CUBE MAZE

BY PRITISH SAHU

A thesis submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Master of Science

Graduate Program in Computer Science

Written under the direction of

Dr. James Abello

and approved by

New Brunswick, New Jersey

May, 2017

ABSTRACT OF THE THESIS

Cube Maze

by Pritish Sahu

Thesis Director: Dr. James Abello

Conventional data visualization methods are very narrow in terms of the data types on which they are applicable. We present a novel way of viewing multi-attributed dataset by grouping subsets of attributes into facets. Our Cube-Maze interface visually represents each data "entity" as a cube in three dimensional space. Similarity among "data cubes" correspond to 0, 1, and 2 dimensional adjacencies. Our current implementation provides different modes of "EgoNet" navigation and several interaction filters. The graph counterpart for this cube maze representation is a "Labeled Multi Digraph".

Keywords: Multi faceted data visualisation; Egonet navigation; Multi Digraph

Acknowledgements

I would like to sincerely thank my advisor, Dr. James Abello for continued guidance and support during this research and to Dr. Tomasz Imielinski for serving as the committee chair.

I am also grateful to Dr. Mubbasir Kapadia and Dr. Abdeslam Boularias for being a part of my defense committee and sharing their valuable knowledge and insights on my work.

I would also like to thank the Computer Science Department at Rutgers for economic assistance.

Lastly, I would like to thank my family and friends who have always been supportive of me and have made it possible for me to follow my passion.

Thanks for all your encouragement!

Dedication

I would like to dedicate this thesis to my parents.

Table of Contents

List of Figures

Chapter 1

Introduction

In many fields, visualisation has gained recognition as potential tool to explore and understand data. Interaction with the data plays an important role in information visualisation ([1],[2], [3], [4], [5]) . We develop an approach to visualise multi-attributed data in a unified way. Various interaction mechanisms are implemented to explore this data based on the semantic closeness between attributes of data members. We introduce **Cube Maze** as the main visual representation tool to display multi-attributed data in a consolidated manner. We view multi-attributed data as vectors of components called data units. Grouping these k-components allows mappings of each group onto each side of a cube's inner wall.

Objectives of the Research

We provide a novel interface(**Cube Maze**), various navigation modes, and associated interaction tools. In this section, we discuss related challenges and give an overview of how our system tackles the aforementioned challenges. The major challenge consists of visualising multi-attribute data sets. Current exploration systems become so complex that a user loses track of the path used to dive deep into the represented data. For each feature described in the system, the canvas gets redrawn without containing the path information traversed by the user. Previous work has been published based on multiple view approaches for a data containing vector of attributes. These works use multiple coordinate views ([6], [7],[8]]). One major challenge in today's world is data analysis by exploration. For example, exploring and evaluating in a small time frame a university data consisting of departments which consists of faculty members, students, administrators and associated infrastructure. Similarly, data attributes for faculties contain biography, publications, projects, grants, awards, students advised, courses taught. Also, consolidating data in one single visual representation and exploring the data is a complex task, even without factoring in the labor of finding relations between various data units such as faculty members or departments based on certain attributes. Current systems in place don't have a mechanism to support such endeavors, i.e., explore data where all its facets are pooled in one view, find relationships and navigate to data which share similar attributes. In the same view, consider the scenario for movies, health, sports, etc. There are no systems currently available which provide all required information in one place and also provide relationships between several of these entities based on their attributes.

Our work focuses on overcoming the aforementioned issues in designing a novel interface system for multi-medial data which provides global and local contextual information with easy navigation mechanisms. The basic data unit we use to visualise data is a three dimensional cube. Various navigation mechanisms are introduced to demonstrate hierarchy navigation from higher rank order to lower rank order units or to navigate between cubes based on semantic relationships. These semantic relationships are based on data facets similarity and the system allows to a user to teleport from a selected cube to another cube in global space based on EgoNet similarity between the cubes.

Our goal is to enhance a user's experience by creating a simple novel and intuitive interface that helps overcome the challenges mentioned above. In summary, the highlighting points in our design are :

- visualisation all attributes of a data unit in one local visual space.
- enhanced data exploration via novel interaction mechanisms.
- visual display of semantic relationships between data based on their attributes.
- furnishing a visual global context for local view.

In the following chapters, we present about the data model, its visual representation, implementation specifics, summary of findings and future work.

Chapter 2

Data Units Multi-DiGraph Model

2.1 Data Model

We view a multimedial data collection as a “labeled multi-digraph” over a finite set of ranked “data units”[9]. Each data unit is an ordered set of data components each of which posses an identifier, a string name, an optional type or data rank, and a collection of links to possible components satellite data or to other data units.

As an example, a university may be viewed as a collection of data units each focused on a particular set of fields of knowledge, sports or athletics (Rank 1). Each such data unit in turn consists of a collection of departments (Rank 2) each of which can be also considered a data unit with the following set of components:

Dept = < Dept_id, Type = Academic, Rank = 2, Name = Computer Science, Fac = set of faculty, Co = set of courses offered, Stu = set of registered students, Admin = set of administrators, Digital_Infrastructure = Digital_equipment, Physical_Infrastructure = set of office buildings, ImageArchive = set of Images , VideoArchive= set of Videos, AudioArchive = set of Audio Recordings, WebArchive = set of Web Sites >

A Faculty member can be considered also as a data unit of Rank = 3 with the following components:

Fac = < Fac_id, Type = Researcher, Rank = 3, Name = Pythagoras, Bio = Textual Description, Publications = set of Papers-Books, Stu = set of registered students advised, Pro = set of projects involved with, Co = courses taught, Awards = Honors-Distinctions, ImageArchive = set of Images , VideoArchive= set of Videos, AudioArchive = set of Audio Recordings, WebArchive = set of Web Sites >

Two data units (like Depts) of the same rank can be “ semantically related” via some of their components. These relations among data units can be explicitly annotated in the data or they can be used to derive or learn other implicit “latent relations.

In our running example, two departments can share Faculty members, Courses, Students, and Physical Infrastructure. To account for this diversity of relations among two data units we use the language of labeled multi-digraphs over a finite set of vertices. In our case, each vertex corresponds to a data unit, and between a pair of data units u and v we allow two types of multiple labeled edges as follows: Component wise similarity: The label of the i-th edge $(u, v)_i$ encodes a similarity between the two i-th components of u and v Data Unit similarity: the edge label encodes overall data unit similarity rather than component wise similarity.

The notion of rank is intended to capture the level of hierarchical containment among data units. As an example, a university school is composed of departments whose basic constituents are faculty members, students, administrators, courses, projects, etc. Whenever these hierarchical relations among data units are explicitly present in the data we model them by an Explicit Directed Acyclic Graph that we call EDAG. The rank of a data unit in this EDAG is the length of its longest path to the set of EDAG source data units.

After all these preliminaries we are now ready to introduce our Data Units Multi-Graph Model.

Concept 1. A data unit D $= < C_1, C_2, .., C_k >$ is an ordered set of k components C_i: i=1, 2, ... k. We denote by #(D) the number of components of a data unit D. Each component C_i has a descriptor that consists of a unique id, an integer rank value, a string name, a type, and an optional collection of links to data units of greater rank or to a sequence of data units of the same rank. As an example: since a video is a sequence of frames we can consider a frame a more atomic element than a video and therefore the rank of a frame is not higher than the frame of a video. We refer to these links as vertical data links.

Concept 2. Similarity between data units

Given two data units $D_i = < D_{i,1}, D_{i,2}, .., D_{i,k} >$ and $D_j = < D_{j,1}, D_{j,2}, .., D_{j,l} >$ of the same rank, their similarity can be formulated in terms of the similarity of the EDAGs of D_i and D_j when these EDAGs are explicitly provided with the data. When this is not the case, we propose to treat Data Units as simplicial complexes and obtain similarity measures among them derived from their peeling vectors([10],[11]). This approach has been successfully applied in Ref [12] to the unraveling of the inherent structure encoded by the peel distribution of any network. We propose to extend these mechanisms to the case of labeled directed multigraphs since in this case the neighborhood of each vertex can be seen as the simplicial complex formed by its colored adjacent edges. In this way, a plethora of similarity measures between two complex data units can be derived from the combinatorial structure of their corresponding simplicial complexes. This approach can be readily applied at the component wise level or at the entire data unit level.

Concept 3. Universal Labeled MultiGraph MUD = (V= $\{D_1, D_2, ..., D_n$ }, MS= $\{($ $D_i \sim D_j$ $)\}$)

Consider a collection of data units V = $D_1, D_2, ..., D_n$ and denote by MS the block matrix that on entry MS[i,j] contains the similarity matrix between the data components of D_i and D_j, i.e. the f,g entry of the similarity matrix is the similarity between $D_{i,f}$, the f-th data component of D_i, and $D_{j,g}$, the g-th data component of D_j. The pair (D_i, D_j) is labeled by the similarity matrix MS[i , j] . For future reference define Size(V) = SumOf{#(D_i): D_i in V } and max(V) = Max{#(D_i): D_i in V}.

We call the multi-digraph obtained in this manner, a MUD with n data units Di and data units pair wise similarity matrices MS[i,j]. Directed simple paths correspond to linear story lines.

In practice, only portions of this Multi-Graph are provided as input and in this proposals project we will build a multi-medial data exploration system that will provide users with mechanisms to interactively explore, annotate, derive, and synthesize the most "salient" features and directed paths that can be traced and interpreted when querying and navigating these multifaceted data sets. In other words, we want users to be able to "extract the stories behind the data" or "to create stories from the data that can be traced and verified by a computer aided agent". A natural question is then how to define what a "data story" is. We suggest a mechanism to achieve this in the next section.

