Università
della
Svizzera
italiana

Faculty
of Informatics

Bachelor Thesis

June 17, 2010

Ebony

Visualizing the DBLP Database

Remo Lemma

Abstract

The DBLP (Digital Bibliography & Library Project) database archives information on major computer science journals and proceedings. We developed Ebony, a system that gives full access to the information contained in DBLP, offering a number of customizable visualizations with freedom in choosing metrics and parameters of interest.

Ebony aims to help the user to discover, study and visualize relationships among authors, groups of authors and entire research areas, taking advantage of the information stored in the database.

Advisor
Prof. Dr. Michele Lanza
Assistant
Ph.D. Alberto Bacchelli

Advisor's approval (Prof. Dr. Michele Lanza): Date:

Acknowledgments

I would like to thank Professor Michele Lanza for his advice, his support and for giving me the opportunity to work on this interesting project, which gave me the chance to work in a research area that was not covered during my previous studies.

In addition I express my gratitude to Alberto Bacchelli for his help and support that he gave me throughout the whole project. Finally special thanks to all my friends and my family for their patience and their help during my bachelor studies.

Contents

List of Figures

Chapter 1

Introduction

The DBLP (Digital Bibliography & Library Project)[1] database contains useful information for computer scientists. It collects a relevant number (in May 2010 the database contains more than 1.3 million publications and 800'000 different authors) of scientific publications related to computer science.

The DBLP project database is useful for searching books and articles about specific topics offering a single entry point for searches.

In addition, DBLP allows one to study the information contained in the database and identify and analyze different relationships among single authors, group of authors and entire research areas (represented by the authors themselves).

Given the size of the database a mere textual representation, as it is offered by the DBLP website, makes it hard to discover or study the desired correlations.

For human beings it is simpler to interpret and understand visual representations, thus a system for navigating through a database like DBLP should have the following features:

1. Give complete access to the information contained in the database.
2. Visualize information in different but meaningful ways.
3. Give easy and fast instruments to manipulate data and extract relationships or metrics of interest.

The goal of this project is to create Ebony[2], an application that implements all these features and gives the opportunity to people to fully take advantage of the information stored in the DBLP database.

We have developed Ebony as a web application independent of the web browser used on the client machine, to make it easy for the user to access it remotely. Furthermore, as the DBLP database is periodically updated, it would be tiresome for the user to have to download every new version. In the case of Ebony this operation is performed by its maintainers.

[1] http://dblp.uni-trier.de
[2] http://ebony.inf.usi.ch

Currently Ebony includes the following features:

- Selection of the desired metrics.
- Metrics combination to create relationships.
- Creation of groups of authors.
- Time interval selection.
- Visualization of groups and relationships using any of the following fully-customizable visualizations:
    1. Line Chart
    2. Stacked Column Chart
    3. Stacked Bar Chart
    4. Group Graph
    5. Relationships Graph
    6. MDS Graph
    7. Force-directed Graph

1.1 Document Structure

The document is structured as follows.

In Chapter 2 the work related to Ebony is presented and the importance of visualizing data is underlined, supporting this thesis with a short theory about the concept of visualization.

In Chapter 3 we talk about the internals of Ebony and we deeply explain the details about the usage of the software.

In Chapter 4 we validate Ebony presenting a research in which Ebony is used to discover and analyze the group of professors present at the faculty of informatics at the University of Lugano.

In Chapter 5 some conclusions about the entire project are drawn.

In Appendix A we present the implementation and the idea behind the GWTDisplay library, used in the project and created concurrently with Ebony.

In Appendix B the implementation details of Ebony are presented.

Chapter 2

Related Work

Visualizing data is a task that is performed daily by people in different environments for different reasons. Most of the times, we do not even realize why the visualization presented is built that way or whether there is a better representation of the underlying data.

This chapter gives an explanation on the concept of visualization and presents software systems that are similar to Ebony, giving an overview of the motivations that induced us to develop Ebony.

2.1 Information Visualization

Visualizing an information, or a set of information, means to create a graphical representation that emphasizes in some direction the analyzed entities. This allows us to model not only real objects (that already have a graphical representation by themselves), but also to create visualizations of abstract entities, extracting some key attributes and mapping them in a graphical representation.

