between two cities is explicitly defined via a distance table.

Distance Functions based on Attribute Values. Assume a level whose instances are accompanied with a set of attributes. Then every level instance can be described as a tuple of attribute values. In this case, the distance between the two values x and y can possibly be expressed with respect to their attribute values via simple distance function applicable to the attributes' domains (e.g., simple subtraction for arithmetic values). For instance, assume a dimension *Products* accompanied with an attribute *Weight* which describes the weight of the products and assume a level of hierarchy of the dimension named *Drinks*. In addition, assume two specific values x = 'milk' and y = 'orange juice' where their weight attributes are $x.weight$ = 500 and $y.weight$ = 330 respectively. Then the distance between these two values can be expressed according to their weight attribute by making use, for instance, of the Minkowski distance function which is described in the following subsection. Thus, the distance between the values x and y can be defined as $|x.weight - y.weight| = 170$

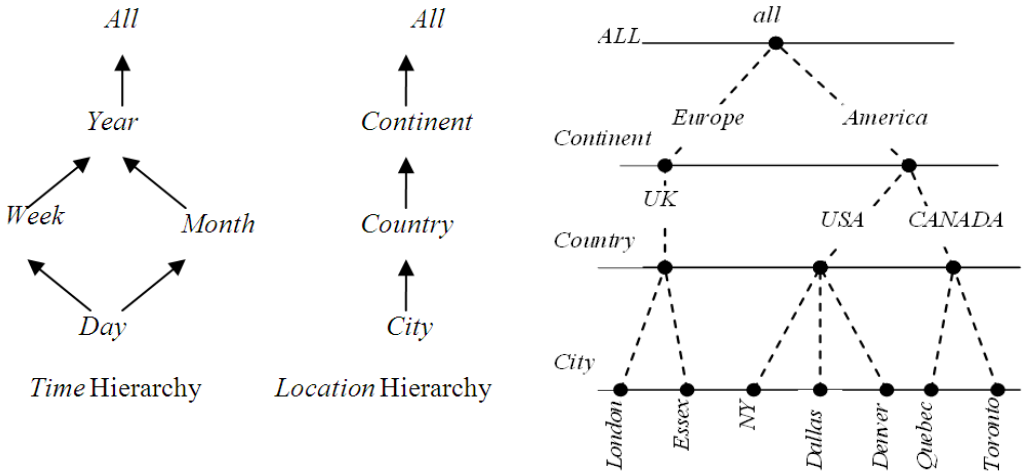
Distance Functions based on the Values x and y. In this subcategory, the distance between two values may be expressed through a function of their actual values whenever this is possible. In this subcategory one option is to make use of the simple identity function for nominal values. Thus, a value from the set {0, 1} where

$$\text{dist}(x, y) = \begin{cases} 0, \text{ if } x = y \\ 1, \text{ if } x \neq y \end{cases}$$

This function is applicable for all type values even for nominal values.

Another option is to make use of the Minkowski family distance functions especially in case where the values are of interval type. Minkowski family distance functions can be applied between two ordinal type values under the condition that the ordinal values have been mapped to the set of integer numbers. In this section, since the distance

function is applied for two specific values, all types of Minkowski distances reduce to the Manhattan distance which is $|x-y|$. As an example, consider the dimension *Time* whose levels are shown in figure 1(a). Assume two instances x and y from the level *Year*, where x= '1995' and y= '2000'. Then the distance between these two values is obviously |1995-2000| = 5. In order to normalize this distance function within the interval [0, 1], we can divide the distance value with the difference between the maximum and minimum values of the level where x and y belong in.



(a) (b)

Figure 3. 1**.** (a) The hierarchy of levels for dimensions Time and Location (b) Values of the Location dimension

3.2.2. Hierarchical Computable Distance Functions

The second category of hierarchical computable distance functions can be divided into four subcategories: (a) Distance functions with respect to an aggregation function, (b) Distance functions with respect to hierarchy path, (c) Percentage distance functions and (d) Highway distance functions.

The distance for two values that do not belong to the detailed level L_1 can be expressed with respect to an aggregation function (e.g., *count*, *max*) applied over the descendants of the two values in a lower level of hierarchy.

Distance functions with respect to an Aggregation Function. Assume an instance x from level L_i and $desc_{L_L}^{L_i}(x)$ the set of its descendants, where L_L is any lower level of L_i. The result of applying an aggregation function over the set $desc_{L_L}^{L_i}(x)$ is denoted as $x_{aggr} = f_{aggr}(desc_{L_L}^{L_i}(x))$. Assume two values x and y with $x_{aggr} = f_{aggr}(desc_{L_L}^{L_i}(x))$ and $y_{aggr} = f_{aggr}(desc_{L_L}^{L_j}(y))$, where L_L could be any lower level of L_i and L_j, $x \in L_i$, $y \in L_j$ and f_{aggr} denotes an aggregation function such as *count*, *min*, *max*, *avg* or *sum*. The distance between the values x and y can now be expressed according to the following formula: $dist(x, y) = g(x_{aggr}, y_{aggr})$, where the function g can be computed from the locally computable functions. The normalized form of this function, within the interval [0, 1], can be expressed as $dist(x, y) = \frac{g(x_{aggr}, y_{aggr})}{max\{g(a_{aggr}, b_{aggr})\}}$, where a and b are any possible values from the same level of hierarchy as x and y, i.e., $a,b \in L_i$.

Distance Functions with respect to Hierarchy Path. The distance between two values x and y can be expressed according to the length of the path in the hierarchy that connects them. Several distance functions and combinations falling into this subcategory where described by Li, Bandar and McLean in [LiBM03]. Here, we describe the distance functions that can be applied between two values x and y from a hierarchy, (a) with respect to the length of the path in the hierarchy, and, (b) with respect to the depth in the hierarchy path. Assume two values x and y such that $x \in L_x$ and $y \in L_y$. We denote the *Lowest Common Ancestor* of x and y as $lca(x,y)$.

The lowest common ancestor *lca*, of two values x and y where $x \in L_x$ and $y \in L_y$, $lca \in L_z$ and L_z is any non lower level of L_x and L_y, $L_z \succ L_x$, L_y is a value such that:

$$lca=\{z|z = anc_{L_x}^{L_z}(x) \wedge z = anc_{L_y}^{L_z}(y) \wedge (\nexists\ z' \mid z'= anc_{L_x}^{L_z}(x) \wedge z'= anc_{L_y}^{L_z}(y) \wedge L_{z'} \prec L_z\} \quad (1)$$

The distance between the values x and y can be expressed with one of the following formulas:

1. $$\text{dist}(x, y) = f_{path}\left(\frac{w_x * | path(x, lca) | + w_y * | path(y, lca) |}{(w_x + w_y) * | path(ALL, L_1) |}\right)$$

2. $\text{dist}(x, y) = f_{\text{depth}}\left(\frac{|\, path\,(\, lca, L_1)\,|}{|\, path(ALL, L_1)\,|}\right)$

The first formula indicates that the distance is a function of the weighted sum of the length of the path from the values *x* and *y* to their lowest common ancestor *lca*. The second formula indicates that the distance of the values is expressed as a function of the length of the path of the lowest common ancestor *lca* from the detailed level L_1 of the hierarchy. In both formulas the functions f_{path} and f_{depth} may be any linear or exponential function such as $f(x) = e^{c*x}$, where *c* is any real parameter. These two functions are normalized in the interval [0, 1] by making use of the height of the hierarchy. Specifically, the first formula is divided by $(w_x + w_y)*|\, path(ALL, L_1)\,|$ whereas the second formula is divided by $|\, path(ALL, L_1)\,|$. As an example, assume two values *x*='NY' and *y*='Canada' from the hierarchy *Location* denoted in figure 1(b) where their lowest common ancestor is the value *lca* = 'America' from the level *Continent*. For simplicity, assume the functions f_{path} and f_{depth} are equal to the identity function and the weighted factors w_x and w_y are set to 1. Therefore, the functions become: f_{path}= (|*path* (*x*, *lca*)| + |*path* (*y*, *lca*)|)/ 2*|*path*(*ALL*, L_1)| and f_{depth}= |*path* (*lca*, L_1)|/ |*path*(*ALL*, L_1)|. The distance between *x* and *y* occurs to be f_{path}= (2+1)/2*3 =0.5 and f_{depth}=2/3.

Percentage Distance Functions. According to this subcategory, the distance between two values *x* and *y*, where *y* is an ancestor of *x*, may be expressed according to a percentage of occurrences over the values of the hierarchy. In other words, the similarity of two values is expressed as the similarity of the number of descendants this two values have. Assume the lattice of level hierarchies be denoted as $L_1 \prec \ldots \prec L_L \prec L_x \prec L_y \prec All$ where L_1 denotes the most detailed level. The distance of a value *x* in a level L_x in regards to its ancestor *y* in level L_y may be calculated according to the function:

$$dist(x, y) = \frac{|\, desc_{L_i}^{L_x}(x)\,|}{|\, desc_{L_i}^{L_y}(y)\,|}\text{ , where } L_i \text{ is one of the levels } L_x, L_L \text{ and } L_1 \quad (3)$$

The above formula expresses the distance between a value *x* and one of its ancestors *y* as a percentage via three ways. In case L_i is L_x, then the distance is expressed as a percentage in regards to the occurrences of all the other values from L_x whose

ancestor is *y*. In case L_i is L_L(or L_1), the distance is expressed as a percentage of occurrences of the descendants of *x* in a lower level of hierarchy L_L(or L_1) in regards to the descendants of *y* in the same lower level L_L(or L_1). As an example, assume the dimension *Location* where its lattice can be visualized in figure 1(a) and the values of this dimension are visualized in figure 1(b). Assume the values *x*='USA' and *y*='America'. Then, in regards to the above formula the distance between these two values can be computed as:

i. $dist('USA','America') = \frac{1}{|\, desc_{Country}^{Continent}('America')\,|} = \frac{1}{2}$ where L_i is chosen to be the level L_x, i.e., $L_{country}$

ii. $dist('USA','America') = \frac{|\, desc_{City}^{Country}('USA')\,|}{|\, desc_{City}^{Continent}('America')\,|} = \frac{3}{5}$ where L_i is chosen to be the detailed level L_1, i.e., L_{city}

As for the third case, in this example it coincides with the second since the lower and detailed level, i.e. *City*, are identical.

Highway Distance Functions. Assume that every level of hierarchy *L* is grouped into *k* groups and every group has its own representative r_k. Then, the distance between two representatives can be thought of as a highway [SaSc05]. We denote with *r(x)* and *r*(*y*) the representatives of the groups where *x* and *y* belong in respectively. There fore, the distance between the values *x* and *y* can be expressed with the following formula:

$$dist\,(x, y) = dist\,(x, r(x)) + dist\,(r(x), r(y)) + dist\,(y, r(y)) \qquad (2)$$

The partial distances between a value and its representative and the distance between the two representatives *r(x)* and *r*(*y*) depend on the way the representative is selected. In most cases the representatives are selected so that they belong in the same level of hierarchy and thus their distance can be computed from the locally computable functions, the path functions or the aggregated functions (in case the two representatives belong in different levels their distance may be computed by applying any distance function from the path section or the aggregated distance function section). The main categories of selecting the representative apart from an explicit

assignment are in regards to (a) an ancestor and (b) a descendant. For the following, *dist*(*a*, *b*) denotes the distance of any two values *a*, *b*. Without loss of generality assume $L_x \prec L_y$. In addition, assume the ancestor of *x* in level L_y denoted as $x_y = anc_{L_x}^{L_y}(x)$ and a representative of *y* in the level of hierarchy L_x denoted as $y_x = f(desc_{L_x}^{L_y}(y))$. These can be visualized through figure 2. The function *f* applied over the descendants of *y* can result either to an explicitly assigned descendant or to the result of an aggregation function (e.g., *min*, *max*) over the set of descendants. In the following we describe the partial distances of formula 2 depending on the way the representative is selected.

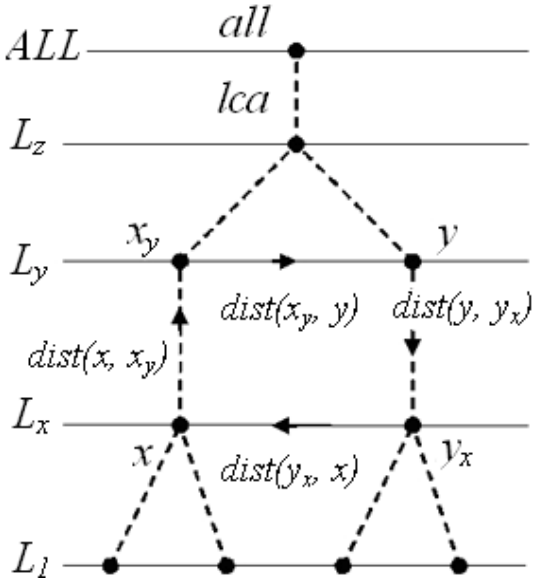


Figure 3.2 Partial distances between two values in different levels of hierarchy.

a) <u>The representative of a group is an ancestor</u>. The representative of each value *x* and *y* could be $r(x) = anc_{L_x}^{L_U}(x)$ and $r(y) = anc_{L_y}^{L_V}(y)$ where L_U and L_V is any upper level of L_x and L_y respectively. L_U and L_V are not obligatory different. In general, the distance between a value *x* and its representative may be computed through any distance function from the path, the percentage or the aggregated functions. For example, assume two values *x*='UK' and *y*='USA' from the level *Country* of the hierarchy *Location* denoted in figure 3.1(b). Assume the representative *r*(*x*)='Europe'

and the representative $r(y)$='America'. The distance of the values x and y is by summing the distances *dist*('UK', 'Europe'), *dist*('Europe', 'America') and *dist*('America', 'USA'). In this category there are two special cases:

1. The representatives $r(x)$ and $r(y)$ coincide in being the lowest common ancestor *lca*, where the formula is simplified as: $dist(x, y) = dist(x, lca) + dist(y, lca)$.
2. The representative $r(y)$ is identical to the actual value of y. In this case the distance is expressed as a summation of $dist(x, x_y)$ and $dist(x_y, y)$, as shown in figure 2, where x_y is the representative of x from the level L_y. Therefore, the distance $dist(y, r(y)) = 0$. Formally this is expressed as:

$$dist(x, y) = dist(x, x_y) + dist(x_y, y) = dist(x, anc_{L_x}^{L_y}(x)) + dist(anc_{L_x}^{L_y}(x), y).$$

In case the representative x_y of x and y coincide, the distance is simplified as $dist(x, y) = dist(x, x_y)$. Since $dist(x, x_y)$ and $dist(x_y, y)$ are within the interval [0, 1], the normalized form of $dist(x, y)$ occurs by dividing it with 2. For example, assume two values x = 'USA' and y = 'Europe' from the dimension *Location* as seen in figure 1. The ancestor x_y of x is $anc_{Country}^{Continent}(x) = \text{'America'}$. Assume $dist(x, x_y)$ is computed from the percentage family functions. $dist(x_y, y)$ is computed through the first formula from the path family functions where the weighted factors w_x and w_y are set to 1. The distance between x and y becomes $dist(\text{'}USA\text{'}, \text{'}Europe\text{'}) = (dist(x, x_y) + dist(x_y, y))/2 = (dist(\text{'}USA\text{'}, \text{'}America\text{'}) + dist(\text{'}America\text{'}, \text{'}Europe\text{'}))/2 = (1/2 + 2/3)/2 = 7/12$.

b) <u>The representative of a group is a descendant</u>. The representative of a group can be selected with respect to the descendants of the group where x belongs. For example, consider countries whose representatives can be selected among their cities, based for instance on the major airport or the highest population. In case the representative $r(x)$ is a value from the domain of L_L (i.e., $r(x)$ picked explicitly from the set $desc_{L_x}^{L_L}(x)$ or by applying a *min* or *max* aggregation over the set $desc_{L_x}^{L_L}(x)$), the distance between x and $r(x)$ can be any function from the families of path, percentage or aggregated functions. In case $r(x)$ is an arithmetic type value (i.e., a *sum* or *count* aggregation function applied over the set $desc_{L_x}^{L_L}(x)$), the distance between x and $r(x)$ can be any simple arithmetic function such as the Minkowski. There is a special case where the representative $r(x)$ is identical to the actual value of x. Thus, the distance is expressed

as a summation of $dist(y, y_x)$ and $dist(y_x, x)$, where y_x is the representative of y from the level L_x as shown in figure 2. Therefore, the distance $dist(x, r(x))=0$. Formally this is expressed as:

$$dist(x, y) = \frac{dist(y, y_x) + dist(y_x, x)}{2} = \frac{dist(y, f(desc_{L_x}^{L_y}(y))) + dist(f(desc_{L_x}^{L_y}(y)), x)}{2}$$

where the denominator is set to 2 for normalization reasons. For example, assume two values from the hierarchy *Location*, x='USA' and y='Europe', where the descendant of y is selected as $f(desc_{L_x}^{L_y}(y)) = \text{'UK'}$. Assume the distance between y and its descendant y_x is computed through the formula $dist(y_x, y) = \frac{|desc_{L_x}^{L_x}(y_x)|}{|desc_{L_x}^{L_y}(y)|}$ from the percentage family functions. The distance between x and y_x is computed through the first formula from the path family functions with w_x and w_y set to 1. Consequently, the distance between x and y becomes $dist(\text{'USA'}, \text{'Europe'}) = \frac{dist(y, y_x) + dist(y_x, x)}{2} = \frac{dist(\text{'Europe'}, \text{'UK'}) + dist(\text{'UK'}, \text{'USA'})}{2} = \frac{1/1 + 4/6}{2} = \frac{5}{6}$.