Concept 4. Data Stories Some of the most basic elements of a narrative are "its characters" and their "interactions" through "time". Since in our data model the basic "characters" of a collection V are its "data units" and their multiple directed interactions are encoded by their similarity matrix, it makes sense to conceptually dismantle the overall Data Units MultiDigraph into its max(V) component wise digraphs. On each such digraph we can compute its strongly connected components macro-DAG. Linearization of each of these DAGs provides a partial order that can be interpreted as the "event" partial ordering of the data units "story plot" when restricted to a particular data units components entry type (the story projections). The length of the longest path on each such DAG can be interpreted as a data story completion time on that story projection. Composition of these story projections via shared vertices will allow us to identify overall data stories. This method can be seen as an adaptation of an approach proposed by [12] to analyze a variety of networks by decomposing their edge set into maximal subsets of edges that are fixed points with respect to iterative degree driven vertex deletion. The approach has been applied to citation networks and literature classics like "Les Miserables"[9] , and Danish Folklore [13]

In summary: any MUD = (V, S) whose edges are labeled by Max(V) labels can be naturally decomposed into Max(V) edge disjoint DAGs. Each of these DAGs provides a partial order on the involved data units. Composition operations among these DAGs provide mechanisms to build a variety of complementary MUD stories. A simple mechanism to achieve this is to take the union of all source data units across the DAG projections. Notice that since there may be connections among the sources in different DAGs it is natural to consider the subgraph induced among the different iterative story projection sources as important "events" in the Data story. Removing these sources iteratively give us a natural coarse story flow of "events" derived from the MUD connectivity. An appealing aspect of this multigraph -strongly connected approach is that the strongly connected components can be updated efficiently in a streaming fashion [14].

2.2 Advantages of the Data Model

The data unit definition in the model can fit any multi-attribute data set by grouping subsets of attributes into facets. This model can be used to analyse and explore components of each data unit via similarity measures between the different data facets (cube faces). Grouping of cubes via similarity measure can be achieved. A user can navigate between cubes via relationship computed between cube face. These navigation can be neighborhood navigation i.e. movement between neighboring cubes or teleport which is bring cube not necessarily a neighbor but out in the global space to the visible screen space.

Chapter 3

Visual Representation

3.1 Mapping Data Units to Three Dimensional Cubes

This novel interface represents each data unit as a cube in three dimensional space. A data unit may contain lower rank data units as component or can be an atomic data unit. The faces of the cube correspond to particular "facets" of the data. This mapping takes into account the number of components in the data unit and is based on grouping semantically coherent components. One facet is reserved for EgoNet navigation and the other facets are used to represent the remaining data components. A cube's EgoNet provides the user a global context for the cube's local space.

Higher rank data units contain components which in turn can be data units. For instance, a university department as a higher rank data unit contains faculty which can be considered a lower rank data unit. Hierarchy based navigation allows movement from higher ranked data units to lower ranked data units. As an example, considering department as a data unit of rank 2, it can be viewed as

Department = < name of the department, set of faculty, set of courses offered, set of registered students, set of administrators, digital equipment, set of office buildings, image archive, video archive, audio archive, web archive >

Faculty data can then be mapped cube faces with components:

Faculty = < name, biography, publications, projects, awards, courses taught, students advised >

3.2 Visualisation

The system is designed in a layered fashion, moving from a higher rank to a lower rank data unit. It also provides a clear analysis and mapping of the data on each layer without losing the global context.

3.2.1 Global View

The global space contains the set of data units of a particular rank. The data units are represented as cubes placed adjacent to each other in m rows and n columns [fig:3.1] in the canvas area. We followed responsive design guidelines to accommodate different screen sizes by adjusting the values of m, n and cube sizes. Each cube in the global space displays a summary of all its components. User actions influence the flow of data which is reflected by changes in several visual mappings. A detailed explanation of these visualisation is included next.

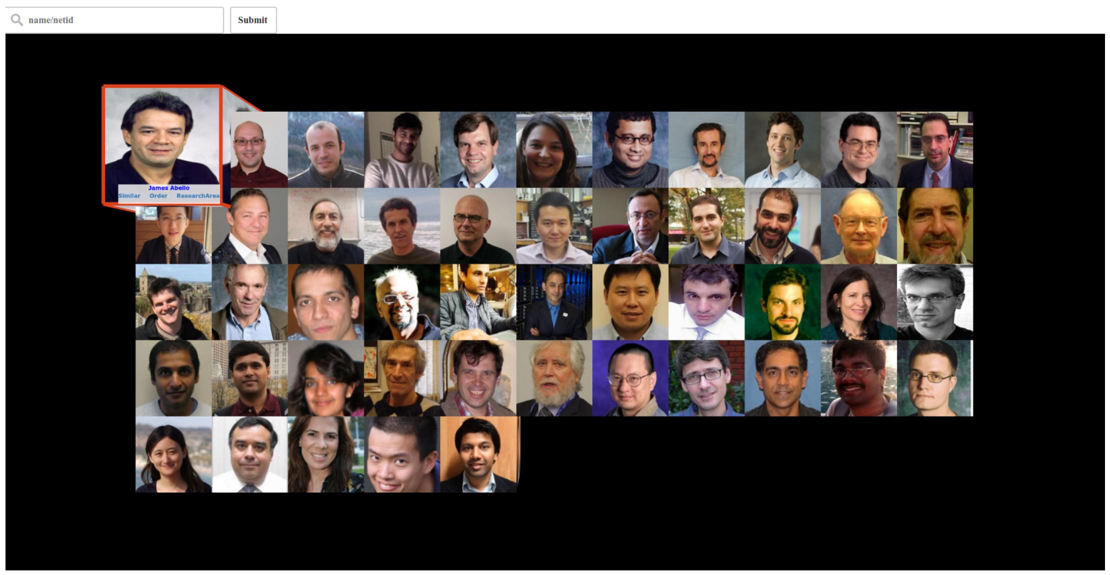
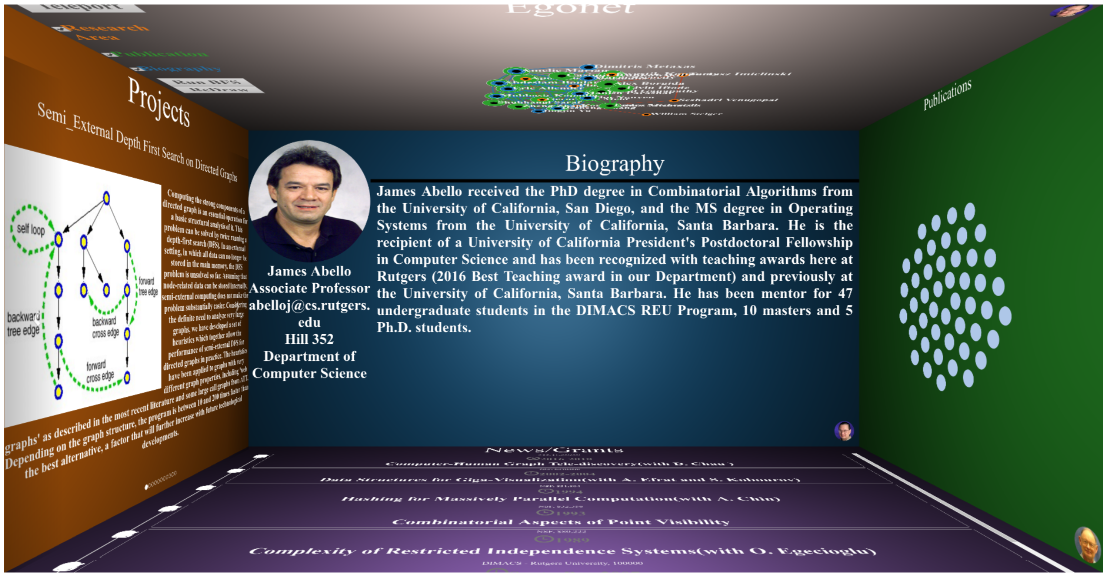


Figure 3.1: Global View

On the global space, the user can rotate each cube on its local axis in the horizontal and vertical directions to view all the cube face and the summarized information mapped to them. In our example, the summarized information on each face includes faculty image, keywords describing the research area, videos from a faculty video archive, slide show of image, office information, hobbies etc. To address the issue of visualising user data in a constrained space, we enlarge the face of the cube to cover the entire global space, thereby providing more canvas space in order to include all user information.

3.2.2 Local View

Clicking a cube triggers a swift transition from global space to local space depicting the cube's inner space. The transition involves zooming in action on the faculty's image until it occupies the entire screen or the entire canvas space. It splits in two halves with each half moving horizontally outwards revealing the local space of that cube. This gives the the user the impression that he/she has moved inside of that cube and now all the cube walls are visible except the one behind. A light object is placed in the center of the cube to allow the user to see all the cube faces from user's view point. The walls/faces of the cube are mapped to the components of the data units.