The human brain is able to process much information in parallel, but such potential cannot be exploited with a simple text representation or by listening to someone: texts and speeches give sequential information to the brain.

Therefore, when we face the presence of a huge or complex data set, it is more desirable to create a visualization, because by looking at a graphical representation makes it possible to extract many information in a shorter period of time and find new relations in the data.

Information visualization is an active research area, because it is not obvious how to represent abstract entities in a meaningful and intuitive way. Moreover, such research helps to identify the graphical representations that are better understood by our brain and that can consequently be processed more effectively.

For this reason, it is not sufficient to map an entity (which might also represent an abstract concept) to a visualization, because if this one results not intuitive or badly designed it will not exploit the potentiality of our brain. Therefore the result will be misunderstood or the information will be difficultly absorbed: the same issues that are encountered with very long texts or speeches.

A good example for demonstrating the potential of a visualization is a roadmap.

If you have to find a street, a roadmap will help you to identify the route that you have to cover, much more than reading all the indications in textual form. In fact, nowadays, visualizations are often used to speed up the information that human beings are able to absorb in a short period of time, and to simplify tasks that otherwise would be really complicated. Other popular examples of successful visualizations are weather forecasts and air traffic radars.

2.1.1 Scientific Visualization

Scientific research usually has to deal with a huge amount of data, which often represent abstract concepts. Representing these data in a graphical way is fundamental for analyzing information, understanding relationships, study the real meaning of the collected data and communicating results. Moreover, visualizing data allows to discover new trends and correlations that might have been unexpected.

Dealing with abstract concepts makes it difficult to create a valid and meaningful representation of the information. This representation should usually give a logical representation of the model behind the data, as there is most probably no physical model that can be used.

Another problem, which has to be solved while representing data, is the selection of the *type* of visualization: there are many different types of visualizations that are suitable for the same model of data. Choosing one instead of another is basically linked to what is essential in the analysis and what is important to emphasize.

Taking the wrong decision leads to views that might still represent a meaningful representation of the data, but most probably will not convey the desired message.

2.2 DBLP Visualization Tools

Mainly there are two projects that try to offer visualizations for the DBLP database: **DBLPVis**[3] and **DBL-Browser**[2].

2.2.1 DBLPVis

DBLPVis (Figure 2.1) is a web application developed and provided by Florian Reitz (University of Trier, Germany), which aims to search connections between different entities in the DBLP database.

DBLPVis has three main entities that can be visualized:

- **Person:** Each author in the database is defined as a person.
- **Word:** Terms that are used in the publications.
- **Stream:** Each conference and journal (generally each venue) is defined as a stream.

In DBLPVis, it is possible to separately analyze any element that belongs to one of these three categories and view the graph that represents the relationship among the selected entities.

The visualization consists in a radial graph that is modeled taking into account a specified number of nodes and a specified period of time. It gives the possibility to switch between the *intensity view*, in which the single values are emphasized, and the *timecolor view*, in which each year gets a different color, allowing to identify different periods.

A problem that can be noticed by using DBLPVis is that the user has no real choice over the data that the visualization takes into consideration.