In the special case where x is a descendant of y the above formula is simplified as: $\text{dist}(x, y) = \text{dist}(y, y_x)$.

3.3. Distance Functions between two Cells of Cubes

In this section we describe the distance functions that can possibly be applied in order to measure the distance between two cells from a cube. Assume an OLAP cube C defined over the detailed schema $C = [L_1^0, L_2^0, \ldots, L_n^0, M_1^0, M_2^0, \ldots, M_m^0]$, where L_i^0 is a detailed level and M_i^0 is a detailed measure. In addition assume two cells from this cube, $c_1 = (l_1^1, l_2^1, \ldots, l_n^1, m_1^1, m_2^1, \ldots, m_m^1)$ and $c_2 = (l_1^2, l_2^2, \ldots, l_n^2, m_1^2, m_2^2, \ldots, m_m^2)$, where $l_i^1, l_i^2 \in dom(L_i^0)$ and m_i^1, m_i^2 denote the values of the corresponding measure M_i^0. The distance between two cells c_1 and c_2 can be expressed in regards to a) their level coordinates $d_l(L_i^1, L_i^2)$ and b) their measure values $d_l(M_i^1, M_i^2)$. In other words, $dist(c_1, c_2) = f(d_l(L_i^1, L_i^2), d_l(M_i^1, M_i^2))$. The function f can possibly be (a) a weighted sum, (b) Minkowski distance, (c) min or (d) proportion of common coordinates.

3.3.1. Distance functions between two Cells of a Cube Expressed as a Weighted Sum.

In this category the distance between two cells c_1, c_2 where c_1, $c_2 \in C$ can be expressed through the formula $f : \frac{\sum_{i=1}^{n} w_i d_i(l_i^1, l_i^2)}{\sum_{i=1}^{n} w_i} + \frac{\sum_{i=1}^{m} w'_i d_i(m_i^1, m_i^2)}{\sum_{i=1}^{m} w'_i}$, where w_i and w'_i are parameters that assign a weight for the level L_i and the measure M_i respectively, $d_i(l_i^1, l_i^2)$ denotes the partial distance between two values of the detailed level L_i^0 from dimension D_i and $d_i(m_i^1, m_i^2)$ denotes the partial distance between two instances of the measure M_i^0. Regarding the distance $d_i(l_i^1, l_i^2)$, this is expressed through the various formulas from the section 3.1 which describes the possible distance functions between two values from the same level of hierarchy over a dimension. The distance $d_i(m_i^1, m_i^2)$ between two instances of a measure can be calculated through the Minkowski family distance when m_i^1, m_i^2 are of arithmetic type, or through the simple identity function in case m_i^1, m_i^2 are of character type. The above formula is a general expression of the distance between two cells. Simplifications of this can be applied. For instance, the distance of two cells can be calculated only with respect to the coordinates that define each cell and without taking into consideration the measure values of each cell, i.e., by omitting from the above formula the second fraction. Moreover, in case the partial distances are normalized in the interval [0, 1] then, f expresses the overall distance between two cells normalized in the same interval [0, 1]. For example, assume we want to compute the distance between cells c_1, c_2 as shown in figure 3.3. Both cells consist of two dimensions (*Time, Location*), where their hierarchy levels can be seen in figure 3.1, and contain one measure (*Sales*). In the above formula we set the weight factors of the dimensions (w) and the weight factors of the measures (w') equal to 0.5. The distance between dimensions is computed according to the function f_{path} that takes into account the length of the path of the hierarchy. The distance between the measures is computed through the normalized Manhattan distance function. In addition, assume that the overall maximum and minimum values of the measure sales are 10 and 1 respectively. With the above settings we obtain: $d(c_1,c_2)=$

$$\frac{w * d(Month_{c_1}, Month_{c_2}) + w * d(Country_{c_1}, Country_{c_2})}{w + w} + \frac{w' * d(Sales_{c_1}, Sales_{c_2})}{w'} =$$

$$\frac{0.5*1/3+0.5*1/3}{0.5+0.5}+\frac{0.5*(|4-3|/|10-1|)}{0.5}=4/9$$

	Month	Country	Sales
C_1	May/2000	USA	4

	Month	Country	Sales
C_2	Apr/2000	canada	3

Figure 3.3 Instances of cells c_1 and c_2

To compute the distances $d(Month_{c_1}, Month_{c_2})$ and $d(Country_{c_1}, Country_{c_2})$ we refer the reader to the figures 3.4 and 3.5. In figure 3.4 we see that the length of the path between the nodes *a* and *lca* is 1, and the length of the path between the nodes *b* and *lca* is 1 again. According to the function f_{path}, $d(Month_{c_1}, Month_{c_2})=\frac{1+1}{6}=\frac{1}{3}$. In a similar manner, by using the information that derives from the figure 3.5 $d(Country_{c_1}, Country_{c_2})=\frac{1+1}{6}=\frac{1}{3}$.

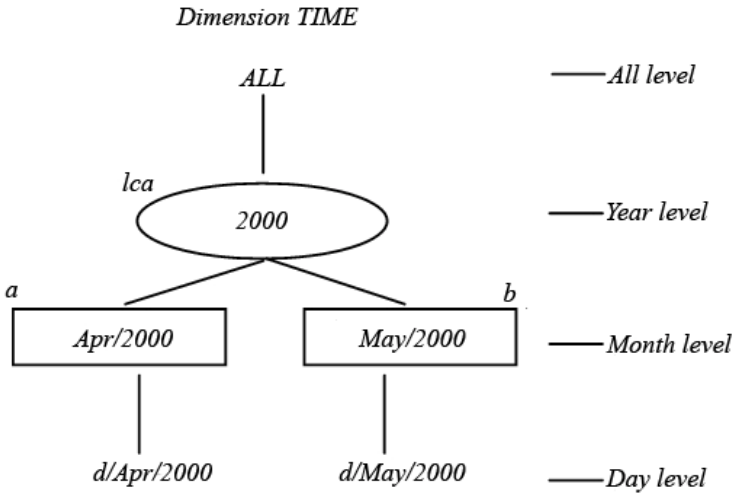


Figure 3.4 Lattice of the dimension *TIME* for the values of cells of figure 3.3

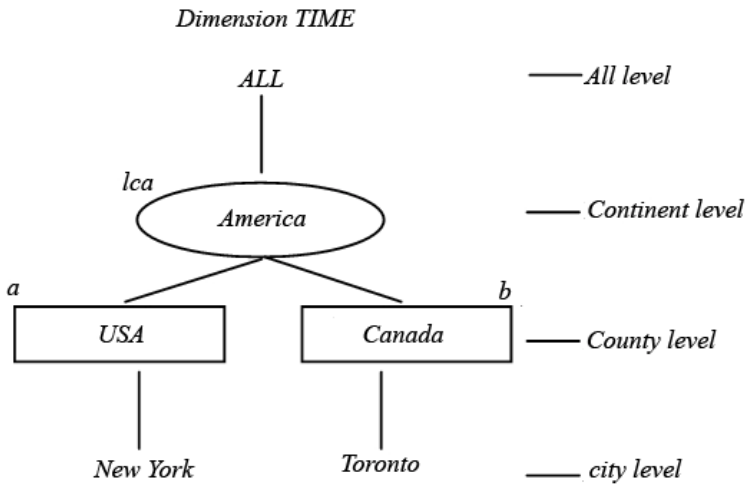


Figure 3.5 Lattice of the dimension *LOCATION* for the values of cells of figure 3.3

3.3.2. Distance functions between two Cells of a Cube Expressed in regards to the Minkowski Family Distances.

In this section we describe the possible distance functions between two cells from a cube by making use of the Minkowski family distances. In general the Minkowski distance is defined via the formula $L_p[(x_1,...,x_n),(y_1,...,y_n)] = \sqrt[p]{\sum_{i=1}^{n} d_i(x_i, y_i)^p}$, where $d_i(x_i, y_i)$ denotes the distance between the two coordinates x_i and y_i of two given points x and y. Assume two cells $c_1 = (l_1^1, l_2^1, \ldots, l_n^1, m_1^1, m_2^1, \ldots, m_m^1)$ and $c_2 = (l_1^2, l_2^2, \ldots, l_n^2, m_1^2, m_2^2, \ldots, m_m^2)$, where $l_i^1, l_i^2 \in dom(L_i)$ and m_i^1, m_i^2 denote the values of the corresponding measure M_i. The Minkowski distance can be applied in this category, by substituting point coordinates x_i and y_i with cell coordinates, thus l_i^1 and l_i^2. In general, in the Minkowski family distances the partial distances are defined as $d_i(x_i, y_i)=|x_i - y_i|$. When applying the Minkowski distance over cell coordinates, then the partial distances $d_i(l_i^1, l_i^2)$ can be expressed as the distance between two values from the same level of hierarchy as described in section 3.1.

So far, the distance between two cells is described only in regards to their level coordinates. However, the distance between two cells can also be expressed by taking into consideration the instance values of the cells, thus their measure values. The Minkowski family distances can be applied, as well, in regards to the partial distances

$d_i(m_i^1, m_i^2)$. Therefore, the distance between two cells can be expressed by adding the equivalent two formulas. Depending on the value of *p* the Minkowski distances over two cells are defined as:

$$L_1 = \sum_{i=1}^{n} d_i(l_i^1, l_i^2) + \sum_{i=1}^{m} d_i(m_i^1, m_i^2)\text{ , 1-norm distance}$$

$$L_2 = \sqrt{\sum_{i=1}^{n} (d_i(l_i^1, l_i^2))^2} + \sqrt{\sum_{i=1}^{m} (d_i(m_i^1, m_i^2))^2}\text{ , 2-norm distance}$$

$$L_p = \sqrt[p]{\sum_{i=1}^{n} (d_i(l_i^1, l_i^2))^p} + \sqrt[p]{\sum_{i=1}^{m} (d_i(m_i^1, m_i^2))^p}\text{ , p-norm distance}$$

$$L_\infty = \lim_{p\to\infty}\left(\sqrt[p]{\sum_{i=1}^{n} (d_i(l_i^1, l_i^2))^p}\right) + \lim_{p\to\infty}\left(\sqrt[p]{\sum_{i=1}^{m} (d_i(m_i^1, m_i^2))^p}\right) =$$

$$\max\left(d_1(l_1^1, l_1^2), d_2(l_2^1, l_2^2), ..., d_n(l_n^1, l_n^2)\right) +$$

$$\max\left(d_1(m_1^1, m_1^2), d_2(m_2^1, m_2^2), ..., d_m(m_m^1, m_m^2)\right)$$

infinity norm distance or Chebyshev distance.

3.3.3. Distance Functions between two Cells of a Cube Expressed as the Minimum Partial Distance.

In this category the distance between two cells $c_1 = (l_1^1, l_2^1, \ldots, l_n^1, m_1^1, m_2^1, \ldots, m_m^1)$ and $c_2 = (l_1^2, l_2^2, \ldots, l_n^2, m_1^2, m_2^2, \ldots, m_m^2)$ can be expressed as:

$$\min_{d_i}\{d_i(l_i^1, l_i^2)\} + \min_{d_i}\{d_i(m_i^1, m_i^2)\} = \min\left\{d_1(l_1^1, l_1^2), d_2(l_2^1, l_2^2), ..., d_n(l_n^1, l_n^2)\right\}$$
$$+ \min\left\{d_1(m_1^1, m_1^2), d_2(m_2^1, m_2^2), ..., d_m(m_m^1, m_m^2)\right\}.$$

Therefore, the distance between two points is expressed as the minimum distance of their level coordinates plus the minimum distance of their measure values.

3.3.4. Distance Functions between two Cells of a Cube Expressed as a Proportion of Common Coordinates.

In this category the distance between two cells can be expressed as a proportion of their common values of their level coordinates and their measure values. Therefore, the distance between two cells $c_1 = (l_1^1, l_2^1, \ldots, l_n^1, m_1^1, m_2^1, \ldots, m_m^1)$ and $c_2 = (l_1^2, l_2^2, \ldots, l_n^2, m_1^2, m_2^2, \ldots, m_m^2)$ can be expressed through the formula f: $\frac{count(l_i^1 = l_i^2 \forall i \in \{1,2,...,n\})}{n} + \frac{count(m_i^1 = m_i^2 \forall i \in \{1,2,...,m\})}{m}$. The above formula states the distance between two cells as a summation of two fractions. The first fraction is the number of level values that are same for both cells, divided by the number of all level values that describe a cell. The second fraction expresses the number of measures that have the same value for both cells divided by the number of all possible measures in a cell.

3.4. Distance Functions between two OLAP Cubes

Assume two OLAP cubes C and C' defined through the same detailed schema $[L_1^0, L_2^0, \ldots, L_n^0, M_1^0, M_2^0, \ldots, M_m^0]$, where L_i^0 is a detailed level and M_i^0 is a detailed measure. In addition assume that cube C consists of l cells of the form $c = (l_1, l_2, \ldots, l_n, m_1, m_2, \ldots, m_m)$ and cube C' consists of k cells of the form $c' = (l_1', l_2', \ldots, l_n', m_1', m_2', \ldots, m_m')$, where $l_i, l_i' \in dom(L_i^0)$ and m_i, m_i' denote the values of the corresponding measure M_i^0 . In general the two cubes can be of different cardinality, i.e., $l \neq k$. Assume $dist(c, c')$ where $c \in C$ and $c' \in C'$ denotes the distance between two specific cells according to the various categories of section 3.3. The distance between the two cubes can be expressed as a synthesis of the partial distances $dist(c, c')$. In other words $dist(C, C') = f\ (dist(c, c'))$ is a function of the partial distances $dist(c, c')$. The function f can possibly belong to one of the following families: (a) closest relative, (b) Hausdorff distance, (c) a weighted sum, (d) Minkowski distance, and (e) Jaccard's coefficient. Specifically, distance functions that fall within the families (c) and (d) include the *Cell Mapping* method which is described in the next subsection. The rest distance function families (i.e., (a), (b), (e)) do not include the cell mapping method.

For example, assume we want to compute the distance between the two cubes $CUBE_1$ and $CUBE_2$ as shown in figure 3.6 $CUBE_1$ consists of three cells whereas $CUBE_2$ consists of 5 cells. Each cell in both cubes consists of two dimensions in different levels of hierarchy and the measure *Sales*. Specifically, each cell of $CUBE_1$ is of the form c = (*Day*, *City*, *Sales*) and each cell of $CUBE_2$ is of the form c' = (*Year*, *Country*, *Sales*). The distance between the two cubes can be expressed by applying a function f over the partial distances $dist(c,c')$ of the cells of the two cubes.

$CUBE_1$

	Day	City	Sales
c_1	3/5/2000	London	5
c_2	3/5/2001	New York	6
c_3	4/5/2001	New York	7

$CUBE_2$

	Year	Country	Sales
c_4	2000	USA	3
c_5	2000	USA	6
c_6	2001	Canada	8
c_7	2001	UK	5
c_8	2000	USA	9

Figure 3.6 Instances of two cubes

3.4.1. Cell Mapping and Categories of Distance Functions according to it

In this section we introduce the method that is used in order to map the cells of one cube to the cells of another cube. We refer to this method as *Cell Mapping*. For two cubes C_1 and C_2, the simple mapping of their cells includes the connection of every cell of the cube C_1 with one cell of the cube C_2. Intuitively, the mapping of a cell in cube C_1 tries to capture the discovery of the "closest possible representative" of this cell in cube C_2. The "closest representative" is the cell of the cube C_2 with the less distance among the dimension values with the cell of the cube C_1. In principle, the Cell Mapping method can be thought of as a relation that connects the cells of a cube to the cells of another cube (i.e., one can consider several candidate "representatives" of a cell). However, in our setting, this relation is reduced to a function, since we are interested in mapping each cell from the first cube to only one cell from the second cube. This is done for reasons of simplicity and allows the elegant definition of cube distances (see next). We impose the restriction that the function is total, i.e., each and every cell from the first cube is mapped to a cell of the second cube. We do not

require that the mapping is 1:1 and onto; thus, in the second cube there might be a cell in which more than one, or, no cells at all, from the first cube are mapped to it.

As an example assume the cubes that are presented in the figure 3.7. In figure 3.7 (a) the cells *A*, *B*, *C* of $CUBE_1$ are mapped to the cells *E*, *D*, *H* of $CUBE_2$ respectively. Moreover, in the same figure the cells *F*, *G* of $CUBE_2$ are not mapped with any cell of $CUBE_1$. In figure 3.7 (b) we can observe that the cell *E* of $CUBE_2$ is mapped with two cells of *CUBE1*.

The cell mapping method needs to compute the distances between the dimensions of each cell of the first cube with the dimensions every cell of the second cube and ignoring the distance between the measures. So, if the distance between two cells c_1, c_2 is expressed as $f\ (d_i(L_i^1, L_i^2), d_i(M_i^1, M_i^2))$ then the mapping method considers only the $d_i(L_i^1, L_i^2)$. Thus, each cell of the first cube is mapped to the cell of the second cube with the less $d_i(L_i^1, L_i^2)$ distance.

In our taxonomy, two distance functions between cubes make use of the cell mapping method. These are (a) distance functions expressed in regards to the *Closest Relative* and (b) the distance function expressed by Hausdorff distance. After the mapping has been accomplished, the distances between the mapped cells are computed. Finally, the computation of the distance between the two cubes involves the distances among the mapped cells.