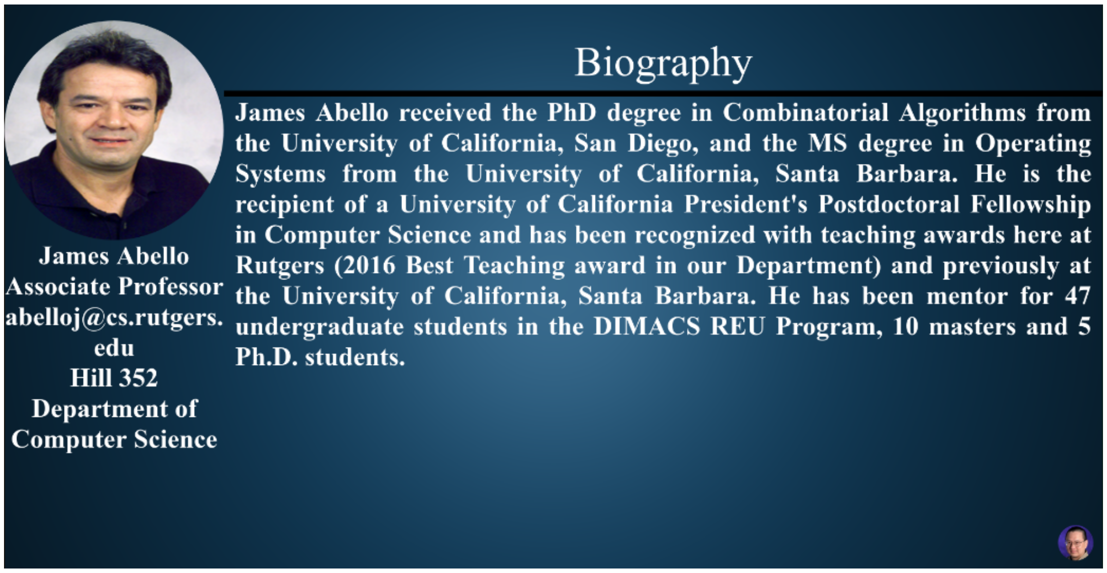
James Abello received the PhD degree in Combinatorial Algorithms from the University of California, San Diego, and the MS degree in Operating Systems from the University of California, Santa Barbara. He is the recipient of a University of California President's Postdoctoral Fellowship in Computer Science and has been recognized with teaching awards here at Rutgers (2016 Best Teaching award in our Department) and previously at the University of California, Santa Barbara. He has been mentor for 47 undergraduate students in the DIMACS REU Program, 10 masters and 5 Ph.D. students.

Computer-Human Graph Tele-discovery(with D. Chau)
2002-2004
Data Structures for Giga-Visualization(with A. Efrat and S. Kobourov)
1994
Hashing for Massively Parallel Computation(with A. Chin)
1993
Combinatorial Aspects of Point Visibility
NSF, $80,222
1989
Complexity of Restricted Independence Systems(with O. Egecioglu)
DIMACS - Rutgers University, 100000

Figure 3.2: Cube Face of a Faculty

For this example, the components, $< C_1, C_2, .., C_k >$ are grouped into 5 clusters as was discussed in section [fig:3.1] and the ceiling plane contains the EgoNet. In our example of faculty as data units, the facets display "Biography", "Publications", "Projects", "Grants/News", "Students Advised, Courses Taught" and the ceiling face contains the ego-centric network of the selected data unit representing the global context of the current local view. The standard layout of the cube is shown in [fig:3.2]

"Biography" contains textual information describing the selected faculty [fig:3.3]. It is placed by default at the center face of the cube i.e directly facing the camera's view. The layout of the face is divided into 3 sections. The first section on the top left contains the image and string name to which the cube belongs to, the second section describes the position, email address, department etc. positioned just below. These two sections take one quarter of the face. The rest of the space is utilised to display the faculty members biography. The information mentioned here is merely of descriptive type and is not interactive.



Biography

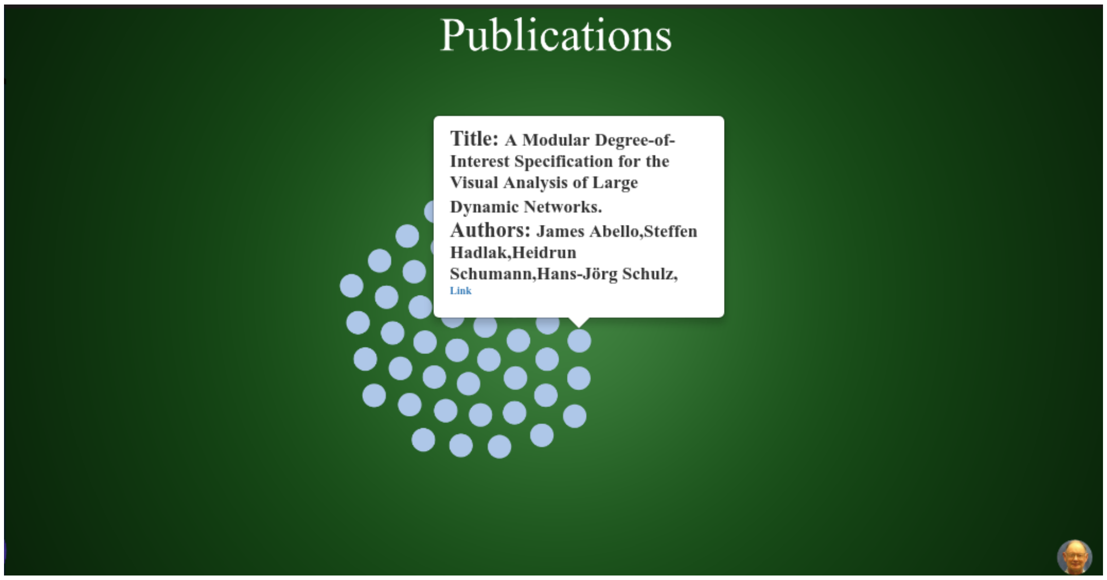
James Abello received the PhD degree in Combinatorial Algorithms from the University of California, San Diego, and the MS degree in Operating Systems from the University of California, Santa Barbara. He is the recipient of a University of California President's Postdoctoral Fellowship in Computer Science and has been recognized with teaching awards here at Rutgers (2016 Best Teaching award in our Department) and previously at the University of California, Santa Barbara. He has been mentor for 47 undergraduate students in the DIMACS REU Program, 10 masters and 5 Ph.D. students.

James Abello
Associate Professor
abelloj@cs.rutgers.
edu
Hill 352
Department of
Computer Science

Figure 3.3: Cube Face of Biography

"Publication" is the list of papers published by the faculty in various conference/journals [fig:3.4]. For a faculty data unit all the publications are displayed as nodes and this set of nodes are mapped on to the right face of the cube in the default orientation. Moving the mouse over each of the dot, a small display window appears containing "Title" and "Authors" for the corresponding paper. A User can also visit the selected paper by clicking on the dot which brings for view the web content of the page to a window attached to the plane.

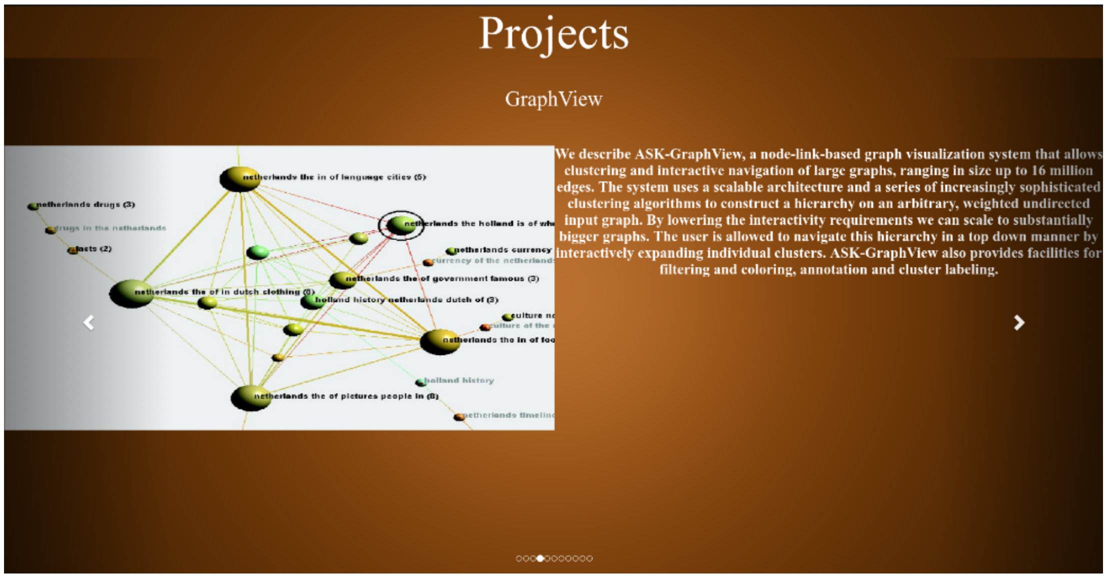
The "Projects" face contains the collection of scholarly projects conducted by the



Title: A Modular Degree-of-Interest Specification for the Visual Analysis of Large Dynamic Networks.
Authors: James Abello,Steffen Hadlak,Heidrun Schumann,Hans-Jörg Schulz,
Link

Figure 3.4: Cube Face of Publications

faculty [fig:3.5]. It is shown as an image on the left followed by a brief about the project on the right. The projects can be browsed sideways (left, right) to reveal more. This projects information is mapped to the left face of the cube.