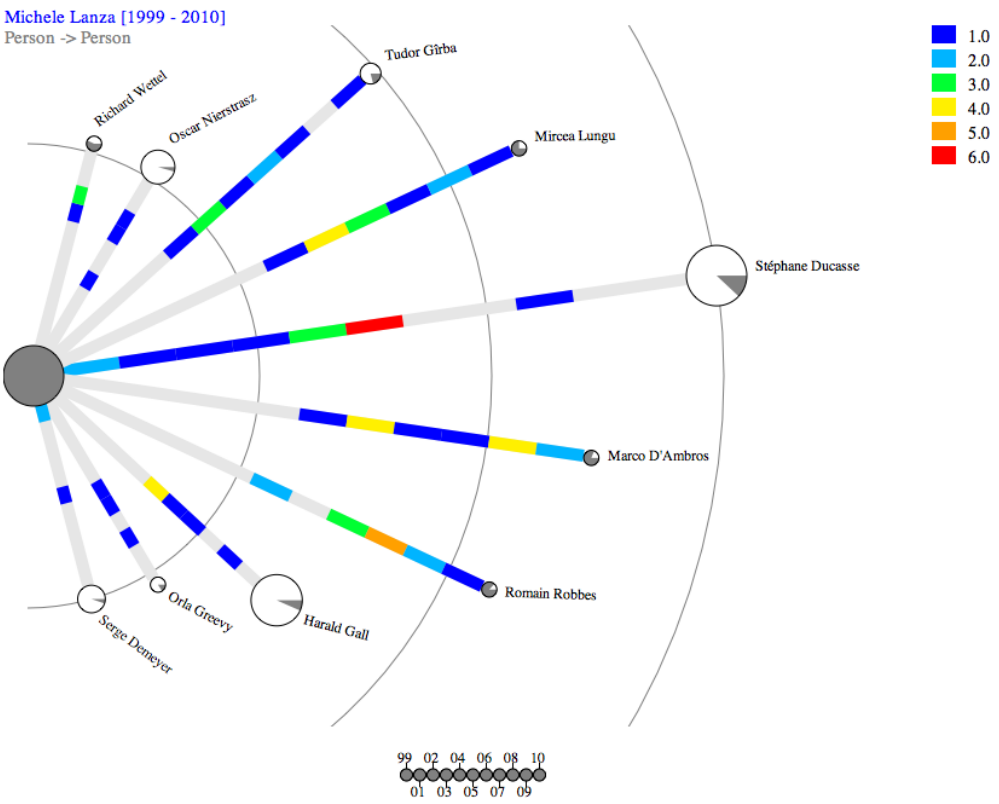


Figure 2.1. DBLPVis Person to Person Intensity View

2.2.2 DBL-Browser

DBL-Browser (Figure 2.2) is a desktop application that can be used to browse the entire DBLP database. It behaves like a real web browser, with the only exception that it works offline.

It is independent from the operating system installed and it only needs a DBLP data file, that can be obtained from the official DBLP website.

One drawback of being a desktop application is that the user needs to download an updated data file when the DBLP database is modified.

At the startup the entire data set has to be loaded (which takes some time) and afterwards it is possible to navigate through the whole database, as it would be a real web browser.

DBL-Browser offers many kinds of visualizations, provides information about relationships among authors and venues.