The distance functions that can be used in order to compute the distance between two OLAP cubes can be divided into two categories. The first category involves distance functions that include the cell mapping method. The second category contains distance functions that do not include the cell mapping method. Following, we describe each distance function and provide its analytical formula. The distance functions of the first category are the *Closest Relative* and the *Hausdorff Distance* (section 3.4.2) that include the cell mapping method. Then, the category of families that do not consider the cell mapping method in their definition, include the *Weighted*

Sum function, the *minkowski family* of distance functions, the *Jaccard's Coefficient* and the *minimum of distances* function.

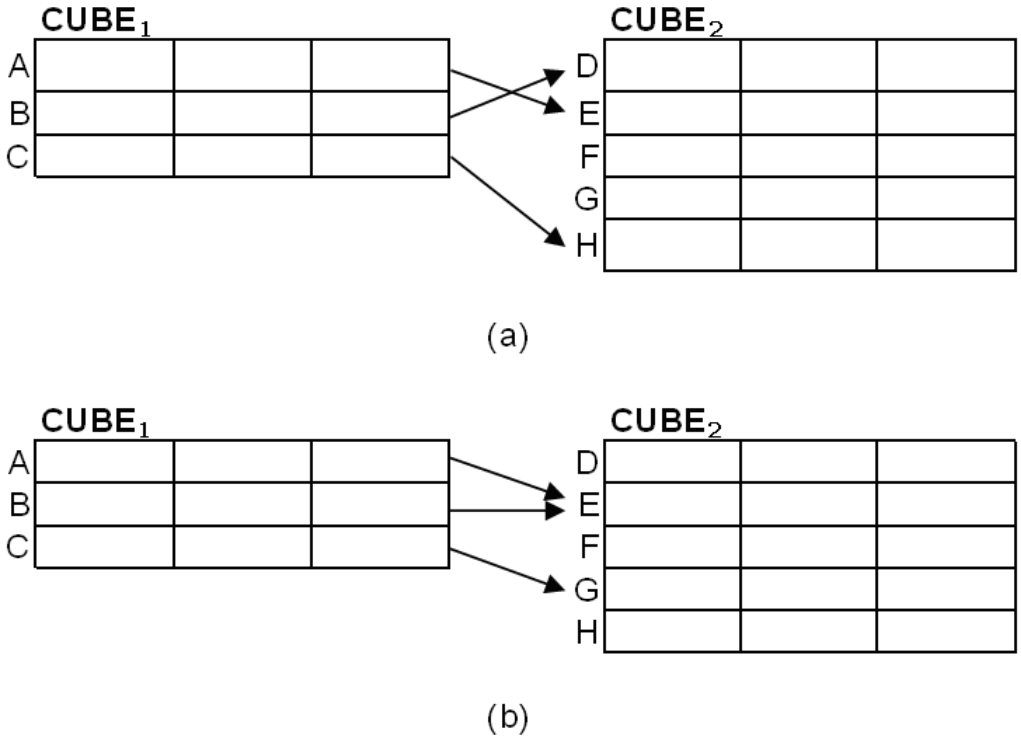


Figure 3.7 (a) cells of cube $CUBE_1$ mapped to the cells of cube $CUBE_2$ (b) cells of cube $CUBE_1$ mapped to the cells of cube $CUBE_2$

3.4.2. Distance Functions that Include Mappings

This subsection contains the description of the distance functions that involve the Cell Mapping method. These distance functions are the *Closest Relative* and the *Hausdorff* and are described as follows.

Distance function between two cubes expressed in regards to the closest relative. In this category the distance between two cubes *C* and *C*' is expressed as the summation of distances between every cell of a cube with the most similar cell of another cube through the formula:

$$dist(C,C') = \frac{\sum_{i=1}^{k}(dist(c_i,c'))}{k} \forall c' \mid dist_{\dim}(c_i,c') = \min\{dist_{\dim}(c_i,c')\} \quad \text{where} \quad dist_{\dim}$$

denotes the distance of two cells excluding the distance of their measures. The

$\forall c' \mid dist_{\dim}(c_i, c') = \min\{dist_{\dim}(c_i, c')\}$ part of the above formula reveals the cell mapping method. Each one of the *k* cells from cube *C* is mapped to the cell of the cube C' that has the minimum $dist_{\dim}$ from it.

As an example, we will analyze the computation of the distance between the cubes $CUBE_1$ and $CUBE_2$ shown in figure 3.8. The first step is to map the cells of the cube $CUBE_1$ to the appropriate cells of the cube $CUBE_2$. In order to simplify the example the computational part of the cell mapping method is not described here, but the cell mapping is denoted in figure 3.8 through arrows between the cells of the two cubes. The distance function used in this example for the purpose of computing the distance between the cells of the two cubes is the weighted sum. The weight that was used is 0.5, equal for both the dimensions and measures. In addition, the distance function used to measure the distance between the dimensions is the f_{path} function. The cells c_1, c_2, c_3, are mapped to the cells c_7, c_5, and c_5 respectively. According to this mapping, in order to compute the distance between the two cubes, the needed distances between cells are:

$$d(c_1, c_7) = \frac{0.5*1/6 + 0.5*1/6}{0.5+0.5} + \frac{0.5*(|5-5|/|10-1|)}{0.5} = 1/6+0=1/6$$

$$d(c_2, c_5) = \frac{0.5*1/6 + 0.5*1/6}{0.5+0.5} + \frac{0.5*(|6-6|/|10-1|)}{0.5} = 1/6+0=1/6$$

$$d(c_3, c_5) = \frac{0.5*1/6 + 0.5*1/6}{0.5+0.5} + \frac{0.5*(|6-7|/|10-1|)}{0.5} = 1/6+1/9=5/18$$

For the above computations we refer the reader to the figures 3.4 and 3.5 where the hierarchies of the dimensions *LOCATION* and *TIME* are presented. With the above distances, we can now compute the full distance between the cubes $CUBE_1$ and $CUBE_2$ through the first formula of the *closest relative* family functions:

$$d(CUBE_1, CUBE_2) = \frac{d(c_1, c_7) + d(c_2, c_5) + d(c_3, c_5)}{3} = \frac{1/6 + 1/6 + 5/18}{3} = 0.319444$$

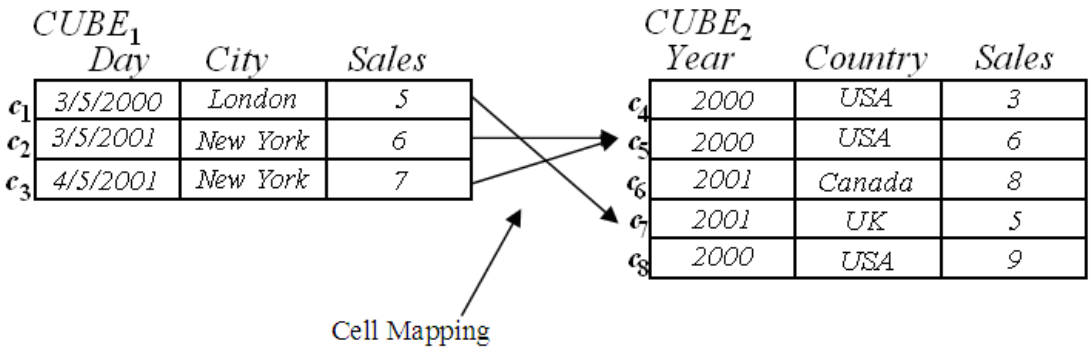


Figure 3.8 Instances of two cubes and the mapping of their cells

Distance functions between two cubes expressed by Hausdorff distance. In this category the distance between two cubes can be expressed by making use of the *Hausdorff* distance [HuKR93]. The Hausdorff distance between two cubes can be defined as $H(C, C') = max(h(C,C'), h(C',C))$ where $h(C,C') = \max_{c \in C}\{\min_{c' \in C'}\{dist(c,c')\}\}$ and *dist* (*c, c'*) is the distance between two cells *c* and *c*' from the cubes *C* and *C*' respectively. The function *h*(*C, C*') is called the *directed* Hausdorff distance from *C* to *C*' and the distance measured is the maximum distance of a cube *C* to the "*nearest*" cell of the other cube *C*'. The Hausdorff distance is the maximum of *h*(*C, C*') and *h*(*C*', *C*).

In the Hausdorff distance function the cell mapping method is bidirectional. That means that except from the mapping that we have examined in the closest relative function we need an extra mapping and that is the mapping from the cells of cube *C*' to the cells of Cube *C*.

When the bidirectional mapping is completed, we obtain two sets of mapped cells. In each set, for every pair of mapped cells, we compute their distance considering now their measures as well. Thus, essentially, we have two sets of minimum distances between cells, the set of minimum distances from the cells of cube *C* to the cells of cube *C*' and the set of minimum distances between from the cells of cube *C*' to the cells of cube *C*. From each of the two sets we pick the greatest distance and finally from these two distances we pick the greater one.

To make things more clear an example follows. Assume again the cubes $CUBE_1$ and $CUBE_2$ as shown in figure 3.9. The figure 3.9 also presents the mapping from the cells of $CUBE_1$ to the cells of $CUBE_2$. In figure 3.9 we can observe the same cubes and the mapping from the cells of $CUBE_2$ to the cells of $CUBE_1$. According to this bidirectional mapping the two resulting sets of minimum distances are:

$S_1\{d(c_1,c_7), d(c_2,c_5), d(c_3,c_5)\}$

$S_2\{d(c_4,c_3), d(c_5,c_3), d(c_6,c_2), d(c_7,c_1), d(c_8,c_3)\}$

The distances of the S_1 are already computed on a previous example, so here we only need to compute the distances of S_2. The distances $d(c_5,c_3)$,$d(c_7,c_1)$ coincide with the distances $d(c_3,c_5)$,$d(c_1,c_7)$ respectively. The computations below use the same distance functions between values and cells and also the same weight factors like the previous example.

$$d(c_4, c_3)=\frac{0.5*1/6+0.5*1/6}{0.5+0.5}+\frac{0.5*(|3-7|/|10-1|)}{0.5}=1/6+4/9=11/18$$

$$d(c_6, c_2)=\frac{0.5*1/6+0.5*3/6}{0.5+0.5}+\frac{0.5*(|8-6|/|10-1|)}{0.5}=4/12+2/9=10/18$$

$$d(c_8, c_3)=\frac{0.5*1/6+0.5*1/6}{0.5+0.5}+\frac{0.5*(|9-7|/|10-1|)}{0.5}=1/6+2/9=7/18$$

Now, the Hausdorff distance between the cubes $CUBE_1$ and $CUBE_2$ is equal to the next formula:

$d(CUBE_1,CUBE_2)=max\{max\{S_1\},max\{S_2\}\}=$

$max\{max\{1/6,1/6,5/18\},max\{11/18,5/18, 1/6,10/18,7/18\}\}=$

$max\{5/18,11/18\}=11/18.$

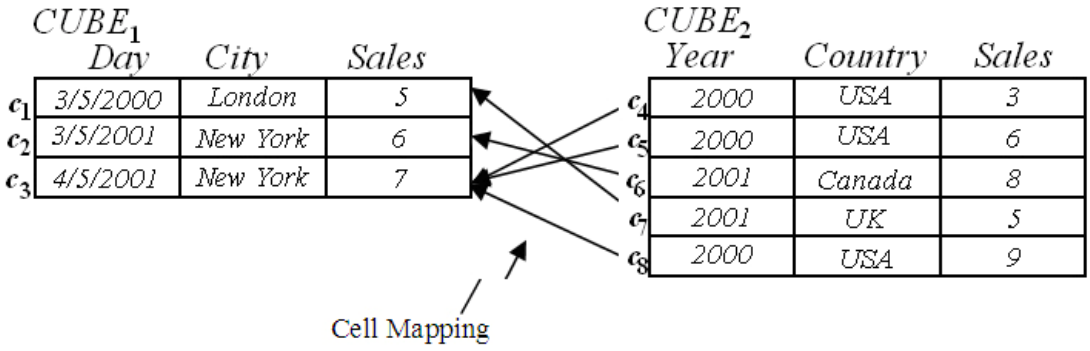


Figure 3.9 Instances of cubes $CUBE_1$ and $CUBE_2$ and the mapping of the cells of the cube $CUBE_2$ to the cells of the cube $CUBE_1$

3.4.3. Distance functions that do not include Mappings

This subsection includes the distance functions that don't include mappings. These functions are the *Weighted Sum* function, the Minkowski family of distance functions, the *Jaccard's Coefficient* and the *minimum of distances* function. The analytical formula of each function is described bellow.

Distance functions between two cubes expressed as a weighted sum. In this category the distance between two cubes can possibly be expressed as a weighted sum over the distances between each cell from one cube to every cell from the other cube. Therefore, the distance can be expressed through the formula: $f : \frac{\sum_{i=1}^{l}\sum_{j=1}^{k} w_{ij} dist(c,c')}{\sum_{i=1}^{l}\sum_{j=1}^{k} w_{ij}}$,

where *dist*(*c, c'*)is the distance between a cell from cube *C* to a cell from cube *C*' and w_{ij} denotes the weight factors assigned to each distance.

Distance functions between two cubes expressed through Minkowski family distances. The distance between two cubes *C* and *C*' can be expressed by making use of a distance function from the Minkowski family. The distance between *C* and *C*' by applying the Minkowski family distances, depending on the values of the parameter *p*, are defined as:

$L_1 = \sum_{i=1}^{l}\sum_{j=1}^{k} dist(c,c')$, 1-norm distance

$L_2 = \sqrt{\sum_{i=1}^{l}\sum_{j=1}^{k} dist(c,c')^2}$, 2-norm distance

$L_p = \sqrt[p]{\sum_{i=1}^{l}\sum_{j=1}^{k} dist(c,c')^p}$, p-norm distance

$L_\infty = \lim_{p\to\infty}\left(\sqrt[p]{\sum_{i=1}^{l}\sum_{j=1}^{k} dist(c,c')^p}\right) =$

$\max\{dist(c_1,c'_1), dist(c_1,c'_2),..., dist(c_1,c'_k),..., dist(c_l,c'_1), dist(c_l,c'_2),..., dist(c_l,c'_k)\}$

infinity norm distance or Chebyshev distance.

Distance functions between two cubes expressed by Jaccard's Coefficient. In this category the distance between two cubes can be expressed in regards to the *Jaccard's coefficient* [ZADB06]. The Jaccard's coefficient is defined as: $dist(C,C') = 1 - \frac{|C \cap C'|}{|C \cup C'|}$. The distance is based on the ratio between the cardinalities of intersection and union of the cubes *C* and *C*'. In addition, based on the Jaccard's coefficient the distance between two cubes can be expressed by applying the Dice's coefficient. For two cubes *C* and *C*' the Dice's coefficient is defined as: $dist(C,C') = \frac{2|C \cap C'|}{|C| + |C'|}$. This formula expresses the similarity between two cubes as the ratio between the cardinality of intersection and the summation of cardinalities of the two cubes.

The Minimum of distances Function. Another option is to express the distance as the minimum distance among all possible distances between the cells of the compared cubes. Therefore the distance between *C* and *C*' is expressed as: $dist(C,C') = \min\{dist(c,c') \mid c \in C, c' \in C'\}$, where *dist*(*c, c'*) is the distance between a cell from cube *C* to a cell from cube *C*'. In case the two cubes are disjoint i.e., $C \cap C' = \emptyset$, then *dist*(*C, C'*) is a positive number, whereas if the two cubes have common cells i.e., $C \cap C' \neq \emptyset$, then *dist*(*C, C'*) is zero.

As a simple example, assume the two cubes from figure 3.7 and ignore the arrows that denote the cell mapping. According to the *minimum of distances* function, the distance between the two cubes is computed through the following formula where j denotes the any cell from $CUBE_2$:

$d(CUBE_1, CUBE_2) = \min_j\{d(c_1, c_j), d(c_2, c_j), d(c_3, c_j)\} \forall j \in \{4,5...,8\} = 1/6$

CHAPTER 4. IMPLEMENTATION AND EXPERIMENTS

This Chapter includes the technical part of this thesis and also the user studies that we conducted in order to examine the user preferences on the distance functions that are described in chapter 3. Thus, in section 4.1 several implementation issues are examined including a short description of the implemented classes and their UML diagram. In section 4.2 we present the findings of the first user study that we conducted in order to examine which of the distance functions between values of dimensions is most preferred by the users. Finally, in section 4.3, we provide the results of the second user study that is conducted taking into account the findings of the previous section. In the second user study users show their preference between the *closest relative* and the *Hausdorff* distance functions.

4.1. Implementation Issues

In this section we will present the implementation part of this thesis, which is organized as follows. In subsection 4.1.1 we describe the architecture of the application and the background of the database and the Database Management system that was used and in subsection 4.1.2 there is the UML diagram and a short description of the implemented classes.

4.1.1. Application Architecture

This section contains the description of the implemented application for the comparison of two OLAP cubes that we call *Cube Comparison OLAP* (*CuCOOL*) tool. The application takes as input two OLAP cubes in the form of two queries and returns their distance taking into account the selected distance functions, firstly between the values of the dimensions, secondly between the cells of the two cubes and finally between the cubes. The code is written in Java and it is implemented in the NetBeans IDE 6.5.1.

The Database Management System (DBMS) that is used is the MySQL Server 5.1. The application connects to the DBMS using the driver MySQL-AB JDBC 5.1.7. The application interacts with the DBMS by sending SQL queries and retrieving the resulting tuples. Further information about the data and the database schema that is used are described analytically in section 4.2.