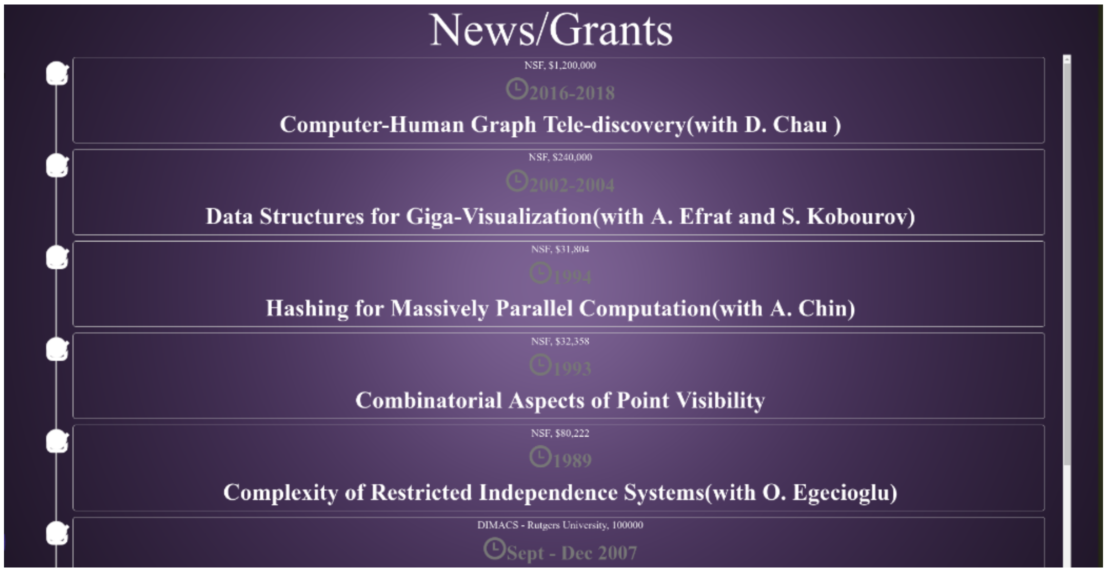


We describe ASK-GraphView, a node-link-based graph visualization system that allows clustering and interactive navigation of large graphs, ranging in size up to 16 million edges. The system uses a scalable architecture and a series of increasingly sophisticated clustering algorithms to construct a hierarchy on an arbitrary, weighted undirected input graph. By lowering the interactivity requirements we can scale to substantially bigger graphs. The user is allowed to navigate this hierarchy in a top down manner by interactively expanding individual clusters. ASK-GraphView also provides facilities for filtering and coloring, annotation and cluster labeling.

Figure 3.5: Cube Face of Projects

“Grants/Events” provides information in a time line about grants, awards, news published related to the faculty [fig:3.6]. “Grant Name”, “Research Title”, “Research Brief”, “Grant Money”, “Time Period” are the types of information mapped to this face located at the bottom of the cube.

"Students Advised and Courses Taught" are visually represented as a horizontal tree where the leaf nodes are the students and courses [fig:3.7]. The student information mapped contains information on name, current status, degree of study, place of work (if any). The other vertex contains courses taught containing course name, course description and webpage link (if any). This is mapped onto the face behind the user.


News/Grants

NSF, \$1,200,000
2016-2018
Computer-Human Graph Tele-discovery(with D. Chau)

NSF, \$240,000
2002-2004
Data Structures for Giga-Visualization(with A. Efrat and S. Kobourov)

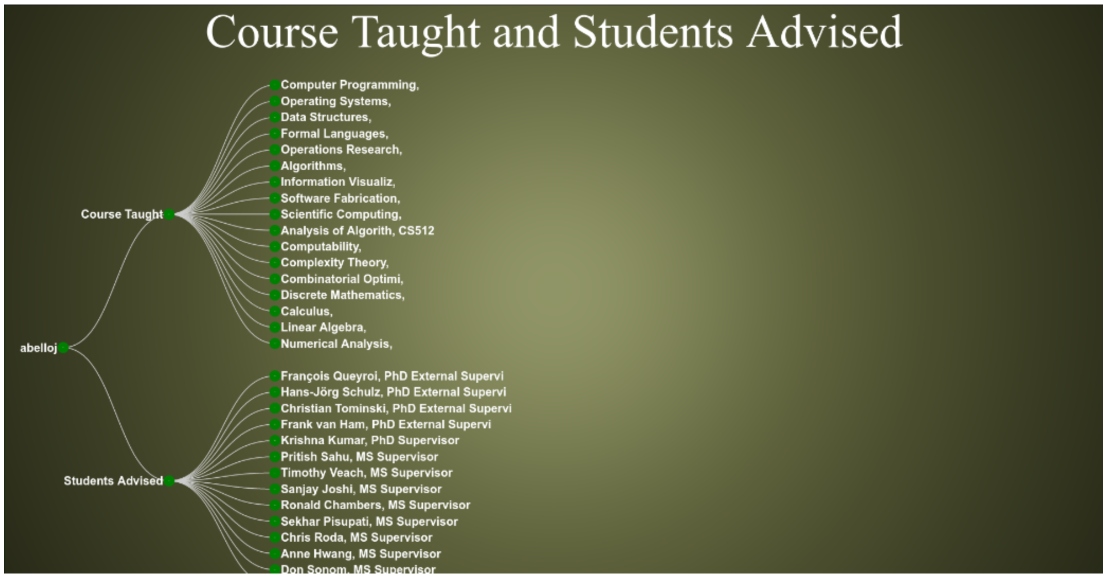
NSF, \$31,804
1994
Hashing for Massively Parallel Computation(with A. Chin)

NSF, \$32,358
1993
Combinatorial Aspects of Point Visibility

NSF, \$80,222
1989
Complexity of Restricted Independence Systems(with O. Egecioglu)

DIMACS - Rutgers University, 100000
Sept - Dec 2007

Figure 3.6: Cube Face of Events


Course Taught and Students Advised

abelloj
Course Taught
Computer Programming,
Operating Systems,
Data Structures,
Formal Languages,
Operations Research,
Algorithms,
Information Visualiz,
Software Fabrication,
Scientific Computing,
Analysis of Algorith, CS512
Computability,
Complexity Theory,
Combinatorial Optimi,
Discrete Mathematics,
Calculus,
Linear Algebra,
Numerical Analysis,

Students Advised
François Queyroi, PhD External Supervi
Hans-Jörg Schulz, PhD External Supervi
Christian Tominski, PhD External Supervi
Frank van Ham, PhD External Supervi
Krishna Kumar, PhD Supervisor
Pritish Sahu, MS Supervisor
Timothy Veach, MS Supervisor
Sanjay Joshi, MS Supervisor
Ronald Chambers, MS Supervisor
Sekhar Pisupati, MS Supervisor
Chris Roda, MS Supervisor
Anne Hwang, MS Supervisor
Don Sonom, MS Supervisor

Figure 3.7: Cube Face of Students Advised and Courses Taught

The ceiling wall contains the Ego-Centric Network ”EgoNet”[fig:3.9]. The EgoNet contains the global context information in local space. The subgraph induced by the neighborof a node X in a graph is known as its EgoNet [fig:3.8]. For a faculty, the EgoNet contains all the nodes which are its direct neighbours and the connections between them without the source node (X) and its edges connecting X to its neighbors. The resulting graph is the ego-centric local sub graph for X. We have mapped multi-medial data to this graph as a labeled (colored) multi-digraph. Nodes are connected based on the what type of similarity they share. In our example, there are K=3 types of similarities defined, which are relations based on “research area”,“publications” and “biography”. Each is mapped to a specific color. The nodes and edges are colored among these colors. A node can have K different colors including the color when the outgoing/incoming edges are nil and two nodes can share at most K edges if they share similarity on all K similarity types. The similarity weight ranges between 0 and 1, We have calculated the mean and standard deviation for each similarity and display on the EgoNet those edges whose similarity lies between the mean and the standard deviation.
Apart from the above components mapped on each face of the cube, icons are added

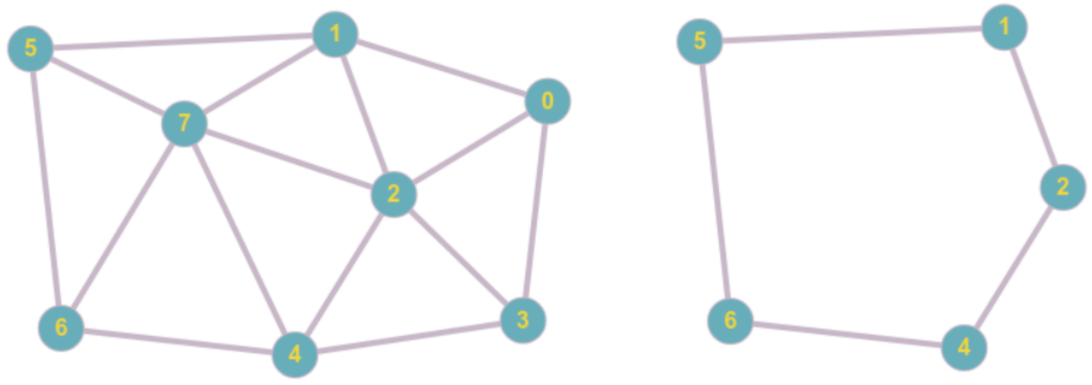


Figure 3.8: Sample EgoNet of Node Seven

at the bottom of each face to provide semantic relationships of the current selected cube to that icon’s type. Some of this semantic information can be neighborhood information or cubes related to the current one based on the face the icon is mapped to in our faculty data. These icons are faculty images mapped at special locations of each face. Clicking on these icons starts a visual transition that shows neighborhood