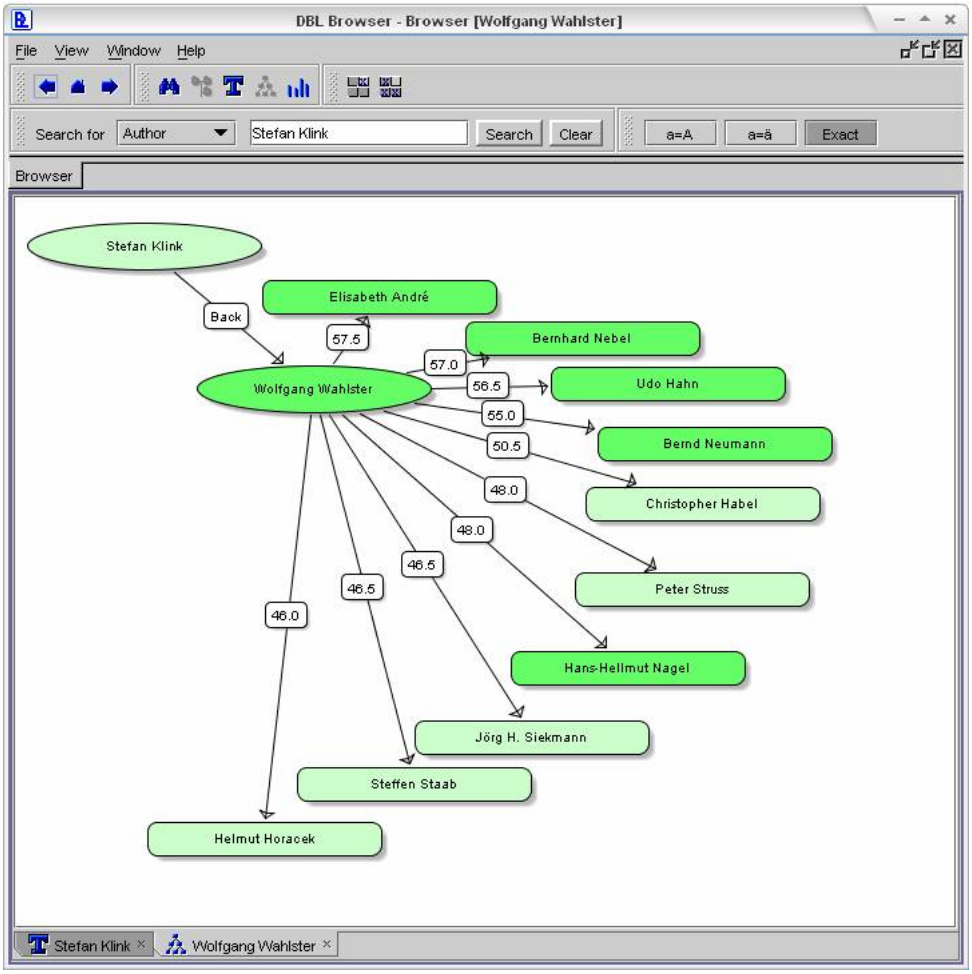


Figure 2.2. DBL-Browser Related Authors View

2.2.3 Ebony

We have seen in the previous sections the software systems that currently give the possibility to visualize the data contained in the DBLP database. Each of them has particular targets and is suited for certain types of analysis.
The goal of Ebony is to combine the *full access* to the data contained in the database with *flexible* and *fully customizable* visualizations.

Using Ebony the user will be able to decide by himself what he wants to visualize, in which way and which metrics have to be considered. Ebony also targets to guarantee a well-performing and scalable service, which automatically offers the data contained in the last update of the DBLP database. The software system is presented in detail in Chapter 3.

Chapter 3

Ebony

In this chapter we are going to present Ebony. We will decompose the application, discussing and analyzing all the important parts separately.