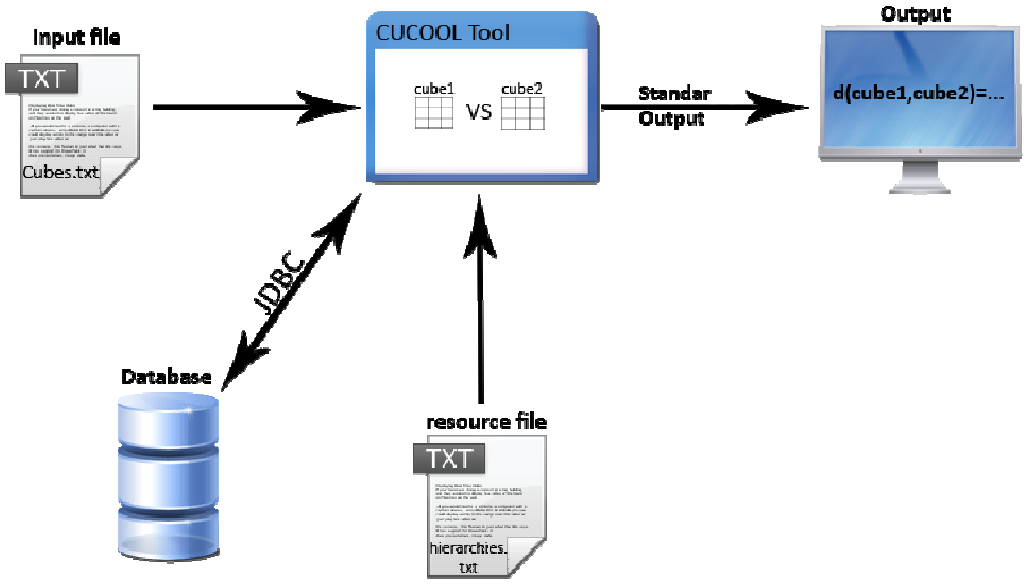


Figure 4.1 *CuCOOL* Tool architecture

4.1.2. UML Diagram and Basic Description of the Implemented Classes

The UML Diagram of the application is shown in the figure 4.2. The part of the implementation that concerns the distance functions includes the classes *Cube_func* and *between_cells* and the interface *functions_between_values*. In addition, there are

several more classes (eg. *Fpath*, *Highway_desc* etc) that implement the function *intercompute*() of the interface *functions_between_values*, according to the distance function between values that we select. The class *between_cells* implements the *weighted sum* function from the functions between two cells of cubes. The *Cube_func* implements the *cell mapping* method as well as the *closest relative* and the *Hausdorff* distance functions.

There are also some classes that are needed to store information about the dimensions, their hierarchies and the levels of each hierarchy. These classes are named *Dim*, *Hierarchy* and *Lev*. Specifically, the class *Hierarchy* contains objects of type *Lev*. So, each object of type *Hierarchy* denotes a hierarchy and contains its levels (*Lev* objects). The class named *Dim is the class* in which the names of the dimensions are stored. Each object *Dim* can contain many *Hierarchy* objects but each *hierarchy* is related to only one dimension.

Parsing. As we mentioned in 4.1.1, the input of the application are two OLAP cube queries. These queries are written in a specific form in a text file called “Cubes.txt”. The form of these queries is shown in figure 4.3. The tag *name* is followed by the name we give to the cube and the tag *Select* is followed by the attributes that we want to retrieve their data. The tag *fact* is followed from the fact table of our database and together with the information of the tag *dimensions* these will create the “From” part of the SQL query. The tag *joins_where* contains the attributes from the dimension tables that we want to connect with the respective foreign keys of the fact table to achieve the join. The tags *where* and *values_where* contain the where conditions of the query. The constraint here is that the order of the information in the *values_where* tag must follow the order of the information in the *where* tag. For more than one where conditions the tag *add_where* must contain the logical connectives (i.e., and, or) in the same order as the conditions in the previous two tags. Finally, the *group_by* tag contains the attribute for the group by condition. The resulting SQL query of the figure 4.3 is:“select ag_level1, ed_level1, hours_per_week from age2, education2, adult where ag_level0=adult.age and ed_level0=adult.education and ag_level2=”27-36” and ed_level2=”Secondary” group_by ed_level0”.

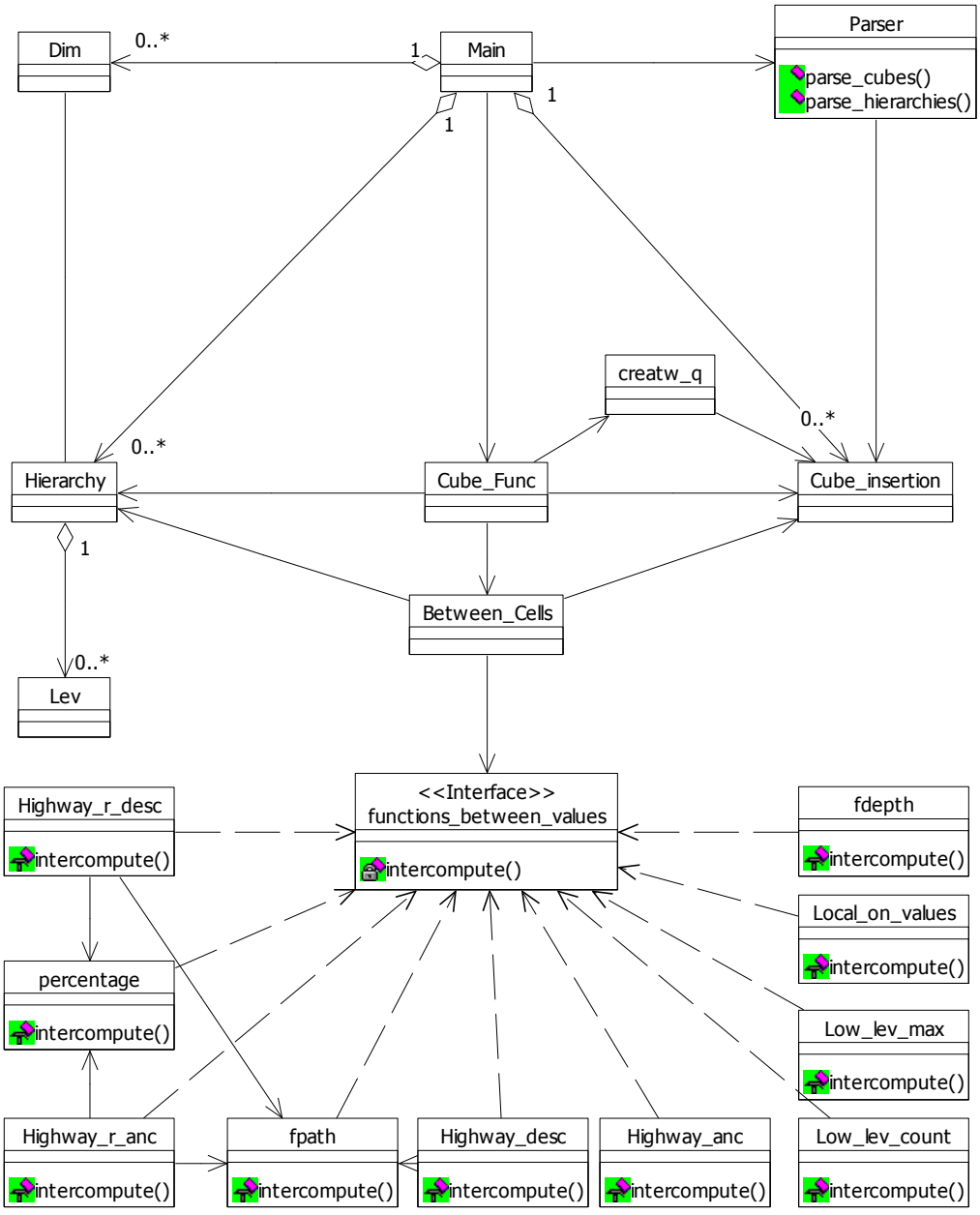


Figure 4.2 The UML Diagram of the *OLAP cube comparison* application

To parse a query given in the form as shown in the figure 4.3, a parser is needed. For this reason the class *Parser* with the function *cube_parser*() is created. Moreover, an extra class named *Cube_Insertion* is created in order to keep the parsed values of each query. Finally, to create the final SQL query, a class *create_q* is constructed. This

class uses the information that is stored in *Cube_Insertion* objects in order to create the appropriate SQL queries.

name cube1
select ag_level1 ed_level1 hours_per_week
fact adult
dimensions age2 education2
joins_where ag_level0=age ed_level0=education
where ag_level2 ed_level2
values_where ="27-36" ="Secondary"
add_where and
groupby ed_level0

Figure 4.3 Form of a query that is given as input in the application

Apart from the queries, the application must be given also the hierarchies of the dimension tables of the database. The file “hierarchies.txt” serves this purpose and an example of such a file is presented in the figure 4.4. In the “hierarchies.txt” file, the word that follows the *name* tag denotes the name of the hierarchy and it must coincide with the dimension table of the database. For example, in figure 4.4 age2 is a dimension table in the database. The word that follows the *FK* tag denotes the foreign key of the dimension table in the fact table, and the words after the tag *levels* denote the levels of the hierarchy with the constraint that every level must be an attribute of the dimension table. The process of parsing for this file is done from the function *parse_hierarchies*() in the *Parser* class. This information is stored in the classes *Hierarchy* and *Lev*.

Name age2
FK age
Levels ag_level4 ag_level3 ag_level2 ag_level1 ag_level0
Name education2
FK education
levels ed_level4 ed_level3 ed_level2 ed_level1 ed_level0

Figure 4.4 A caption from the file “hierarchies.txt”

4.2. User Study for Distances between two Values of Dimensions

In this section we describe a user study we conducted for discovering which distance functions between two values of a dimension seem to be more suitable for user needs. The experiment involved 15 users out of which 10 are graduate students in Computer Science and 5 that are of other backgrounds. In the rest of the paper we refer to the set of users with computer science background as *Users_cs*, the set of users with other background as *Users_non* and the set of all users independently of their background as *Users_all*.

In the experiments we used the "Adult" real data set according to the dimension hierarchies as described in [FuWY05]. This dataset contains the fact table *Adult* and 8 dimension tables which are described in Table 1. The figure 4.5 shows the dimension hierarchies of the dataset "Adult" and the figure 4.6 shows the database schema of the dataset.

Table 4.1 Adult dataset tables

Table	Value Type	# Tuples	# Dim. Levels
Adult fact		30418	-
Age Dim.	Numeric	72	5
Education Dim.	Categorical	16	5
Gender Dim.	Categorical	2	2
Marital Status Dim.	Categorical	7	4
Native Country Dim.	Categorical	41	4
Occupation Dim.	Categorical	14	3
Race Dim.	Categorical	5	3
Work Class Dim.	Categorical	7	4

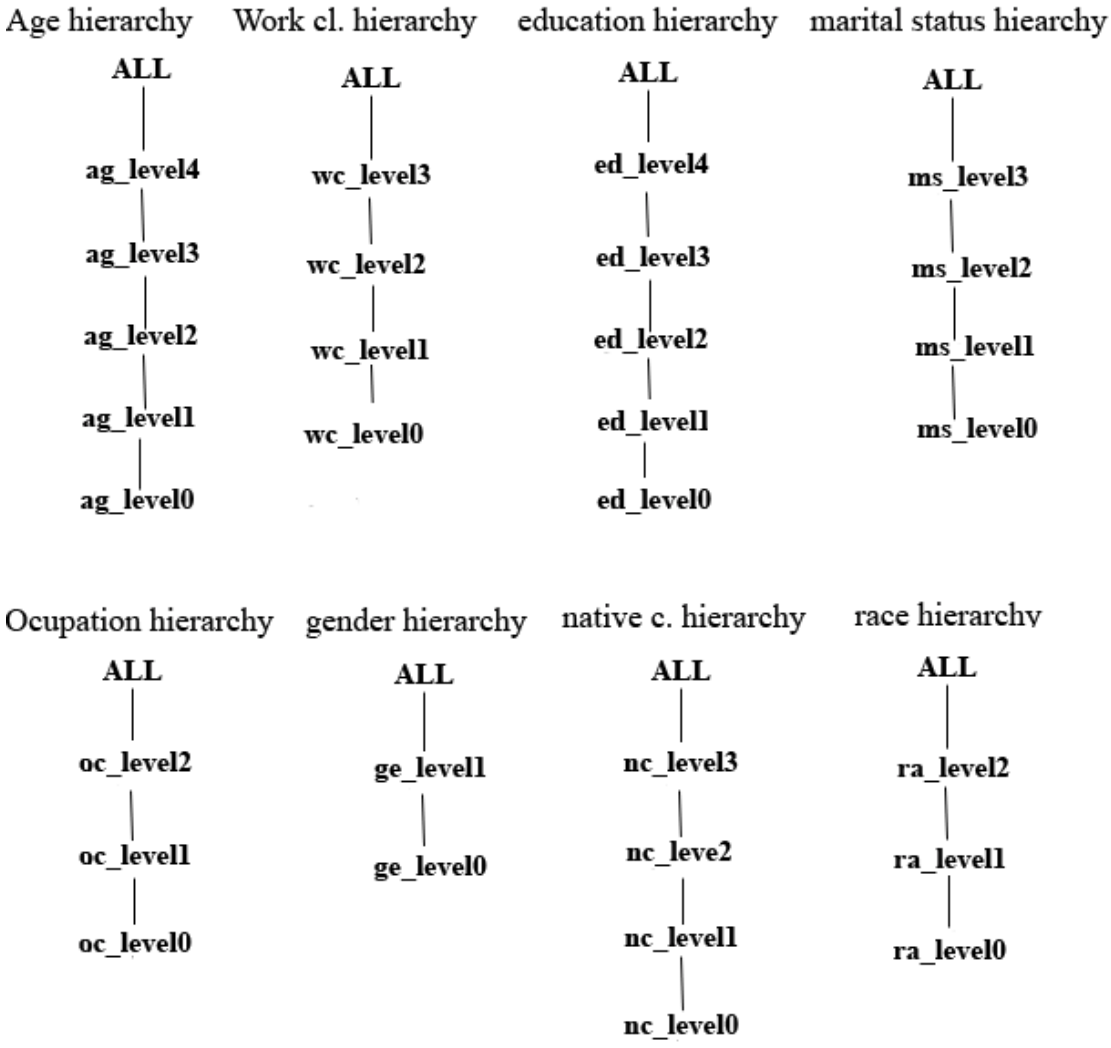


Figure 4.5 Dimension hierarchies of the dataset adult

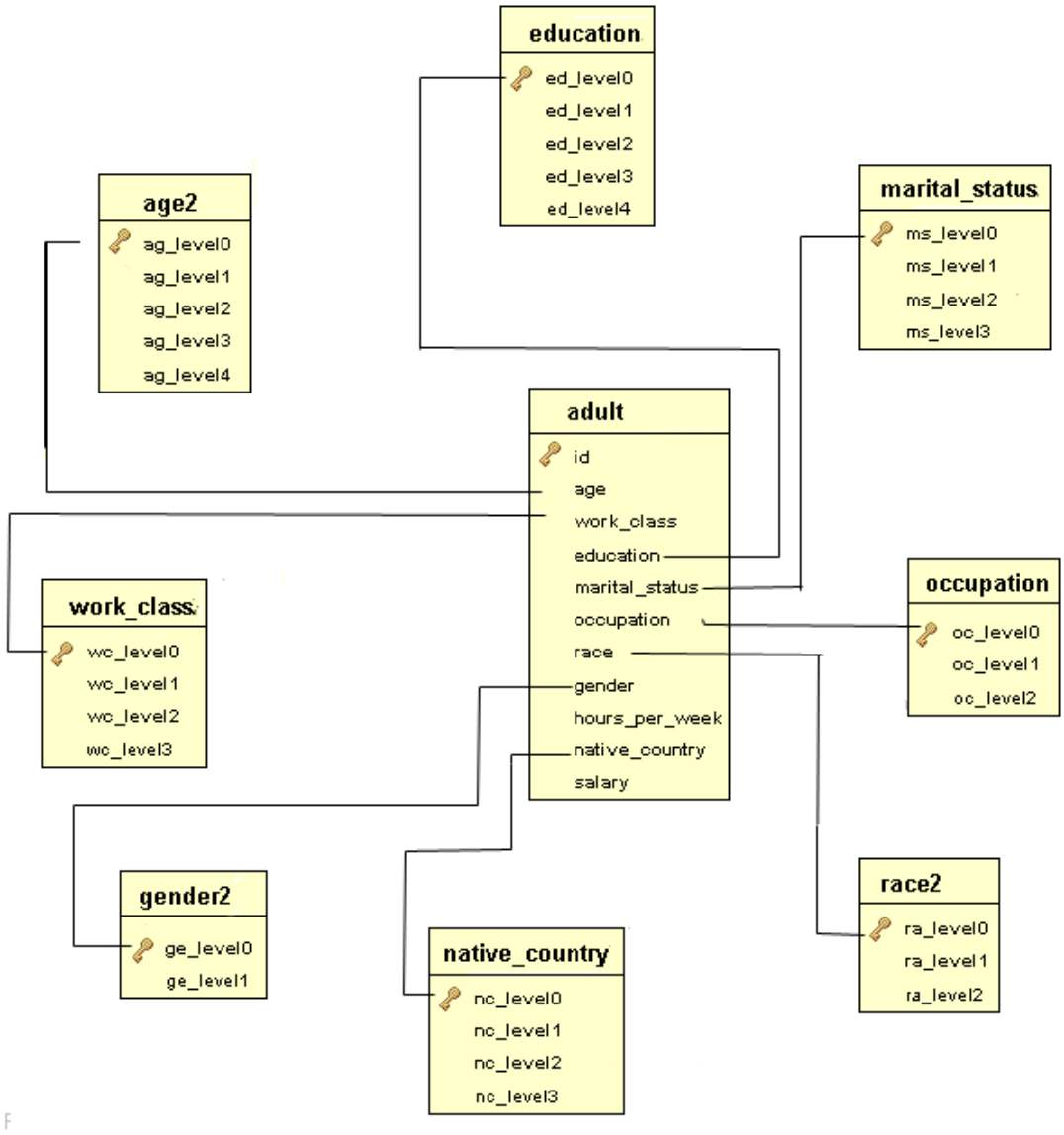


Figure 4.6 Database schema for the Adult dataset

The purpose of the experiment is to assess which distance function between two values is best in regards to the user preferences. Each user was given 14 case scenarios. Each scenario contained a reference cube and a set of cubes, which we call *variant* cubes, that occurred by slightly altering the reference cube. The 14 scenarios included different kinds of cubes in regards to the value types and the different levels of granularity. For each reference cube which was randomly selected, the variant cubes were generated from the fact table by altering the granularity level for one

dimension, or by altering the value range of the reference cube. For instance, assume a reference cube containing the dimension levels Age_level_1, $Education\_level_2$ under the age interval [17, 21]. According to the first type of modification, a variant cube could be generated by changing the dimension level to Age_level_2 or Age_level_0, or changing the level of the Education Dimension. According to the second type of modification, another variant cube could be generated by changing the age interval to [22, 26] or to [17, 26]. Among all possible variations of the reference cube we manually chose the set of variant cubes such that each of them was most similar to the reference cube according to a distance function. In order to observe which distance function is preferred by users depending on the type of data of the cubes, we have organized the 14 scenarios into 3 sets. The first set consists of cubes containing only arithmetic type values (5 scenarios). The second set consists of cubes containing only categorical type values (2 scenarios). The third set consists of cubes containing a combination of both categorical and arithmetic type values (7 scenarios). A sample scenario can be seen in figure 4.7. At this figure the cube with the bolded outline is the reference cube. Due to space limitations all the scenarios used for the user study are not presented here but can be found in the appendix at the end of this thesis.