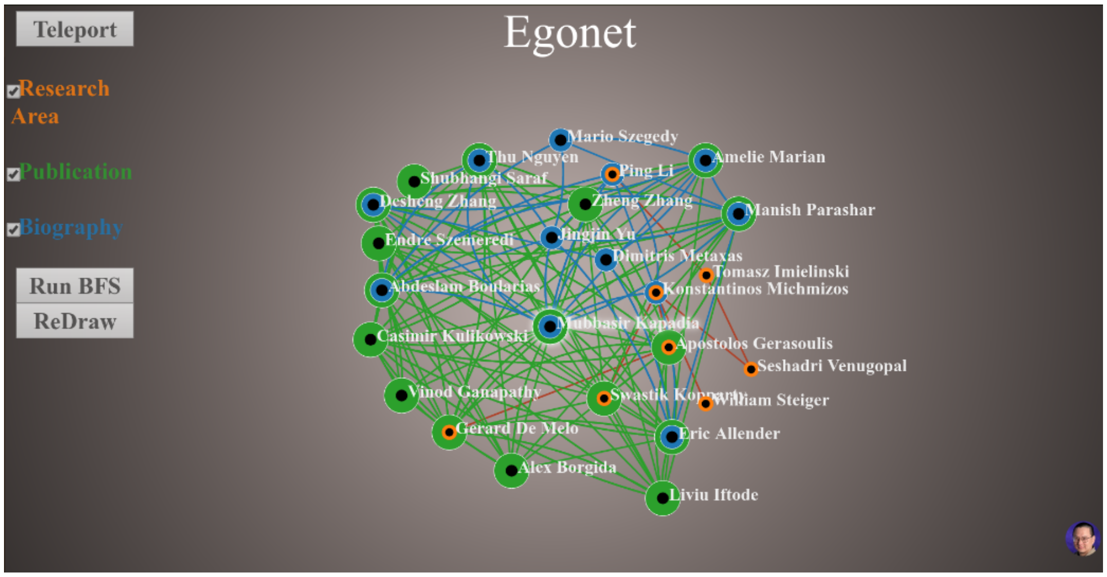
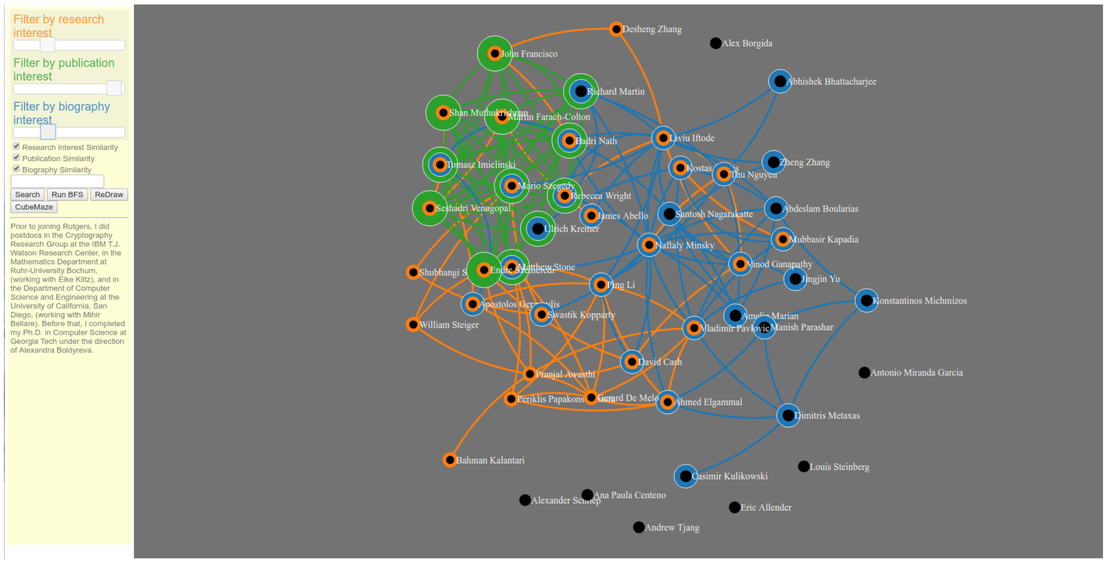


Figure 3.9: Cube Face of EgoNet

navigation to the cube sharing that face to the current local cube. This pushes the original cube out of the screen in the opposite direction. For similarity semantics, clicking the icon brings the corresponding cube to the visible screen space. The only difference between neighborhood semantics and similarity based on face semantics is that the former moves only in the domain of neighboring cubes, but the latter brings the cube positioned anywhere in the global space to the front of the camera.

3.2.3 Labeled Multi-DiGraph View

We view data units (cubes) as nodes in a multi-graph. These nodes can have k types of edges based on k-attributed data units. An edge between two nodes represents a relationship shared by the nodes based on that particular attribute. These edges are color coded symbolizing the color of the wall that these attributes are mapped to in their cubes. A node can have at-most k outgoing edges. Based on the number of different type of outgoing edges, a node is visually represented as surrounded by concentric colored circles, each circle associated with an edge type. This meaningful representation provides information on the types of similarities this node shares with its neighbors in the global space.



Prior to joining Rutgers, I did postdocs in the Cryptography Research Group at the IBM T.J. Watson Research Center, in the Mathematics Department at Ruhr-University Bochum, (working with Eike Kiltz), and in the Department of Computer Science and Engineering at the University of California, San Diego, (working with Mihir Bellare). Before that, I completed my Ph.D. in Computer Science at Georgia Tech under the direction of Alexandra Boldyreva.

Figure 3.10: Labeled Multi-DiGraph

3.3 Semantic Relationships Between Data Units

Cubes with shared faces present a higher degree of adjacency compared to vertical and horizontal "edges", or extreme points, all of which represent weaker forms of adjacency among the data units. Overall, in a particular state of navigation, a data unit may have 18 adjacent data units.

This adjacencies can be used to represent:

1. graph neighborhood relationship

2. and, similarity shared based on their endpoint components. For example, relationships between faculty members data units can be shown by computing similarity measures on research area, publications, biography, grants, news, awards, students advised" or courses taught.

3.3.1 Graph Neighborhood Relationships

In these semantics, the signs mapped at the bottom of each face in the cube represent cubes that are its physical neighbors. In the global view, cubes can be grouped automatically in their current global space layout according to their relationship with other cubes based on certain selected components. Also, cubes can be manually placed as neighbors to each other. In the local view, each cube may have icon at the bottom of each plane denoting its semantic relationship with the cube's icon.

3.3.2 Facet Similarity Relationships

We calculate the similarity for all facets of the cube and map the best matched case to each plane in the cube. For the face containing the cube's EgoNet, the cube's data unit with the highest score on the aggregated similarity measures among the remaining faces is used as the corresponding navigational sign.

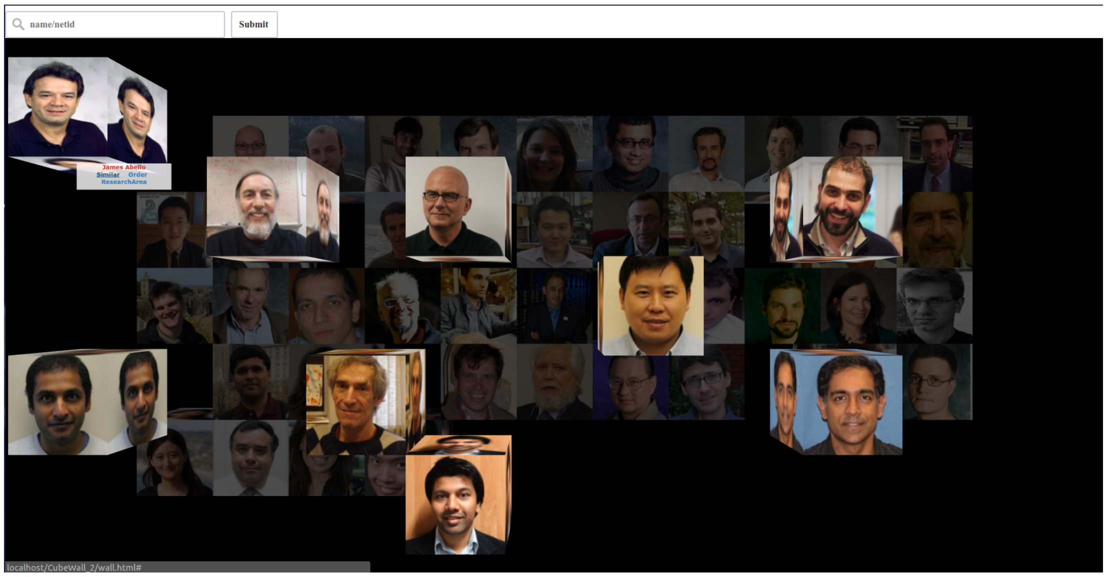


Figure 3.11: Global View: Showing Similar Cubes

3.4 Interaction Mechanism

For visual data exploration, human computer interaction is a necessity. Interaction mechanisms in well-planned modular systems help a user choose relevant data subsets and adjust the visual mapping to suit a particular course of visual exploration. We present a high level description of our fundamental cube maze interactions. To convey an impression of how these basic interactions can be applied by users, they are presented in an order that resembles a real usage scenario, rather than in an order that relates to interaction complexity.