A preview of a visualization produced in Ebony is represented in Figure 3.1

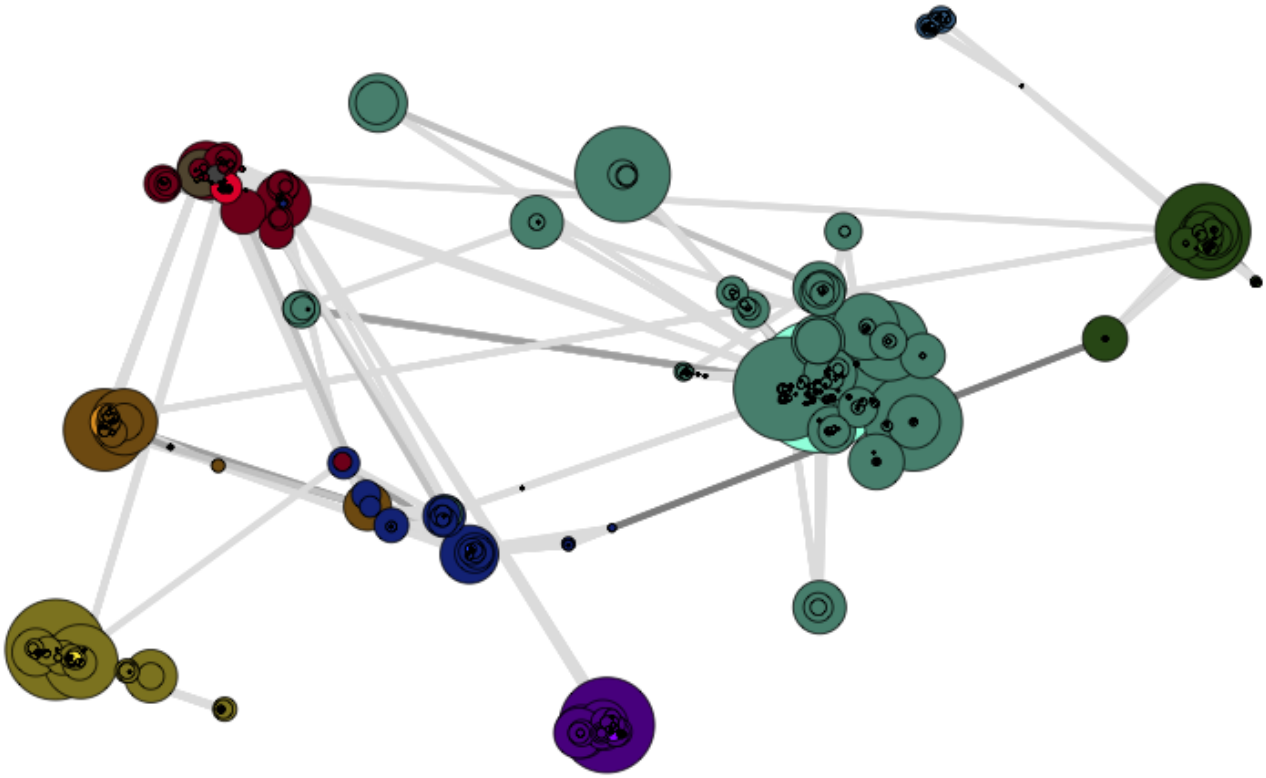


Figure 3.1. Example of a visualization produced by Ebony

3.1 Main Idea

Ebony is a web application that gives to the user the ability to visualize the authors (and their information) contained in the DBLP database, building relationships based on the collaborations on publications.

The information extracted from the DBLP database has been converted into a simple model (Figure 3.2). However Ebony allows the user to create complex associations between any data contained in the model, allowing to traverse the whole data set from different perspectives.

Each association gives raise to a different visualization that visually quantifies and qualifies relationships among the desired set of authors and their set of collaborators under different view points.

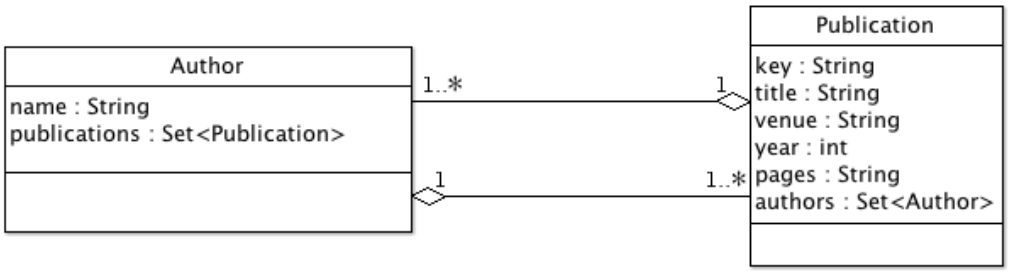


Figure 3.2. Model of the data extracted from the DBLP Database

3.2 Technologies and Libraries

Ebony has been developed using the Google Web Toolkit (GWT [5]), a set of tools to develop Javascript front-end applications in Java. Moreover, GWT implements a powerful asynchronous RPC (Remote Procedure Call) system to handle client-server communication. Furthermore the toolkit takes also care of most of the issues caused by the differences in the standard used by the web browsers. GWT also gives the possibility to implement the server side using Java Servlets, allowing to develop client and server part as they would not be disjoint.

We decided to realize only 2-D visualizations, realized in the HTML5 Canvas using an external GWT library (GWT-Graphics [4]) that automatically generates the needed code.

Other external libraries (LinLogLayout [8] and part of Taxionomy [10]) have been used as support for the implementation of some of the visualizations offered by Ebony.

3.3 System Requirements

A user who is willing to use Ebony simply needs to have access to one of the supported web browsers (Section 3.3.1) and to an internet connection (which directly affects the speed of the client-server communication).

3.3.1 Browser Compatibility

As mentioned before, Ebony uses the HTML5 Canvas for drawings, however currently only the canvas based on the SVG (Scalable Vector Graphics) specification are supported. Thus, the visualizations produced by Ebony can be correctly visualized only by the browsers that support SVG.

Ebony relies on the canvas and uses only standard fonts and SVG standard functionalities. Therefore, as long as the browser supports SVG, Ebony is going to work correctly.

3.4 Performance and Scalability

While designing Ebony, and in general any web application, the developer must take into consideration the load of many concurrent users that could utilize the system.

As a design choice, we decided to minimize the number of client-server communications and the amount of data exchanged, letting the client-side handle all the computations needed for data analysis and drawings.

This makes the software scalable, ensuring stable performances for each client even when many different users stress the service concurrently.

On the other hand, as the client-side has to perform lots of computations, which might be quite heavy (depending on the complexity of the visualization and of the number of authors involved), the general performance of the client machine (and of the browser) affects the performance of Ebony.

The choice of the browser can influence the performance of Ebony as each browser has its own implementation of the HTML5 Canvas and might have a particular way to handle JavaScript, or other optimizations.