Cube6

ed_level1	nc_level1	salary
Bachelors	Central-Europe	<=50K
Senior-Secondary	Eastern-Europe	<=50K
Junior-Secondary	Southern-Europe	<=50K
Assoc-acdm	Western-Europe	<=50K

Cube6_1

ed_level1	nc_level1	salary
Assoc-acdm	Western-Europe	<=50K
5th-6th	Southern-Europe	>50K
Masters	Central-Europe	<=50K
Senior-Secondary	Western-Europe	<=50K
Some-college	Western-Europe	<=50K
Bachelors	Central-Europe	<=50K

Cube6_2

ed_level1	nc_level1	salary
Masters	Eastern-Asia	>50K
Masters	Middle-East	>50K
Senior-Secondary	Southeastern-Asia	>50K
Bachelors	Southern-Asia	>50K

Cube6_3

ed_level1	nc_level1	salary
Bachelors	Central-Europe	>50K
Senior-Secondary	Eastern-Europe	>50K
Assoc-voc	Southern-Europe	>50K
Bachelors	Western-Europe	>50K

Cube6_4

ed_level2	nc_level1	salary
University	Central-Europe	<=50K
Secondary	Eastern-Europe	<=50K
Secondary	Southern-Europe	<=50K
Assoc	Western-Europe	<=50K

Figure 4.7 Sample scenario

Table 4.2 Notation of distance functions used in the experiment

Family	Abbr.	Distance function name
Local	δ_{M}	Manhattan
Aggregation	$\delta_{Low,c}$	With respect to a lower level of hierarchy where f_{aggr} =count
	$\delta_{Low,m}$	With respect to a lower level of hierarchy where f_{aggr} = max
Hierarchical Path	$\delta_{LCA,P}$	Lowest common ancestor through f_{path}
	$\delta_{LCA,D}$	Lowest common ancestor through f_{depth}
Percentage	$\delta_{\%}$	Applying percentage function
Highway	δ_{Anc}	With respect to an ancestor x_y
	δ_{Desc}	With respect to a descendant y_x
	$\delta_{H,Desc}$	Highway, selecting the representative from a descendant
	$\delta_{H,Anc}$	Highway, selecting the representative from an ancestor

In each scenario, the users were asked to select the variant cube that seemed more similar to the reference cube based on their personal criteria. The distance functions that have been used in the experiment are shown in Table 2, where the first column shows the family in which each distance function belongs to according to Chapter 3. In the second column there is an abbreviated name for each function. To compute the distance between two cubes, the *Closest Relative* distance function is used (section 3.4.2). The distance between two cells of cubes is the weighted sum of the partial distances of the two values, one from each cell, with all weights set to 1 (section 3.3).

Table 4.3 Top three most frequent distance functions for each user group.

	Users_all	Users_cs	Users_non
$\delta_{LCA,P}$	40.47%	38.57%	44.28%
δ_{Anc}	18.09%	20%	14.28%
$\delta_{H,Desc}$	9.52%	10.71%	7.14%

The analysis of the collected data provides several findings. The first finding concerns the *top three most preferred distance functions* measured over the detailed data for all scenarios and all users. It is remarkable that the top three distance functions for each of the user groups were the same and with the same ordering and specifically, these

are the $\delta_{LCA,P}$, the δ_{Anc} and the $\delta_{H,Desc}$. The frequencies for each one of the top three distance functions in each group of users is shown in Table 4.3.

The second finding concerns *the most preferred function by users depending on the type of data the cubes contained*. Table 4.4 summarizes the result of the most frequent distance function for each set of scenarios and each set of users. We observe that for the *categorical* type of cubes, all user groups prefer the $\delta_{LCA,P}$ distance function, whereas for the *arithmetic* and the *arithmetic & categorical* sets, the functions that users mainly prefer are the $\delta_{LCA,P}$ and δ_{Anc}. More than one distance functions appear as winners in Table 4 due to ties in the frequency of occurrences for each function.

Table 4.4 The most frequent distance function for each set of scenarios.

	Users_all	Users_cs	Users_non
Arithmetic	δ_{Anc}	$\delta_{LCA,P}$, $\delta_{H,Desc}$, δ_{Anc}	$\delta_{LCA,P}$
Categorical	$\delta_{LCA,P}$	$\delta_{LCA,P}$	$\delta_{LCA,P}$
Arithmetic & Categorical	δ_{Anc}	δ_{Anc}	$\delta_{LCA,P}$, δ_{Anc}

The third finding concerns the *winner distance function per scenario*. For every scenario, we take into account the 15 occurrences by all users and see which distance function is the most frequent. We call this function the winner function of the scenario. The most frequent winner function was $\delta_{LCA,P}$. The percentages were 35.71% for the *Users_all* group, 35,71% for the *Users_cs* group and 57.14% for the *Users_non* group. The most frequent function for 14 users was the $\delta_{LCA,P}$ function. For one user from the *Users_cs* group the most frequent function was the $\delta_{LCA,D}$.

The fourth finding concerns the *diversity and spread* of user choices. There are two major findings: (a) All functions were picked by some user and (b) there are certain functions that appeared as user choices for all users of a user group. Specifically, functions $\delta_{LCA,P}$, $\delta_{H,Desc}$ and δ_{Anc} were selected at least once by users of group *Users_cs*. Similarly, functions $\delta_{LCA,P}$, $\delta_{Low,m}$ and δ_{Anc} were selected at least once by *Users_non*.

The fifth finding concerns the *most preferred family of functions*. Table 4.5 depicts the absolute number of appearances of each distance function family per user group. The most preferred family of distance functions is the *Hierarchy Path* family, which also contains the top one most preferred distance function $\delta_{LCA,P}$. Moreover, we observe that the ranking of the distance function families was exactly the same for each user group.

Table 4.5 Frequencies of preferred distances within each user group for each distance family.

	Local	Aggregation	Hierarchy Path	Percentage	Highway
Users_cs	1	9	69	9	52
Users_non	2	5	34	5	24
Users_all	3	14	103	14	76

The *selection stability* (e.g., how stable are users answers at the same questions) of users was the sixth observation. The *selection stability* was determined by the following results, where the 13th and the 14th scenario were a reordering of the 3rd and 10th scenario respectively. 4 out of 5 users from the set of *Users_non*, 6 out of 10 users from the set of *Users_cs* (consequently, 10 users from *Users_all* set) selected the same function for both of the two similar scenarios. The rest of the users selected the same function for only one out of the two repeated scenarios.

Summary. Overall, the findings indicate that the most preferred distance function is the $\delta_{LCA,P}$, which is expressed in regards to the shortest path of a hierarchy dimension. Apart from the $\delta_{LCA,P}$, the distance functions δ_{Anc} and $\delta_{H,Desc}$ were widely chosen by users. In addition, the most preferred distance function family is the *Hierarchy Path* family.

4.3. User Study for Distances between two Cubes

This second user study is a follow up of the previous user study. In the previous user study the overall observation was that the users prefer the $\delta_{LCA,P}$ distance function between two values of the same dimension. Based on this result and also by setting as the distance function between cells the *weighted sum* function we set up the second

user study such that we can examine which distance function between two cubes the users prefer. Specifically, we try to find out which distance function among the two functions that include the *cell mapping* method (section 3.4.1) is most closely related to the human perception. These two distance functions are namely the *closest relative* and the *Hausdorff* distance function (section 2.4.2). The table 4.6 shows the distance functions that were used in this user study

The user study contained 14 new scenarios. Each scenario included 4 cubes named *A*, *B*, *C* and *D*. The cube *A* in every scenario was the reference cube. The users were asked to order the rest of the three cubes from the most similar to the less similar when compared to the cube *A*. The cubes *B*, *C* and *D* were chosen such that one of them was the closest to the cube *A* according to the *closest relative* function and another was the closest to cube *A* according to the *Hausdorff* distance function. The remaining cube was chosen to be the most distant from cube *A* for both distance functions. A sample scenario can be seen in figure 4.8. In this figure the cube which is filled with light blue color is the reference cube. Due to space limitations all the scenarios used for this user study are not presented here but can be found in the appendix at the end of this thesis.

All scenarios were uploaded as jpeg pictures in an html page where users were asked to complete an answer sheet and send it back to us via email. The url link of this page was sent via a social network and also by email at the email-list of the graduate students of the Computer Science Department of the University of Ioannina.

In order to test a user's answer reliability, in the 6^{th} scenario the cube *B* was identical with the cube *A*. Moreover, the 13^{th} and 14^{th} scenarios were replicas of the 5^{th} and 9^{th} scenarios respectively with a reordering on the columns of the cubes. This was done in order to measure the user stability of their choices.

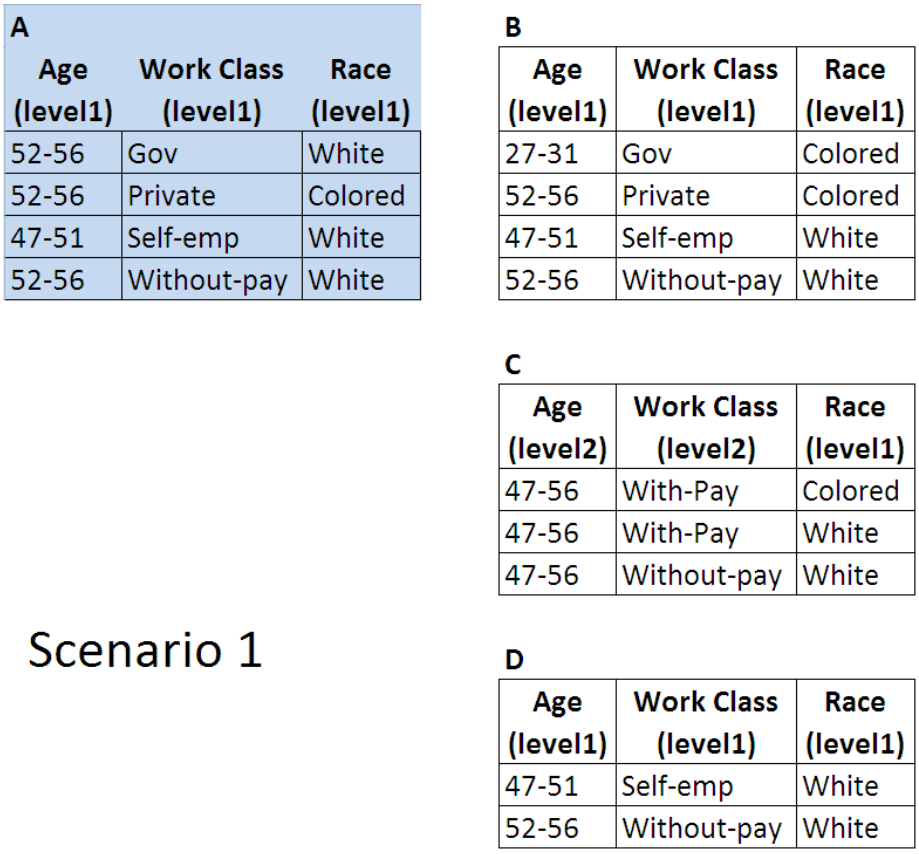


Figure 4.8 Sample scenario

The 12 first scenarios can be divided into three groups according to the weights in the distance function between cells. The first 4 scenarios consist of cubes that they do not include measures. We refer to this group as the *no_measures* group. The next 4 scenarios consist of cubes that include measures where the weight factors on measures and dimensions in the function *between cells* are not equal. Specifically, assuming that cubes consist of k dimensions and l measures, the weight factors for the dimensions was set to be $k/(l+k)$ and for the measures was set to be $l/(l+k)$. We refer to this group as the *not_equal* group. Finally, the last four scenarios consist of cubes that include measures and the weight factors on the measures and on the dimensions in the *between cells* distance function are equal and set to 0.5. We refer to this group as the *equal* group.

Table 4.6 The distance functions that are used in the second user study

Distance functions between two cubes	Hausdorff
	Closest relative
Distance function between two cells of cubes	weighted sum
Distance function between two values of a dimension	δLCA,P
Distance function between two measures	Manhattan

The number of users that responded with an answer sheet was 39. Two from the 39 users did not choose the cube *B* in the sixth scenario as the most similar to the cube *A*. For that reason their answers were not taken into consideration. We refer to the remaining 37 users as *valid_users*.

The first finding of this user study concerns the most *frequent distance function* that was chosen from the users as their first choice. Among all the 11 (scenarios) * 37 (users) = 407 answers (the sixth scenario is excluded), 232 times ($\approx$57%) the users gave as their first choice the cube that represents the *closest relative* distance function. The cube that represents the *Hausdorff* distance function was chosen 154 times ($\approx$38%) as the first choice of the users. Only 21 times ($\approx$5%) the users chose the most distant cube as their first choice. The summarization of the above results is shown in the table 4.7.

Table 4.7 Frequency of chosen as first distance function among all the 444 answers

	Frequency	Percentage
Hausdorff	154	38%
Closest relative	232	57%
Most distant cube	21	5%

The second finding of the user study concerns the stability of the user choices. As we mentioned before, the 13th and 14th scenario were replicas of the 5th and 9th scenario respectively. In each of these two scenarios a user that orders the cubes in the same way as in the original scenario is denoted as *user_OK*. A user that gave the same answer for the most similar cube but the order of the other cubes was not the same is denoted as *user_Half_OK*. Finally, a user that was denoted as *user_OK* for both

replicas scenarios, or denoted as *user_OK* for the one replica scenario and *user_Half_OK* for the other replica scenario is denoted as *user_ Stable*. According to the answers of the valid 37 users of this user study, in the 13th scenario there were 28 *user_OK* users and 5 *user_Half_OK* users. In the 14th scenario there were 19 *user_OK* users and 8 *user_Half_OK* users. The 24 of the 37 ($\approx$65%) users were *user_Stable* users. We believe that a 65% is a safe number that can ensure the stability and reliability of their answers. The table 4.8 summarizes the above results and percentages.

Table 4.8 User stability

	User_OK		user_Half_OK		user_Stable	
	Frequency	Percentage	Frequency	Percentage	Frequency	Percentage
13th scenario	28	75%	5	13%	24	65%
14th Scenario	19	51%	8	21%	24	65%

The third observation concerns the *wining function* per scenario. The term *wining function* refers to the function that was mostly selected as the first choice from the users in one scenario. The *closest relative* function was the wining function for 6 scenarios and the *Hausdorff* function was the wining function for the rest 5 scenarios. These results cannot ensure that one of the two functions is more preferred than the other.

The fourth observation concerns the *winner function* per scenario group. For a group of scenarios its *winner function* occurs to be the function that appeared as *wining function* in most scenarios of the group. For the *no_measures* group the *winner function* was the *closest relative* function which it was the *wining function* for the 3 out of the 4 scenarios. For the *not_equal* group the *winner function* was the Hausdorff which it was the *winning function* for the 2 out of the 3 scenarios. Finally, for the group *equal*, in two scenarios the wining function was the *closest relative* function and in two scenarios the wining function was the *Hausdorff* function. The above results reveal a user preference in the *closest relative* function for scenarios that do no include measures. On the other hand for the other types of scenarios the results are not

clear. The analytical results of the third and fourth observation are presented in table 4.9.