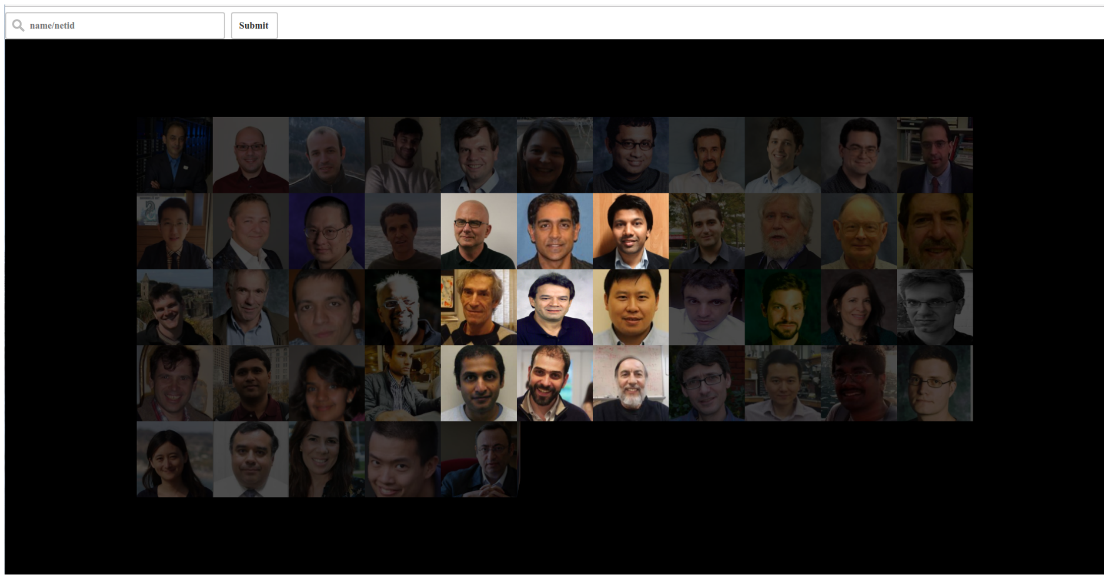


Figure 3.12: Grouping of Similar Cube at the Center

3.4.1 Hover

For a user to understand and explore the system, objects should portray information readily without the user having to perform some complex maneuvers. Hover is one such simple tool, in which a user moves the mouse over on an object the corresponding information is displayed. The Scenarios in our Cube Maze where hover is utilised are:

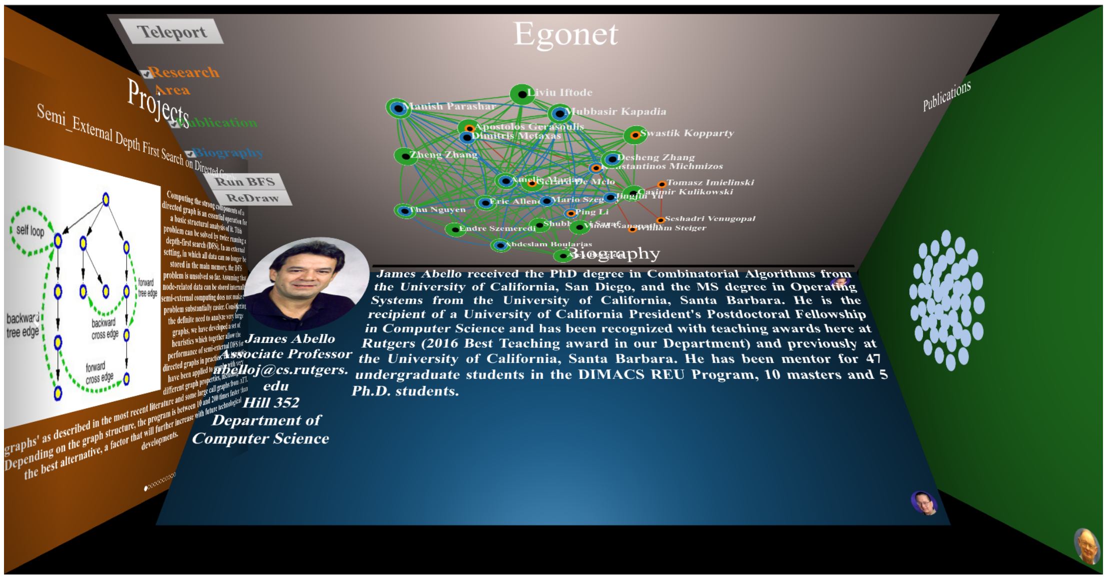
1. In the global view, hovering over a cube brings to view the data units “name”, and its attribute links such as “similar”, “order” and “research area” [fig:3.1].

2. After a user clicks the “similar” link on a cube, the cubes which share similarity are moved towards the user in the z-direction. Hovering on any of these cubes, displays relationship between this cube and the cube on which the user clicked similar [fig:3.11].

3. In the local view, the face containing the publications contains nodes which represent publications. On hovering the mouse on any node, a small window appears to describe the “Title” and “Authors” for the corresponding research paper [fig:3.4].

3.4.2 Click

A user can perform click operations that bring to view faculty member based on their research interests. The system visually groups them together at the center of the global view [fig:3.12] the user can double check and see each faculty members research area. On double click, a scene transition action is triggered which renders the local view for the selected cube. In the publication facet, clicking a node creates an i-frame window to display the webpage for that publication. Clicking is used also to pan projects sideways [fig:3.5]. On double click of any cube face, the face rotates to bring that face to the center facing the user [fig:3.13]. In the EgoNet view and in the labeled multi-digraph view, single clicks brings to view the neighboring nodes of the selected cube[fig:3.14]. On double clicking, the EgoNet of the selected cube is appended to the existing EgoNet. By clicking on the teleport button, the user is moved in the global space to the selected cube.



James Abello
Associate Professor
abelloj@cs.rutgers.edu
Hill 352
Department of Computer Science

James Abello received the PhD degree in Combinatorial Algorithms from the University of California, San Diego, and the MS degree in Operating Systems from the University of California, Santa Barbara. He is the recipient of a University of California President's Postdoctoral Fellowship in Computer Science and has been recognized with teaching awards here at Rutgers (2016 Best Teaching award in our Department) and previously at the University of California, Santa Barbara. He has been mentor for 47 undergraduate students in the DIMACS REU Program, 10 masters and 5 Ph.D. students.

Figure 3.13: Cube Rotation in Local View

3.4.3 Drag and Drop

A user can drag and drop cubes. It means a user can drag a cube by clicking and holding onto the mouse and drop it by releasing the mouse on top of another cube, the

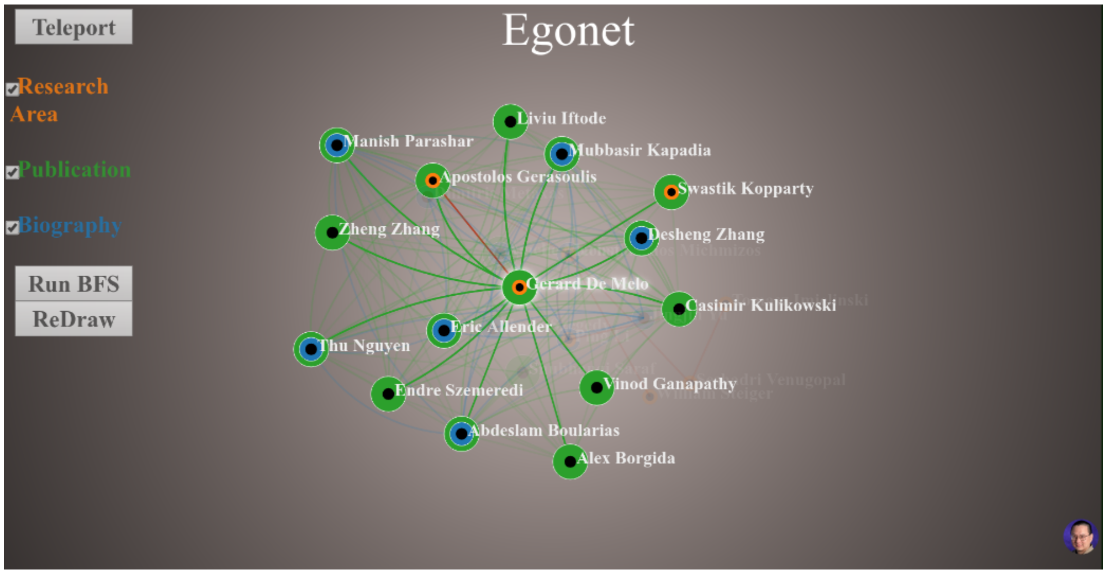


Figure 3.14: Showing Neighbor Nodes of a Highlighted Node in the EgoNet View

below is displaced to fit the dragged cube in the position occupied by the bottom cube. The user is given two mechanisms to complete this action:

- Lets denote the cube being dragged as ”source cube” and the cube at the position (index) where the source cube is dropped at, the ”destination cube”. The destination cube moves out in the z-direction allowing for the source cube to take its position. Afterwards, the destination cube moves to the position previously occupied by the source cube.

- Using the same convention as above, the destination cube shifts side ways either to the left/right based on the hole created by the source cube which in turn pushes the rest of the cube filling the source cube’s position and the source cube taking the position of the destination cube.

3.4.4 Similarity Sliders

In the labeled multi-digraph view, each edge contains a similarity weight. Sliders are provided to put a threshold on the edge to be displayed[fig:3.10].

3.4.5 Zooming, Searching and Selection Filters

Using the zooming feature, a user can focus on a particular section of the entire graph. A pinch-to-zoom mechanism is used for this purpose.Option of finding a faculty is provided in the global view [fig:3.1] and in the multi-digraph view [fig:3.10]. A Check box option is provided for the EgoNet and the graph view inorder to make data exploration more convenient.

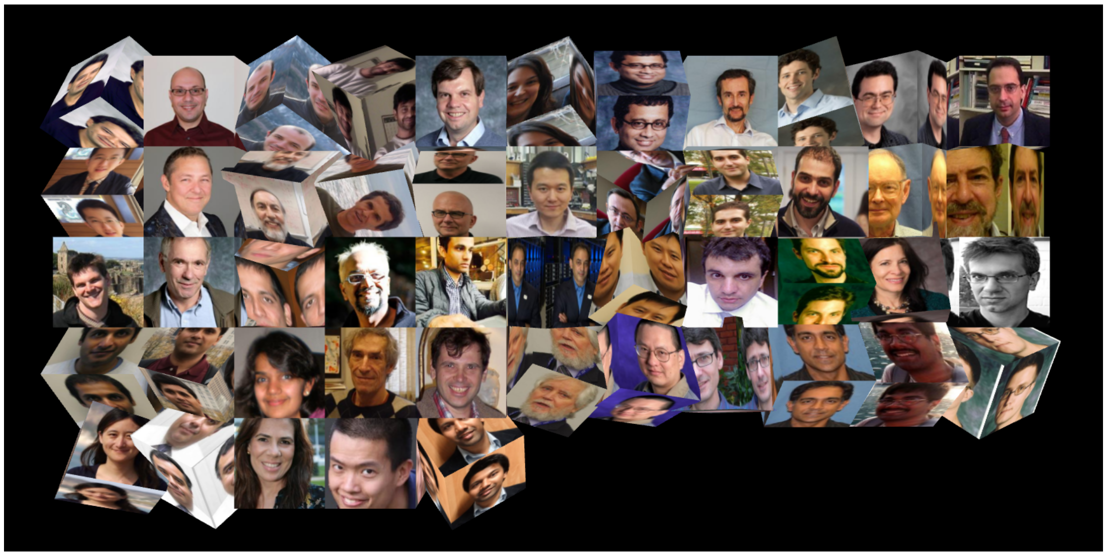


Figure 3.15: Local cube rotation in Global View

Figure 3.16: Cube Face expansion to Entire Canvas Space

3.4.6 Key Board Mapped Events

Key Board Mapping interactions are provided on this state of the interface. The user can hover on any cube and press the arrows ($\leftarrow, \rightarrow, \uparrow, \downarrow$). On pressing these keys the cube rotates on its local horizontal and vertical axes [fig3.15]. Key board mapping is provided to expand the image/information mapped to the cube face to the entire canvas size giving clarity on the mapped information [fig:3.16].

Chapter 4
Design and Architecture

4.1 System Design

The above challenges are daunting in nature, so it becomes very important to carve out a modular system design to accommodate different sections of the code without entanglement. In this section, we present the system layout and provide an example based on it.

We have followed a Model-View-Controller (MVC) design to implement Cube-Mazes. The **Model** represents an object carrying the data. The **View** is focused on the visualisation aspect on the front end using data received from the model. The **Controller** is the layer in between the Model and the View. The controller controls the flow of data into the view and updates the model whenever the data changes.

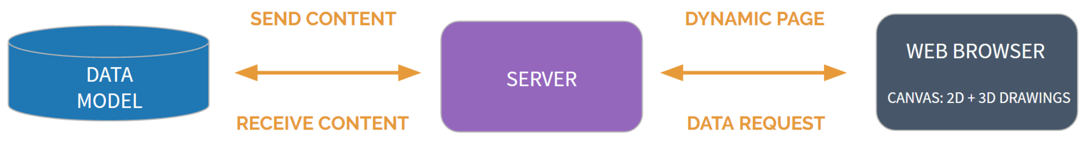


Figure 4.1: System Design based on MVC Design Pattern

4.1.1 Model

The model is designed to allow access to each attribute without having to access the entire database. The entities in the model are “Faculty”, “Publication”, “Project”, “Students Advised”, “Courses Taught”, “Events”. Fig:4.2 depicts the schema diagram of the model. These entities are created to contain information pertaining to each attribute in each data unit. Each data unit has an identifier which is a unique id. In our example, we can consider the "Faculty" list as nodes, each is represented as a cube in our system and the edges among them are determined by the similarity measures computed between their facets. These data facets can be entities themselves or entity attributes. Entity-Relationship diagram needs to be sturdy and robust for quick access of the data objects. This makes the visualisation and interaction both flexible and enhanced.

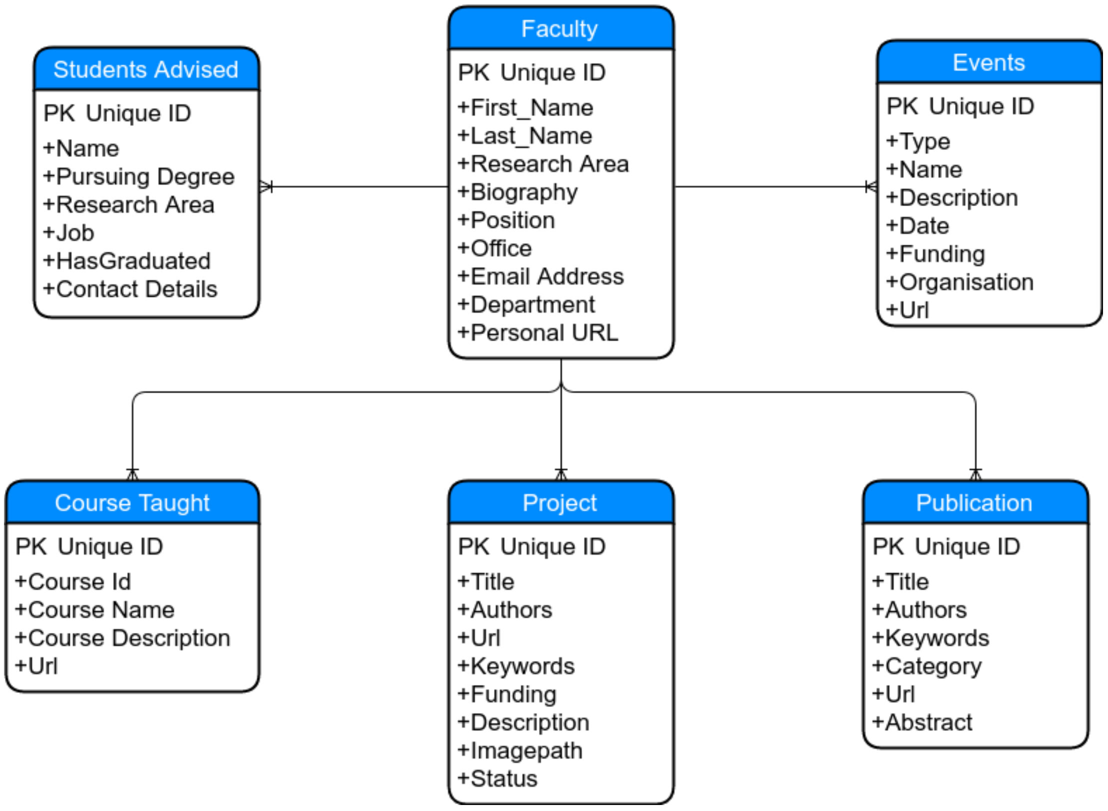


Figure 4.2: Entity-Relationship Diagram

4.1.2 View

The visualisation provides a couple of dedicated views, Global and Local. Each view is designed as a large interactive component that implements its own visual mapping and interaction as well as a common interface to access the data and visualization parameters. We have used two graphics engine to drive this visualisation part with animation and interactions. We use a three dimensional graphics engines that uses WebGL for rendering three dimensional objects and for the two dimensional graphics engine we use HTML and CSS for two dimensional drawing. Most of the latest browsers support both 3D and 2D drawing. The real challenge was merging these two worlds together as the WebGL rendering engine is loosely coupled with the DOM element [fig:4.3]. The layout of the interface consists of two dimensional elements attached to these three dimensional objects. This design pattern has made way for our content to become dynamic. The three dimensional layout or two dimensional drawings can change without affecting each other.

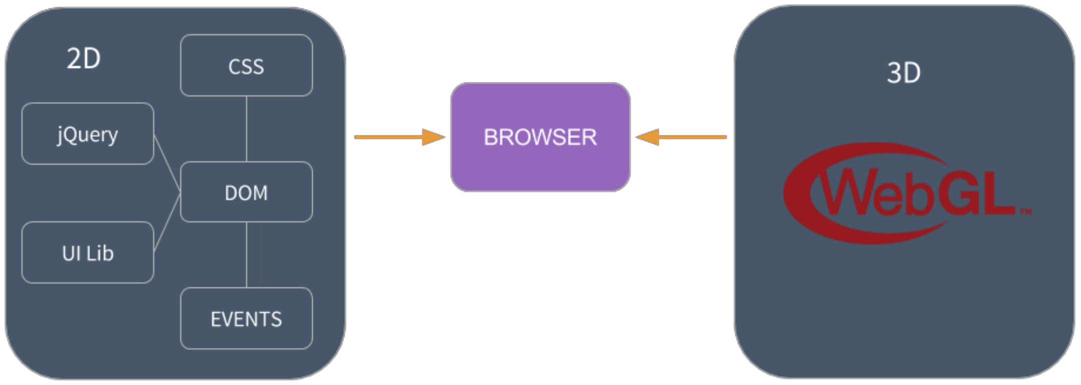


Figure 4.3: Bringing Three Dimensional and Two Dimensional together.