Table 4.9 The *winning functions* and the *winner functions*

Scenario Group	**Scenario**	**Winning function**	**Winner function**
no_measures	Scenario1	*Closest relative*	*Closest relative*
	Scenario2	*Closest relative*	
	Scenario3	*Closest relative*	
	Scenario4	*Hausdorff*	
not_equal	Scenario5	*Hausdorff*	*Hausdorff*
	Scenario7	*Closest relative*	
	Scenario8	*Hausdorff*	
equal	Scenario9	*Hausdorff*	-
	Scenario10	*Hausdorff*	
	Scenario11	*Closest relative*	
	Scenario12	*Closest relative*	

CHAPTER 5. CONCLUSIONS

This thesis presented a variety of distance functions that can be used in order to compute the similarity between two OLAP cubes. The functions were described with respect to the properties of the dimension hierarchies and based on these they were grouped into functions that can be applied (a) between two values from a dimension of a multidimensional space, (b) between two points of a multidimensional space and (c) between two sets of points of a multidimensional space.

In order to assess which distance functions are more close to human perception, we conducted two user study analysis. The first user study analysis was conducted in order to discover, which distance function between two values of a dimension is best in regards to the user needs and data type. Our findings indicated that the distance function $\delta_{\mathrm{LCA,P}}$, which is expressed as the length of the path between two values and their common ancestor in the dimension's hierarchy was the most preferred by users in our experiments. Two more functions were widely chosen by users. These were the highway functions δ_{Anc} that is expressed in regards to the ancestor x_{y} and $\delta_{\mathrm{H,Desc}}$ that is expressed by selecting the representative from a descendant.

The second user study we conducted, took into account the results of the first user study analysis. Specifically, the second user study analysis aimed in discovering which distance function (the *closest relative* or the *Hausdorff* distance function) from the category of distance function between two data cubes, users prefer. The findings of this user study analysis indicated that the closest relative distance function was rather preferred by users in contrast to the Hausdorff distance functions.

Future work can be pursued in various directions including (a) the deeper examination of the presented families of functions with more complicated scenarios and (b) the discovery of the foundational reasons for the observed user preferences.

REFERENCES

[Cou] The OLAP Council. The OLAP benchmark. http://www.olapcouncil.org

[JB05] Cliff Joslyn and William J. Bruno. Weighted pseudo-distances for categorization in semantic hierarchies. In ICCS, pages 381– 395, 2005.

[JK00] Jiawei Han, Micheline Kamber: Data Mining: Concepts and Techniques Morgan Kaufmann 2000

[JO04] Cliff Joslyn. Poset Ontologies and Concept Lattices as Semantic Hierarchies. In ICCS, pages 287-302, 2004

[JMFH04] Cliff Joslyn, Susan M. Mniszewski, Andy W. Fulmer, and Gary Heaton. The gene ontology categorizer. In ISMB/ECCB (Sup- plement of Bioinformatics), pages 169–177, 2004.

[LDH+08] Cindy Xide Lin, Bolin Ding, Jiawei Han, Feida Zhu, and Bo Zhao. Text cube: Computing IR measures for multidimensional text database analysis. In ICDM, pages 905–910, 2008. 2

[MI95] G.A. Miller. WordNet: A lexical Database for English. Comm. ACM , pages 39-41, 1995

[Mic98a] Microsoft corporation. Microsoft decision support services version 1.0, 1998

[MUFL06] Heiko Müller Johann-Christoph Freytag and Ulf Leser. Describing differences between databases. In CIKM, pages 612-621, 2006

[PP03] Dennis Pedersen and Torben Bach Pedersen. Achieving adaptivity for olap-xml federations. In DOLAP, pages 25–32, 2003.

[PRP02] Dennis Pedersen, Karsten Riis, and Torben Bach Pedersen. Xml-extended olap querying. In SSDBM, pages 195–206, 2002.

[Sar99] Sunita Sarawagi. Explaining differences in multidimensional aggregates. In VLDB, pages 42–53, 1999.

[Sar00] Sunita Sarawagi. User-adaptive exploration of multidimensional data. In VLDB, pages 307–316, 2000.

[Sar01] Sunita Sarawagi. idiff: Informative summarization of differences in multidimensional aggregates. Data Min. Knowl. Dis- cov., 5(4):255–276, 2001.

[SaSc05] P. Sanders and D. Schultes. Highway Hierarchies Hasten Exact Shortest PathQueries. In ESA, LNCS 3669, pages 568-579, Springer, 2005.

[SJ95] Simone Santini and Ramesh Jain. Similarity matching. In ACCV, pages 571–580, 1995.

[SJ99] Simone Santini and Ramesh Jain. Similarity measures. IEEE Trans. Pattern Anal. Mach. Intell., 21(9):871–883, 1999. 3

[SS01] Gayatri Sathe and Sunita Sarawagi. Intelligent rollups in multidimensional olap data. In VLDB, pages 531–540, 2001.

[SS05] Peter Sanders, Dominik Schultes: Highway Hierarchies Hasten Exact Shortest Path Queries. In ESA, pages 568-579, 2005

[TDP06] Igor Timko, Curtis E. Dyreson, and Torben Bach Pedersen. Pre-aggregation with probability distributions. In DOLAP, pages 35–42, 2006.

[VS00] P. Vassiliadis, S. Skiadopoulos, “Modelling and Optimisation Issues for Multidimensional Databases”, In CAiSE '00, pp. 482-497, Stockholm, Sweden, 5-9 June 2000.

[XH07] Dong Xin and Jiawei Han. Integrating olap and ranking: The ranking-cube methodology. In ICDE Workshops, pages 253– 256, 2007.

[YZM03] Yuhua Li, Zuhair Bandar and David McLean. An Approach for Measuring Semantic Similarity between Words Using Multiple Information Sources. In IEEE Trans. Knowl. Data Eng, pages 871-882, 2003

[YP04] Xuepeng Yin and Torben Bach Pedersen. Evaluating xmlextended olap queries based on a physical algebra. In DOLAP, pages 73–82, 2004.

[ZADB06] Pavel Zezula, Giuseppe Amato, Vlastislav Dohnal and Michal Batko. Similarity Search: The Metric Space Approach. Springer, 2006

APPENDIX

Scenarios of the 1st user study

Cube 1

ag_level2	ed_level2	oc_level0
37-46	Assoc	Craft-repair
37-46	Elementary	Machine-op-inspct
37-46	Post-grad	Exec-managerial
37-46	Preschool	Machine-op-inspct
37-46	Secondary	Handlers-cleaners
37-46	Some-college	Exec-managerial
37-46	University	Adm-clerical

Cube1_1

ag_level3	ed_level2	oc_level0
37-56	Assoc	Craft-repair
37-56	Elementary	Machine-op-inspct
37-56	Post-grad	Exec-managerial
37-56	Preschool	Machine-op-inspct
37-56	Secondary	Handlers-cleaners
37-56	Some-college	Exec-managerial
37-56	University	Adm-clerical

Cube1_2

ag_level2	ed_level2	oc_level0
47-56	Assoc	Prof-specialty
37-46	Elementary	Machine-op-inspct
37-46	Post-grad	Exec-managerial
47-56	Preschool	Machine-op-inspct
37-46	Secondary	Handlers-cleaners
37-46	Some-college	Exec-managerial
37-46	University	Adm-clerical

Cube1_3

ag_level2	ed_level1	oc_level0
37-46	Assoc-voc	Craft-repair
37-46	5th-6th	Machine-op-inspct
37-46	Masters	Exec-managerial
37-46	Preschool	Machine-op-inspct
37-46	Senior-Secondary	Handlers-cleaners
37-46	Some-college	Exec-managerial
37-46	Bachelors	Adm-clerical

Cube1_6

ag_level2	ed_level2	oc_level0
37-46	University	Adm-clerical
37-46	Secondary	Armed-Forces
37-46	Assoc	Craft-repair
37-46	Post-grad	Exec-managerial
37-46	Secondary	Farming-fishing
37-46	Secondary	Handlers-cleaners
37-46	Elementary	Machine-op-inspct
37-46	Secondary	Other-service
37-46	Secondary	Priv-house-serv
37-46	Post-grad	Prof-specialty
37-46	Post-grad	Protective-serv
37-46	Secondary	Sales
37-46	Some-college	Tech-support
37-46	Secondary	Transport-moving

Cube1_4

ag_level2	ed_level2	oc_level0
37-46	Secondary	Handlers-cleaners
37-46	Secondary	Other-service
37-46	Secondary	Sales

Cube1_5

ag_level2	ed_level3	oc_level0
37-46	Post-Secondary	Craft-repair
37-46	Without-Post-Secondary	Machine-op-inspct
37-46	Post-Secondary	Exec-managerial
37-46	Without-Post-Secondary	Machine-op-inspct
37-46	Without-Post-Secondary	Handlers-cleaners
37-46	Post-Secondary	Exec-managerial
37-46	Post-Secondary	Adm-clerical

Figure A.1 Cube scenario 1

Cube2

nc_level1	ed_level2	oc_level1
Western-Europe	Assoc	white-collar
Central-Europe	Elementary	Blue-collar
Southern-Asia	Post-grad	white-collar
Southern-Asia	Preschool	Blue-collar
Western-Europe	Secondary	white-collar
Southern-Asia	Some-college	Blue-collar
Southern-Asia	University	white-collar

Cube2_1

nc_level1	ed_level2	oc_level1
South-America	Assoc	Other
South-America	Elementary	Blue-collar
South-America	Post-grad	white-collar
Southern-Asia	Preschool	Blue-collar
South-America	Secondary	Other
South-America	Some-college	white-collar
South-America	University	white-collar

Cube2_2

nc_level1	ed_level1	oc_level1
Western-Europe	Assoc-acdm	white-collar
Central-Europe	7th-8th	Blue-collar
Southern-Asia	Masters	white-collar
Southern-Asia	Preschool	Blue-collar
Western-Europe	Senior-Secondary	white-collar
Southern-Asia	Some-college	Blue-collar
Southern-Asia	Bachelors	white-collar

Cube2_3

nc_level1	ed_level3	oc_level1
Western-Europe	Post-Secondary	white-collar
Central-Europe	Without-Post-Secondary	Blue-collar
Southern-Asia	Post-Secondary	white-collar
Southern-Asia	Without-Post-Secondary	Blue-collar
Western-Europe	Without-Post-Secondary	white-collar
Southern-Asia	Post-Secondary	Blue-collar
Southern-Asia	Post-Secondary	white-collar

Cube2_4

nc_level1	ed_level2	oc_level1
Western-Europe	Secondary	Blue-collar
Southern-Asia	Secondary	Other
Southern-Asia	University	white-collar

Cube2_5

nc_level1	ed_level1	oc_level1
Western-Europe	Assoc-acdm	white-collar
Central-Europe	7th-8th	Blue-collar
Southern-Asia	Masters	white-collar
Southern-Asia	Preschool	Blue-collar
Western-Europe	Senior-Secondary	white-collar
Southern-Asia	Some-college	Blue-collar
Southern-Asia	Bachelors	white-collar

Cube2_6

nc_level1	ed_level2	oc_level1
Western-Europe	Secondary	Blue-collar
Southern-Asia	Secondary	Other
Southern-Asia	University	white-collar

Figure A.2 Cube scenario 2

Cube3

ag_level1	wc_level1
27-31	Gov
27-31	Private
27-31	Self-emp
27-31	Without-pay

Cube3_1

ag_level1	wc_level1
22-26	Gov
22-26	Private
22-26	Self-emp
22-26	Without-pay

Cube3_2

ag_level1	wc_level0
27-31	State-gov
27-31	Private
27-31	Self-emp-not-inc
27-31	Without-pay

Cube3_3

ag_level1	wc_level1
27-31	Private
27-31	Without-pay

Cube3_4

ag_level1	wc_level1
27-31	Gov
27-31	Gov
27-31	Private
27-31	Self-emp
27-31	Self-emp
27-31	Gov
27-31	Without-pay

Cube3_5

ag_level2	wc_level1
27-36	Gov
27-36	Private
27-36	Self-emp
27-36	Without-pay

Figure A.3 Cube scenario 3

Cube4

ag_level1	wc_level1	ra_level1
52-56	Gov	White
52-56	Private	Colored
47-51	Self-emp	White
52-56	Without-pay	White

Cube4_1

ag_level1	wc_level1	ra_level1
37-41	Gov	White
37-41	Private	White
47-51	Self-emp	White
62-66	Without-pay	White

Cube4_4

ag_level1	wc_level1	ra_level1
52-56	Gov	White
52-56	Private	Colored
52-56	Self-emp	White
52-56	Without-pay	White

Cube4_2

ag_level1	wc_level1	ra_level1
52-56	Gov	White
47-51	Private	White
47-51	Self-emp	White
52-56	Without-pay	White

Cube4_5

ag_level1	wc_level1	ra_level1
27-31	Gov	Colored
52-56	Private	Colored
47-51	Self-emp	White
52-56	Without-pay	White

Cube4_3

ag_level1	wc_level1	ra_level1
37-41	Gov	White
37-41	Private	White
42-46	Self-emp	White
42-46	Without-pay	White

Cube4_6

ag_level1	wc_level1	ra_level1
47-51	Self-emp	White
52-56	Without-pay	White

Cube4_7

ag_level1	wc_level2	ra_level1
52-56	With-Pay	White
52-56	With-Pay	Colored
47-51	With-Pay	White
52-56	Without-pay	White

Cube4_8

ag_level2	wc_level1	ra_level1
47-56	Gov	White
47-56	Private	Colored
47-56	Self-emp	White
47-56	Without-pay	White

Figure A.4 Cube scenario 4

Cube5

ed_level1	wc_level1	ms_level1	ra_level1
Assoc-acdm	Gov	Never-married	White
5th-6th	Private	Partner-present	White
Masters	Private	Partner-present	White
Preschool	Gov	Never-married	White
Senior-Secondary	Private	Partner-absent	White
Some-college	Gov	Partner-present	White
Bachelors	Gov	Never-married	White

Cube5_1

ed_level1	wc_level1	ms_level1	ra_level1
Bachelors	Gov	Never-married	White
Senior-Secondary	Private	Partner-absent	White
Bachelors	Self-emp	Partner-present	White

Cube5_2

ed_level1	wc_level1	ms_level1	ra_level1
Some-college	Gov	Partner-present	White
Bachelors	Gov	Partner-present	White
Senior-Secondary	Private	Partner-absent	White
Senior-Secondary	Self-emp	Partner-absent	White
Bachelors	Self-emp	Partner-present	White
Bachelors	Gov	Never-married	White
Senior-Secondary	Without-pay	Never-married	White

Cube5_3

ed_level1	wc_level1	ms_level1	ra_level1
1st-4th	Private	Partner-present	White
5th-6th	Private	Partner-present	White
7th-8th	Gov	Partner-present	White
Assoc-acdm	Gov	Never-married	White
Assoc-voc	Gov	Partner-present	White
Bachelors	Gov	Never-married	White
Doctorate	Private	Partner-present	White
Junior-Secondary	Private	Partner-present	White
Masters	Private	Partner-present	White
Preschool	Gov	Never-married	White
Prof-school	Private	Partner-present	White
Senior-Secondary	Private	Partner-absent	White
Some-college	Gov	Partner-present	White

Cube5_4

ed_level1	wc_level1	ms_level1	ra_level1
Assoc-acdm	Private	Never-married	Colored
7th-8th	Private	Partner-present	Colored
Masters	Self-emp	Partner-absent	Colored
Preschool	Private	Never-married	Colored
Senior-Secondary	Gov	Never-married	Colored
Some-college	Private	Partner-present	Colored
Bachelors	Private	Partner-present	Colored

Cube5_5

ed_level1	wc_level2	ms_level1	ra_level1
Assoc-acdm	With-Pay	Never-married	White
5th-6th	With-Pay	Partner-present	White
Masters	With-Pay	Partner-present	White
Preschool	With-Pay	Never-married	White
Senior-Secondary	With-Pay	Partner-absent	White
Some-college	With-Pay	Partner-present	White
Bachelors	With-Pay	Never-married	White

Cube5_6

ed_level2	wc_level1	ms_level1	ra_level1
Assoc	Gov	Never-married	White
Elementary	Private	Partner-present	White
Post-grad	Private	Partner-present	White
Preschool	Gov	Never-married	White
Secondary	Private	Partner-absent	White
Some-college	Gov	Partner-present	White
University	Gov	Never-married	White

Figure A.5 Cube scenario 5

Cube6

ed_level1	nc_level1	salary
Bachelors	Central-Europe	<=50K
Senior-Secondary	Eastern-Europe	<=50K
Junior-Secondary	Southern-Europe	<=50K
Assoc-acdm	Western-Europe	<=50K

Cube6_1

ed_level1	nc_level1	salary
Assoc-acdm	Western-Europe	<=50K
5th-6th	Southern-Europe	>50K
Masters	Central-Europe	<=50K
Senior-Secondary	Western-Europe	<=50K
Some-college	Western-Europe	<=50K
Bachelors	Central-Europe	<=50K

Cube6_2

ed_level1	nc_level1	salary
Masters	Eastern-Asia	>50K
Masters	Middle-East	>50K
Senior-Secondary	Southeastern-Asia	>50K
Bachelors	Southern-Asia	>50K

Cube6_3

ed_level1	nc_level1	salary
Bachelors	Central-Europe	>50K
Senior-Secondary	Eastern-Europe	>50K
Assoc-voc	Southern-Europe	>50K
Bachelors	Western-Europe	>50K

Cube6_4

ed_level2	nc_level1	salary
University	Central-Europe	<=50K
Secondary	Eastern-Europe	<=50K
Secondary	Southern-Europe	<=50K
Assoc	Western-Europe	<=50K

Figure A.6 Cube scenario 6

Cube7os

ed_level1	nc_level1	hours_per_week
Senior-Secondary	Central-Europe	55
Bachelors	Eastern-Europe	65
Senior-Secondary	Southern-Europe	75
Senior-Secondary	Western-Europe	62

Cube7_2

ed_level1	nc_level2	hours_per_week
Bachelors	Europe	40
Senior-Secondary	Europe	50
Junior-Secondary	Europe	40
Assoc-acdm	Europe	40

Cube7_3

ed_level1	nc_level1	hours_per_week
Bachelors	Central-Europe	40
Some-college	Eastern-Europe	40
Junior-Secondary	Southern-Europe	40
Assoc-acdm	Western-Europe	40

Cube7_6

ed_level2	nc_level1	hours_per_week
University	Central-Europe	40
Secondary	Eastern-Europe	50
Secondary	Southern-Europe	40
Assoc	Western-Europe	40

Cube7_1

ed_level1	nc_level1	hours_per_week
Assoc-acdm	Western-Europe	40
5th-6th	Southern-Europe	55
Masters	Central-Europe	30
Senior-Secondary	Western-Europe	40
Some-college	Western-Europe	42
Bachelors	Central-Europe	40

Cube7_4

ed_level1	nc_level1	hours_per_week
Bachelors	Central-Europe	40
Bachelors	Eastern-Europe	40
Bachelors	Southern-Europe	50
Bachelors	Western-Europe	40

Cube7_5

ed_level1	nc_level1	hours_per_week
Prof-school	Middle-America	60
Some-college	North-America	80
Senior-Secondary	South-America	72

Figure A.7 Cube scenario 7

Cube8

ed_level1	wc_level1	salary
Assoc-voc	Gov	>50K
7th-8th	Private	>50K
Masters	Private	>50K
Senior-Secondary	Self-emp	>50K
Some-college	Private	>50K
Bachelors	Private	>50K

Cube8_1

ed_level1	wc_level1	salary
Assoc-voc	Gov	>50K
7th-8th	Private	>50K
Masters	Private	>50K
Senior-Secondary	Self-emp	>50K
Some-college	Private	>50K

Cube8_2

ed_level1	wc_level1	salary
Assoc-acdm	Private	<=50K
7th-8th	Private	<=50K
Masters	Private	<=50K
Preschool	Private	<=50K
Senior-Secondary	Private	<=50K
Some-college	Private	>50K
Bachelors	Private	<=50K

Cube8_3

ed_level1	wc_level1	salary
Assoc-acdm	Private	<=50K
7th-8th	Private	<=50K
Masters	Private	<=50K
Preschool	Private	<=50K
Senior-Secondary	Private	<=50K
Some-college	Private	<=50K
Bachelors	Private	<=50K

Cube8_4

ed_level1	wc_level1	salary
Assoc-acdm	Private	>50K
7th-8th	Private	>50K
Masters	Private	>50K
Senior-Secondary	Private	>50K
Some-college	Private	>50K
Bachelors	Private	>50K

Cube8_5

ed_level1	wc_level1	salary
Assoc-voc	Gov	>50K
5th-6th	Gov	>50K
Doctorate	Gov	>50K
Senior-Secondary	Gov	>50K
Some-college	Gov	>50K
Bachelors	Gov	>50K

Cube8_6

ed_level1	wc_level1	salary
Bachelors	Gov	>50K
Bachelors	Gov	>50K
Masters	Private	>50K
Some-college	Self-emp	>50K
Senior-Secondary	Self-emp	>50K
Bachelors	Gov	>50K

Cube8_7

ed_level2	wc_level1	salary
Assoc	Gov	>50K
Elementary	Private	>50K
Post-grad	Private	>50K
Secondary	Self-emp	>50K
Some-college	Private	>50K
University	Private	>50K

Figure A.8 Cube scenario 8

Cube9

ag_level0	salary	hours_per_week
28	<=50K	40
30	<=50K	40
32	<=50K	55
32	<=50K	40
28	<=50K	50
27	<=50K	35
29	>50K	50
33	<=50K	45
29	<=50K	40
35	>50K	40

Cube9_1

ag_level0	salary	hours_per_week
34	<=50K	40
35	<=50K	35
32	<=50K	40
35	<=50K	40
34	>50K	40
35	<=50K	60
33	<=50K	40
34	<=50K	60
34	>50K	35
33	<=50K	35
36	<=50K	40

Cube9_3

ag_level0	salary	hours_per_week
36	<=50K	40
33	<=50K	55
35	>50K	50
32	<=50K	55
32	<=50K	25
32	<=50K	40
35	<=50K	55
33	<=50K	45
35	>50K	40

Cube9_2

ag_level0	salary	hours_per_week
28	<=50K	40
30	<=50K	40
27	>50K	65
28	<=50K	50
27	<=50K	35
29	>50K	50
31	<=50K	30
29	<=50K	40
31	<=50K	40

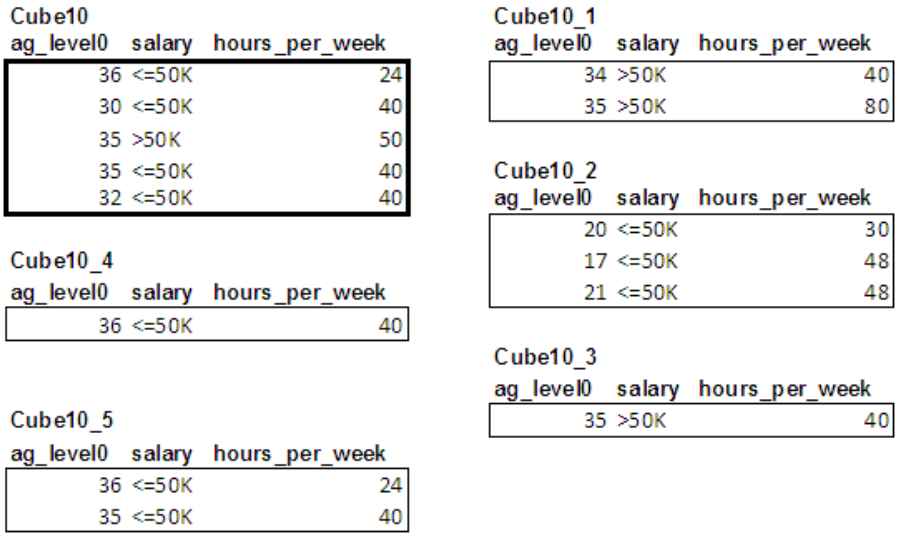
Cube9_5

ag_level0	salary	hours_per_week
27	>50K	40
31	<=50K	60
30	>50K	40
30	<=50K	50
29	<=50K	40
30	<=50K	40
27	<=50K	40
30	>50K	50

Cube9_4

ag_level0	salary	hours_per_week
28	<=50K	40
27	<=50K	35
29	<=50K	40

Figure A.9 Cube scenario 9



Cube10

ag_level0	salary	hours_per_week
36	<=50K	24
30	<=50K	40
35	>50K	50
35	<=50K	40
32	<=50K	40

Cube10_1

ag_level0	salary	hours_per_week
34	>50K	40
35	>50K	80

Cube10_2

ag_level0	salary	hours_per_week
20	<=50K	30
17	<=50K	48
21	<=50K	48

Cube10_3

ag_level0	salary	hours_per_week
35	>50K	40

Cube10_4

ag_level0	salary	hours_per_week
36	<=50K	40

Cube10_5

ag_level0	salary	hours_per_week
36	<=50K	24
35	<=50K	40

Figure A.10 Cube scenario 10

Cube11

ag_level1	salary	hours_per_week
27-31	<=50K	40
32-36	<=50K	55
32-36	<=50K	40
27-31	<=50K	50
27-31	<=50K	35
27-31	>50K	50
32-36	<=50K	45
32-36	>50K	40

Cube11_1

ag_level1	salary	hours_per_week
32-36	<=50K	35
32-36	>50K	40
32-36	<=50K	40
32-36	<=50K	60
32-36	>50K	35

Cube11_2

ag_level1	salary	hours_per_week
32-36	<=50K	40
32-36	>50K	50
32-36	<=50K	25
32-36	<=50K	55
32-36	<=50K	45
32-36	>50K	40

Cube11_3

ag_level1	salary	hours_per_week
32-36	<=50K	35
32-36	>50K	40
32-36	<=50K	40
32-36	<=50K	60
32-36	>50K	35

Cube11_4

ag_level1	salary	hours_per_week
17-21	<=50K	12
17-21	<=50K	35
17-21	<=50K	30
17-21	<=50K	20
17-21	<=50K	40

Cube11_5

ag_level1	salary	hours_per_week
27-31	<=50K	40
27-31	>50K	65
27-31	<=50K	50
27-31	<=50K	35
27-31	>50K	50
27-31	<=50K	30

Cube11_6

ag_level1	salary	hours_per_week
27-31	<=50K	40
27-31	<=50K	35

Cube11_7

ag_level1	salary	hours_per_week
32-36	<=50K	40
32-36	<=50K	35
32-36	<=50K	45

Cube11_8

ag_level1	salary	hours_per_week
32-36	>50K	40

Cube11_9

ag_level1	salary	hours_per_week
32-36	<=50K	46
32-36	<=50K	25
32-36	>50K	40
32-36	<=50K	40

Figure A.11 Cube scenario 11

Cube12

ag_level1	salary	hours_per_week
32-36	<=50K	24
27-31	<=50K	40
32-36	>50K	50
32-36	<=50K	40

Cube12_1

ag_level1	salary	hours_per_week
32-36	>50K	40
32-36	>50K	80

Cube12_2

ag_level1	salary	hours_per_week
32-36	<=50K	40

Cube12_3

ag_level1	salary	hours_per_week
27-31	<=50K	40

Cube12_4

ag_level1	salary	hours_per_week
32-36	>50K	40

Cube12_5

ag_level1	salary	hours_per_week
27-31	<=50K	43
27-31	<=50K	35
27-31	<=50K	40

Cube12_6

ag_level1	salary	hours_per_week
32-36	<=50K	24
32-36	<=50K	40

Figure A.12 Cube scenario 12

Cube13

wc_level1	ag_level1
Gov	27-31
Private	27-31
Self-emp	27-31
Without-pay	27-31

Cube13_1

wc_level1	ag_level1
Gov	22-26
Private	22-26
Self-emp	22-26
Without-pay	22-26

Cube13_2

wc_level0	ag_level1
State-gov	27-31
Private	27-31
Self-emp-not	27-31
Without-pay	27-31

Cube13_3

wc_level1	ag_level1
Private	27-31
Without-pay	27-31

Cube13_4

wc_level1	ag_level1
Gov	27-31
Gov	27-31
Private	27-31
Self-emp	27-31
Self-emp	27-31
Gov	27-31
Without-pay	27-31

Cube13_5

wc_level1	ag_level2
Gov	27-36
Private	27-36
Self-emp	27-36
Without-pay	27-36

Figure A.13 Cube scenario 13

Cube14

salary	hours_per_week	ag_level0
<=50K	24	36
<=50K	40	30
>50K	50	35
<=50K	40	35
<=50K	40	32

Cube14_1

salary	hours_per_week	ag_level0
>50K	40	34
>50K	80	35

Cube14_2

salary	hours_per_week	ag_level0
<=50K	30	20
<=50K	48	17
<=50K	48	21

Cube14_3

salary	hours_per_week	ag_level0
>50K	40	35

Cube14_4

salary	hours_per_week	ag_level0
<=50K	40	36

Cube14_5

salary	hours_per_week	ag_level0
<=50K	24	36
<=50K	40	35

Figure A.14 Cube scenario 14

Scenarios of the 2nd user study

Scenario 1

A

Age (level1)	Work Class (level1)	Race (level1)
52-56	Gov	White
52-56	Private	Colored
47-51	Self-emp	White
52-56	Without-pay	White

B

Age (level1)	Work Class (level1)	Race (level1)
27-31	Gov	Colored
52-56	Private	Colored
47-51	Self-emp	White
52-56	Without-pay	White

C

Age (level2)	Work Class (level2)	Race (level1)
47-56	With-Pay	Colored
47-56	With-Pay	White
47-56	Without-pay	White

D

Age (level1)	Work Class (level1)	Race (level1)
47-51	Self-emp	White
52-56	Without-pay	White

Figure A.15 Scenario 1of the 2nd user study

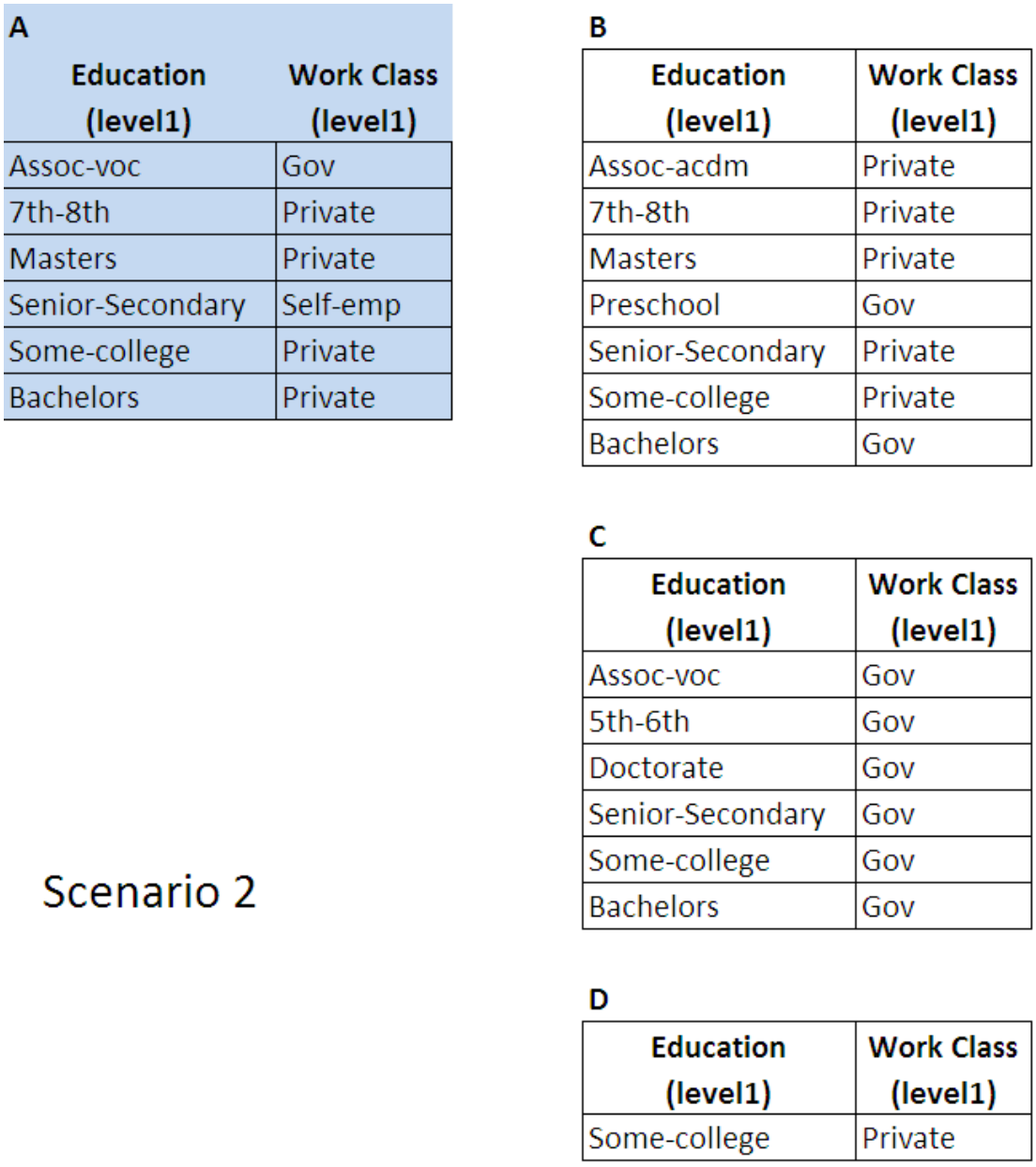


Figure A.16 Scenario 2 of the 2nd user study

A

Education (level1)	Work Class (level1)	Marital Status (level1)
Assoc-acdm	Private	Never-married
7th-8th	Private	Partner-present
Masters	Private	Partner-present
Preschool	Private	Never-married
Senior-Secondary	Private	Partner-absent
Some-college	Private	Partner-present
Bachelors	Gov	Never-married

B

Education (level2)	Work Class (level1)	Marital Status (level1)
Assoc	Private	Never-married
Elementary	Private	Partner-present
Post-grad	Private	Partner-present
Preschool	Gov	Never-married
Secondary	Private	Partner-absent
Some-college	Private	Partner-present
University	Gov	Never-married

C

Education (level1)	Work Class (level1)	Marital Status (level1)
Bachelors	Gov	Partner-absent
Senior-Secondary	Gov	Partner-present
Bachelors	Gov	Partner-present
Some-college	Gov	Partner-absent
Bachelors	Gov	Never-married
Senior-Secondary	Gov	Partner-absent
Masters	Gov	Partner-absent

Scenario 3

D

Education (level2)	Work Class (level2)	Marital Status (level1)
Assoc	With-Pay	Never-married
Elementary	With-Pay	Partner-present
Post-grad	With-Pay	Partner-present
Preschool	With-Pay	Never-married
Secondary	With-Pay	Partner-absent
Some-college	With-Pay	Partner-present
University	With-Pay	Never-married

Figure A.17 Scenario 3 of the 2nd user study

Scenario 4

A

Education (level1)	Native Country (level1)
Senior-Secondary	Central-Europe
Bachelors	Eastern-Europe
Senior-Secondary	Southern-Europe
Senior-Secondary	Western-Europe