In the Global View, the data units represented as cubes are rendered as a three dimensional object using WebGL and are placed next to each other in rows and columns. In the Local View, each data unit is mapped to the entire screen space to give the user a feeling that he/she is placed inside the cube and is looking at the wall present in front. The user can see sideways and rotate the cube in the horizontal and vertical direction to bring any plane/wall to the front. Each wall is mapped with a data facet to yield information about the data unit.

Another module, completely based on two dimensional drawing focuses on the multi-colored multi-digraph. In this view, we draw cubes as nodes of a graph and a node can have k type of edges. These edges are defined for each component in the data unit, so for a k attributed data unit, a node can have at-most k edges. These k edges are color-coded with the color as the wall in the cube which contains its corresponding attribute. The edge computation is based on the relationship two nodes share between each attributes in a k attributed data unit. Similarly, nodes are filled with colors based on the different types of outgoing edges. Each edge has certain similarity value mapped to it. To assist the user in interactively exploring the data and the parameter space, we have incorporated a slider mechanism to add or remove edges based on mean and standard deviation for the k components in the data unit.

4.1.3 Controller

This regulates the flow of data from the model to the view. Currently, we do not update the model on request from the view. In information visualization, interaction is modeled as adjustment of the data model, which includes the raw data and its visualization parameters [15]. Basic checks are put in place for determining the validity of the requested parameters, which control the movement of data to the front end. There are two ways to adjust drawing in the front end, either by direct manipulation or via dedicated graphical user interface. Further distinction has been made in differentiating the global effect from the local effect mechanisms. For example, in the global view rotating a cube is a local effect but grouping cubes based on relationship changes the global layout. Interacting on a single cube in the global view pushes information only about the selected cube. In case of a local view, data sent from the controller contains information on all components of a data unit. Apart from regulating flow of raw data, the controller contains logic to process the data to determine relationships between cubes based on attributes of the data unit. The following subsection provides details on the algorithm used to compute similarity measure.

4.1.4 Similarity Measures

In our running data example, the attributes of a faculty are “publication”, “biography”, “project”, “news”, “students advised” and “courses taught”. Let us take the case of finding relationships between two faculty members based on their publications abstract data.

We used web harvesting to collect the list of publications[16] for each faculty in the department. Using the publications list, we again harvest “abstracts” and add them to our database. The next step in the pre-processing creates bag of words model. This means that we create a list which contains all the unique words in the entire corpus (collections of publications) with no frequency count. We create, for each publication, a vector of words with their count. We use "TF-IDF"[17] (Term Frequency- Inverse Document Frequency) to compute the weight of each term in the corpus. Term Frequency is a count of a term in one publication. Inverse Document Frequency of a term is the log of the number of publications in the corpus over the count of publications which contain the term. We smoothen the IDF by adding one. The TF-IDF value is the product of TF and IDF. Here, the publication is a vector where each component of a vector corresponds to the TF-IDF value of a particular term in the corpus dictionary. Dictionary terms that do not occur in a document are weighted zero. Using this kind of representation in a common vector space is called the vector space model [18]. Before computing the similarity measure, we group vectors of each faculty and normalize them, so that the value is spaced equally for faculty with more publications and faculty with less publications.

$$idf(t, D) = 1 + \log \frac{N}{|d \in D : t \in d|} \tag{4.1}$$

with,

N: total number of documents in the corpus, $N = |D|$

$\frac{N}{|d \in D : t \in d|}$ number of documents where the term t appears.

We have used two kinds of similarity measures, Cosine Similarity and Jacard Similarity. We have used cosine similarity to avoid the bias caused by different document lengths. The inner product of the two vectors (sum of the pairwise multiplied elements) is divided by the product of their vector lengths. This has the effect that the vectors are normalized to unit length and only the angle, more precisely the cosine of the angle, between the vectors accounts for their similarity.

$$similarity(Faculty_i, Faculty_j) = \frac{V(\vec{Faculty_i}).V(\vec{Faculty_j})}{||V(\vec{Faculty_i})||.||V(\vec{Faculty_j})||} \tag{4.2}$$

Secondly, Jacard Similarity is based on intersection over union of the set of objects

$$JS(Faculty_i, Faculty_j) = \frac{|Faculty_i \cap Faculty_j|}{|Faculty_i \cup Faculty_j|} \tag{4.3}$$

where, Set Faculty is the cardinality of Faculty denoted by $\|Faculty\|$ counts how many elements are in Faculty. The intersection between two sets of $Faculty_i$ and $Faculty_j$ is denoted by $Faculty_i \cap Faculty_j$ and reveals all items which are in both sets. The union between two sets A and B is denoted by $Faculty_i \cup Faculty_j$ and reveals all items which are in either set.

4.1.5 Path Traversal Algorithm

Breadth First Search(BFS) is implemented on the Labeled Multi Digraph and the EgoNet views. A node can be selected by clicking on it or by using the locate interaction mechanism. After that BFS is computed on the entire graph using the selected vertex as source. Breadth First Search is an algorithm for traversing or searching graph data structures. Starting from the source vertex, it explores the neighbor nodes first, not moving to the new next level neighbors.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

The main focus of the work was on portraying data in an intuitive manner. We focused on a novel three dimensional representation enhanced by various multi-attributed interaction tools. These include hover, click, drag and drop, zoom, sliders, check boxes, key board mapped events, and search option. Cube Maze provides unified views of the multi-attributed data at both global and local levels. The graph theoretical counterpart of cube mazes is a labeled multi-digraph. The labeled multi-diagraph provides an alternative view of the cube similarities based on a variety of threshold values.

5.2 Future Work

Some of the possible pathways that could be undertaken to make this work more robust and flexible are:

I If the data is not structured as Data Units, the system needs to learn "Entity", "Components" and "Data Units" by adding machine learning recognition modules and detect labeled data relationships.

II Since, the server side needs to maintain the corresponding labeled multi-digraph among the discovered Data Units, the Scalability of this multi-digraph needs to be addressed both when the data is at rest and when the data is being streamed.

III In this work, we have mapped Data Units into cubes in 3D. However, one could envision other visual representations and "novel" interaction mechanisms.

IV We did not focus in this work on visual analytics. We intend to incorporate algorithms to:

   i detect the most "influential" Data Units in terms of their facet similarity.

   ii discover the most "cohesive" clusters.

   iii measure Data Units World Reachability.

V We would like to extend the work to cover more general ranked data units, ant to provide specialized ways to navigate between them. This will require us to extend the strength of similarities to include lower dimensional similarities via edge and vertex covering.

References

[1] B. Shneiderman, "Direct manipulation: A step beyond programming languages," *ACM SIGSOC Bulletin*, vol. 13, no. 2-3, p. 143, 1982.

[2] C. Ahlberg, C. Williamson, and B. Shneiderman, "Dynamic queries for information exploration: An implementation and evaluation," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 619–626, ACM, 1992.

[3] R. Kosara, H. Hauser, and D. L. Gresh, "An interaction view on information visualization," *State-of-the-Art Report. Proceedings of EUROGRAPHICS*, 2003.

[4] M. O. Ward and J. Yang, "Interaction spaces in data and information visualization.," in *VisSym*, pp. 137–145, 2004.

[5] J. S. Yi, Y. ah Kang, and J. Stasko, "Toward a deeper understanding of the role of interaction in information visualization," *IEEE transactions on visualization and computer graphics*, vol. 13, no. 6, pp. 1224–1231, 2007.

[6] C. E. Weaver, *Improvise: A User Interface for Interactive Construction of Highly-coordinated Visualizations.* PhD thesis, Madison, WI, USA, 2006. AAI3234816.

[7] M. Q. Wang Baldonado, A. Woodruff, and A. Kuchinsky, "Guidelines for using multiple views in information visualization," in *Proceedings of the working conference on Advanced visual interfaces*, pp. 110–119, ACM, 2000.

[8] C. Tominski, J. Abello, and H. Schumann, "Cgvan interactive graph visualization system," *Computers & Graphics*, vol. 33, no. 6, pp. 660–678, 2009.

[9] J. Abello, F. Hohman, and D. C. Chau, "Edge decomposition and 3d visualisation," in *Preparation*, 2017.

[10] N. Linial and Y. Peled, "On the phase transition in random simplicial complexes," in *Annals of Mathematics*, vol. 184, pp. 745–773, 2016.

[11] P. Cignoni, L. De Floriani, C. Montani, E. Puppo, and R. Scopigno, "Multiresolution modeling and visualization of volume data based on simplicial complexes," in *Proceedings of the 1994 symposium on Volume visualization*, pp. 19–26, ACM, 1994.

[12] J. Abello and F. Queyroi, "Network decomposition into fixed points of degree peeling," in *Social Network Analysis and Mining*, May 2014.

[13] J. Abello, P. Broadwell, and T. R. Tangherlini, "Computational folkloristics," in *Communications of the ACM*, vol. 55, pp. 60–70, July 2012.

[14] L. Laura and F. Santaroni, "Computing strongly connected components in the streaming model," in *Theory and Practice of Algorithms in (Computer) Systems. Lecture Notes in Computer Science*, pp. 193–205, 2011.

[15] T. Jankun-Kelly, K.-L. Ma, and M. Gertz, "A model and framework for visualization exploration," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 2, 2007.

[16] "Dblp: Computer science bibliography."

[17] Wikipedia, "Tfidf — wikipedia, the free encyclopedia," 2017.

[18] G. Salton, A. Wong, and C.-S. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol. 18, no. 11, pp. 613–620, 1975.