B

Education (level1)	Native Country (level1)
Masters	Eastern-Asia
Masters	Middle-East
Bachelors	Southeastern-Asia
Bachelors	Southern-Asia

C

Education (level1)	Native Country (level1)
Bachelors	Central-Europe
Senior-Secondary	Eastern-Europe
Junior-Secondary	Southern-Europe
Assoc-acdm	Western-Europe

D

Education (level1)	Native Country (level2)
Bachelors	Europe
Senior-Secondary	Europe
Junior-Secondary	Europe
Assoc-acdm	Europe

Figure A.18 Scenario 4 of the 2nd user study

Scenario 5

A

Education (level1)	Native Country (level1)	AVG hours_per_week
Senior-Secondary	Central-Europe	60.8636
Bachelors	Eastern-Europe	60.75
Senior-Secondary	Southern-Europe	64.8095
Senior-Secondary	Western-Europe	63.5652

B

Education (level1)	Native Country (level1)	AVG hours_per_week
Masters	Eastern-Asia	41.9768
Masters	Middle-East	44.0714
Bachelors	Southeastern-Asia	39.8717
Bachelors	Southern-Asia	41.53

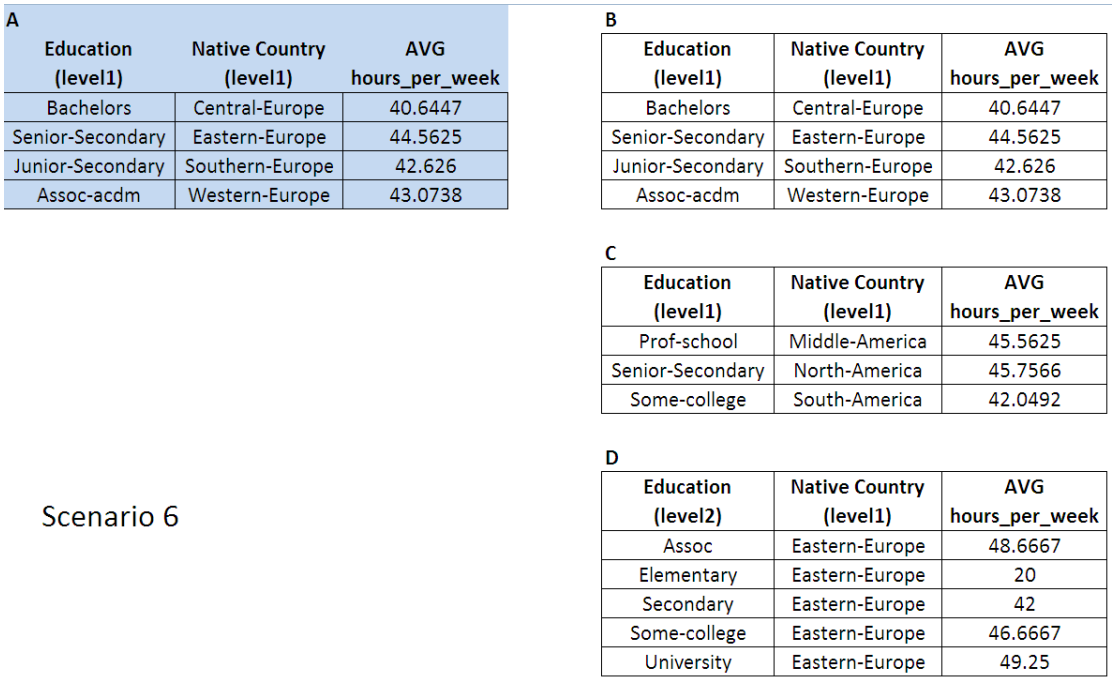
C

Education (level1)	Native Country (level1)	AVG hours_per_week
Bachelors	Central-Europe	40.6447
Senior-Secondary	Eastern-Europe	44.5625
Junior-Secondary	Southern-Europe	42.626
Assoc-acdm	Western-Europe	43.0738

D

Education (level1)	Native Country (level2)	AVG hours_per_week
Bachelors	Europe	40.6447
Senior-Secondary	Europe	44.5625
Junior-Secondary	Europe	42.626
Assoc-acdm	Europe	43.0738

Figure A.19 Scenario 5 of the 2nd user study



A

Education (level1)	Native Country (level1)	AVG hours_per_week
Bachelors	Central-Europe	40.6447
Senior-Secondary	Eastern-Europe	44.5625
Junior-Secondary	Southern-Europe	42.626
Assoc-acdm	Western-Europe	43.0738

B

Education (level1)	Native Country (level1)	AVG hours_per_week
Bachelors	Central-Europe	40.6447
Senior-Secondary	Eastern-Europe	44.5625
Junior-Secondary	Southern-Europe	42.626
Assoc-acdm	Western-Europe	43.0738

C

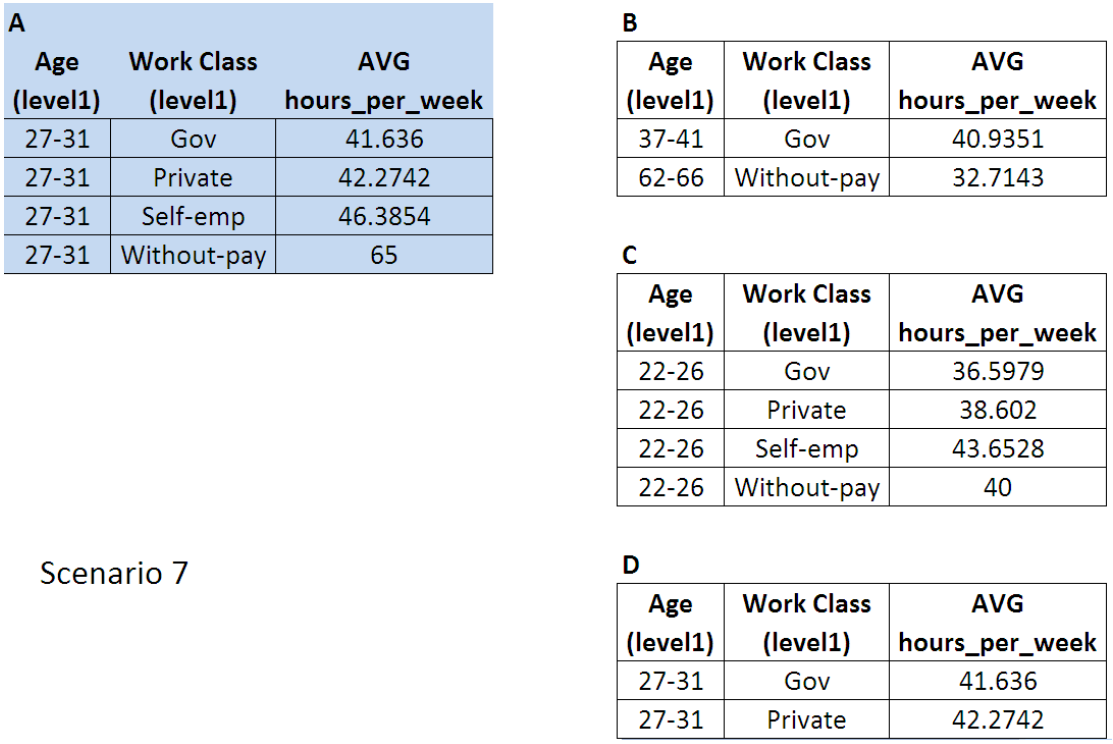
Education (level1)	Native Country (level1)	AVG hours_per_week
Prof-school	Middle-America	45.5625
Senior-Secondary	North-America	45.7566
Some-college	South-America	42.0492

D

Education (level2)	Native Country (level1)	AVG hours_per_week
Assoc	Eastern-Europe	48.6667
Elementary	Eastern-Europe	20
Secondary	Eastern-Europe	42
Some-college	Eastern-Europe	46.6667
University	Eastern-Europe	49.25

Scenario 6

Figure A.20 Scenario 6 of the 2nd user study



A

Age (level1)	Work Class (level1)	AVG hours_per_week
27-31	Gov	41.636
27-31	Private	42.2742
27-31	Self-emp	46.3854
27-31	Without-pay	65

B

Age (level1)	Work Class (level1)	AVG hours_per_week
37-41	Gov	40.9351
62-66	Without-pay	32.7143

C

Age (level1)	Work Class (level1)	AVG hours_per_week
22-26	Gov	36.5979
22-26	Private	38.602
22-26	Self-emp	43.6528
22-26	Without-pay	40

D

Age (level1)	Work Class (level1)	AVG hours_per_week
27-31	Gov	41.636
27-31	Private	42.2742

Scenario 7

Figure A.21 Scenario 7 of the 2nd user study

A

Age (level2)	Education (level2)	Occupation (level0)	AVG hours_per_wee
37-46	Assoc	Craft-repair	42.4773
37-46	Elementary	Machine-op-inspct	41.0847
37-46	Post-grad	Exec-managerial	45.5272
37-46	Preschool	Machine-op-inspct	35.7778
37-46	Secondary	Handlers-cleaners	42.3831
37-46	Some-college	Exec-managerial	43.6332
37-46	University	Adm-clerical	45.3889

B

Age (level2)	Education (level2)	Occupation (level0)	AVG hours_per_wee
27-36	Assoc	Sales	43.143
27-36	Elementary	Transport-moving	40.7958
27-36	Post-grad	Prof-specialty	44.4815
27-36	Preschool	Other-service	38
27-36	Secondary	Machine-op-inspct	42.4468
27-36	Some-college	Adm-clerical	42.5865
27-36	University	Prof-specialty	44.3959

C

Age (level2)	Education (level2)	Occupation (level0)	AVG hours_per_wee
47-56	Assoc	Prof-specialty	42.205
37-46	Elementary	Machine-op-inspct	40.3118
37-46	Post-grad	Exec-managerial	45.7161
47-56	Preschool	Machine-op-inspct	38.5
37-46	Secondary	Handlers-cleaners	41.652
37-46	Some-college	Exec-managerial	42.9059
37-46	University	Adm-clerical	44.7706

D

Age (level2)	Education (level2)	Occupation (level0)	AVG hours_per_wee
37-46	University	Adm-clerical	39.9895
37-46	Secondary	Armed-Forces	45
37-46	Assoc	Craft-repair	43.1377
37-46	Post-grad	Exec-managerial	46.4003
37-46	Secondary	Farming-fishing	50.8405
37-46	Secondary	Handlers-cleaners	41.3594
37-46	Elementary	Machine-op-inspct	41.2668
37-46	Secondary	Other-service	38.6933
37-46	Secondary	Priv-house-serv	30.7727
37-46	Post-grad	Prof-specialty	43.8034
37-46	Post-grad	Protective-serv	45.549
37-46	Secondary	Sales	46.3443
37-46	Some-college	Tech-support	40.9124
37-46	Secondary	Transport-moving	45.5857

Scenario 8

Figure A.22 Scenario 8 of the 2nd user study

A

Age (level1)	Work Class (level1)	Race (level1)	AVG hours_per_week
47-56	Gov	White	42.3757
47-56	Private	Colored	42.3026
47-56	Self-emp	White	47.9392
47-56	Without-pay	White	30

B

Age (level1)	Work Class (level1)	Race (level0)	AVG hours_per_week
32-36	Gov	Black	39.4125
52-56	Private	Black	38.241
67-71	Self-emp	Black	42.8727
17-21	Without-pay	Black	40

C

Age (level1)	Work Class (level1)	Race (level1)	AVG hours_per_week
57-61	Gov	Colored	39.5879
57-61	Private	White	39.3317
57-61	Self-emp	White	41.8661
62-66	Without-pay	White	34

Scenario 9

D

Age (level1)	Work Class (level1)	Race (level1)	AVG hours_per_week
47-51	Self-emp	White	43.5387
52-56	Private	Colored	42.8894

Figure A.23 Scenario 9 of the 2nd user study

A

Age (level1)	Work Class (level1)	AVG hours_per_week
27-31	Gov	41.636
27-31	Private	42.2742
27-31	Self-emp	46.3854
27-31	Without-pay	65

B

Age (level1)	Work Class (level1)	AVG hours_per_week
37-41	Private	40.2509
62-66	Without-pay	32.7143

C

Age (level1)	Work Class (level1)	AVG hours_per_week
22-26	Gov	36.5979
22-26	Private	38.602
22-26	Self-emp	43.6528
22-26	Without-pay	40

Scenario 10

D

Age (level1)	Work Class (level1)	AVG hours_per_week
27-31	Gov	41.636
32-36	Private	42.8008

Figure A.24 Scenario 10 of the 2nd user study

A

Native Country (level1)	Education (level2)	Occupation (level1)	AVG hours_per_week
Western-Europe	Assoc	white-collar	42
Central-Europe	Elementary	Blue-collar	39.0625
Southern-Asia	Post-grad	white-collar	42.2688
Southern-Asia	Preschool	Blue-collar	40.5714
Western-Europe	Secondary	white-collar	40.9172
Southern-Asia	Some-college	Blue-collar	39.6615
Southern-Asia	University	white-collar	43.0597

B

Native Country (level1)	Education (level2)	Occupation (level1)	AVG hours_per_week
Western-Europe	Assoc	white-collar	41.9388
Central-Europe	Elementary	Blue-collar	40.4
Western-Europe	Post-grad	white-collar	45.5106
Western-Europe	Secondary	white-collar	41.494
Central-Europe	Some-college	white-collar	40.1772
Central-Europe	University	white-collar	43.3036

C

Native Country (level1)	Education (level2)	Occupation (level1)	AVG hours_per_week
North-America	Assoc	white-collar	41.6362
North-America	Elementary	Other	39.4748
North-America	Post-grad	white-collar	44.9085
North-America	Preschool	Blue-collar	36.8667
North-America	Secondary	Blue-collar	39.9944
North-America	Some-college	white-collar	39.4113

Scenario 11

D

Native Country (level1)	Education (level2)	Occupation (level1)	AVG hours_per_week
Western-Europe	Secondary	Blue-collar	41.3601
Southern-Asia	Secondary	Other	39.9945
Southern-Asia	University	white-collar	41.9779

Figure A.25 Scenario 11 of the 2nd user study

A

Education level1	Work Class level1	AVG hours_per_week)
Assoc-voc	Gov	44.2217
7th-8th	Private	47.3962
Masters	Private	46.672
Senior-Secondary	Self-emp	45.1358
Some-college	Private	45.0569
Bachelors	Private	46.3124

B

Education level1	Work Class level1	AVG hours_per_week
Assoc-acdm	Private	40.7085
7th-8th	Private	39.035
Masters	Private	42.2935
Preschool	Gov	36.8667
Senior-Secondary	Private	39.1367
Some-college	Private	37.9991
Bachelors	Gov	41.0494

C

Education level1	Work Class level1	AVG hours_per_week
Bachelors	Gov	43.3342
Bachelors	Gov	44.0033
Masters	Private	45.4934
Some-college	Self-emp	50.2533
Senior-Secondary	Self-emp	46.7451
Bachelors	Gov	44.1744

Scenario 12

D

Education level1	Work Class level1	AVG hours_per_week
Some-college	Gov	45.0569

Figure A.26 Scenario 12 of the 2nd user study

A

Native Country level1	Education level1	AVG hours_per_week
Central-Europe	Senior-Secondary	60.8636
Eastern-Europe	Bachelors	60.75
Southern-Europe	Senior-Secondary	64.8095
Western-Europe	Senior-Secondary	63.5652

B

Native Country level1	Education level1	AVG hours_per_week
Eastern-Asia	Masters	41.9768
Middle-East	Masters	44.0714
Southeastern-Asia	Bachelors	39.8717
Southern-Asia	Bachelors	41.53

C

Native Country level1	Education level1	AVG hours_per_week
Central-Europe	Bachelors	40.6447
Eastern-Europe	Senior-Secondary	44.5625
Southern-Europe	Junior-Secondary	42.626
Western-Europe	Assoc-acdm	43.0738

D

Native Country level2	Education level1	AVG hours_per_week
Europe	Bachelors	40.6447
Europe	Senior-Secondary	44.5625
Europe	Junior-Secondary	42.626
Europe	Assoc-acdm	43.0738

Scenario 13

Figure A.27 Scenario 13 of the 2nd user study

A

Work Class level1	Race level1	Age level2	AVG hours_per_week
Gov	White	47-56	42.3757
Private	Colored	47-56	42.3026
Self-emp	White	47-56	47.9392
Without-pay	White	47-56	30

B

Work Class level1	Race level0	Age level1	AVG hours_per_week
Gov	Black	32-36	39.4125
Private	Black	52-56	38.241
Self-emp	Black	67-71	42.8727
Without-pay	Black	17-21	40

C

Work Class level1	Race level1	Age level1	AVG hours_per_week
Gov	Colored	57-61	39.5879
Private	White	57-61	39.3317
Self-emp	White	57-61	41.8661
Without-pay	White	62-66	34

D

Work Class level1	Race level1	Age level1	AVG hours_per_week
Self-emp	White	47-51	43.5387
Private	Colored	52-56	42.8894

Scenario 14

Figure A.28 Scenario 14 of the 2nd user study

SHORT CV

Giorgos Rogkakos was born in Ioannina in 1986. He received his BSc degree in 2008 from the Department of Computer Science of the University of Ioannina in Greece. Then, he entered the Graduate Program of the same institution under the supervision of Panos Vassiliadis and he was a member of the Distributed Management of Data (DMOD) laboratory. He has been also an instructor in the Epirus Institute of Technology teaching the course of Data Mining. So far, his research is based on On-Line Analytical Processing (OLAP) tools under the context of comparing and exploring the similarity between OLAP Cubes.