The speed of the Internet connection influences directly the rapidity of the client-server communications.

Since the performances of computers are growing over the time, the single client performance is most probably going to improve in the future, which would not be the case if the system was not scalable.

3.4.1 Client-side Cache

To improve the performance in the client-side of Ebony, we implemented a small caching system. When an author of a group is loaded, he is automatically saved in the cache, so that it will not be necessary to communicate again with the server for retrieving the same author.

Although the size of the cache is small (approximately 50 entries), it improves the performances heavily, especially when the same authors are frequently analyzed.

3.5 Metrics

The DBLP database also contains data that is not relevant for the goals of Ebony. Thus, we choose to offer the user a set of important and relevant metrics that can be combined to obtain different kinds of relationships.

Obviously, only some combinations are meaningful, but the choice of letting the user combine the metrics makes it simpler to expand the number of metrics in the future, also enlarging the possibility of having novel relationships.

The metrics are divided in three main categories: data series, relationships data, and single-value data.

3.5.1 Data Series

These types of metric represent a collection of data, generally used to construct a chart. The data series available are:

- **Year:** Series data that contains all the years over which data for the publications is available.
- **# Total Publications:** Each year for which there is data available for the author, the number of total publications, until that specific year, is added to this series.
- **# Single Year Publications:** This data series contains the number of publications that have been produced by the author each year.
- **Venues:** The name of each venue for which the author has published.
- **# Venues Total Publications:** The number of total publications per venue. The order in the collection follows that in the data series that contains the name of the venues.
- **Coauthor Name:** The names of the coauthors.
- **# Collaborations:** The number of collaborations per coauthor. The order in the collection follows that in the data series that contains the name of the coauthors.

3.5.2 Relationships Data

Relationships data is used to have a metric for defining connections among different authors.

Currently the only metric available is the number of collaborations.

3.5.3 Single-value Data

This data is used to characterize an author. The metrics currently available are:

- **Total Publications:** The number of total publications.
- **Relative Importance:** The number of total publications divided by the number of years of activity (the interval of time in which the author has published).

3.6 Definition of Concepts

In Ebony there are some basic and recurring concepts, that must be understood to fully comprehend the structure of the system.

View

A view is a type of visualization properly configured, with the design metrics to focus on and the options that are relevant for the analysis having the desired value.

Group

During an analysis the set of data is always linked to the authors that the user wants to focus on. These authors are part of a group. It is possible to define different groups.

Coauthor

Each author in the database might have many coauthors (i.e. authors that have worked together on one or more publications). The strength of this partnership is given by the number of publications shared by the two authors have collaborated.

The main idea is to first collect all the authors that a user wants to examine in a group and afterwards configure the view that is suited for the analysis. The coauthors are used by some views, depending on the visualization.

It is possible to visualize any group with any view, but usually, for each group, there are visual options that have to be adjusted to have the best visual impact.

Thanks to this design choice it is easy to analyze relationships and the evolution of a group of authors. Moreover it is possible to determine the position of an author with respect to a topic or a research area (that can be defined in Ebony as a group of authors known to be part of that area).

3.7 Registered Users

Register a user in Ebony gives some advantages:

- Saving and loading different groups of authors. This allows an easy and fast reconstruction of groups that are often used.
- Saving and loading the views. We only save parameters that are independent of the group of authors currently visualized (See sections 3.10 and 3.11 for details about views).

It is not possible to access Ebony with the same user from two different clients at the same time; if this happens the first client is disconnected.

When a registered user logs in, all the saved groups and views are loaded, so they can be immediately used.

3.8 User Interface

The user interface of Ebony has been designed to be intuitive and simple.

3.8.1 User Registration & Login

The first operation that a user can perform when Ebony opens up is login as a registered user, register a new one, or login as an anonymous user.

Anonymous users have not the same privileges as registered users (see Section 3.7), but it is an ideal choice for producing a rapid visualization.

3.8.2 Ebony User Interface

The main user interface of Ebony is shown in Figure 3.3. From there, the user can perform all commands and use all the features provided by the application.

Some views offer the possibility to open popups to further customize the visualization; these special options will be discussed in the chapters in which each view is presented.

Ebony has been developed to be similar to a web browser. The two main parts of the graphic interface are numbered in the figure.

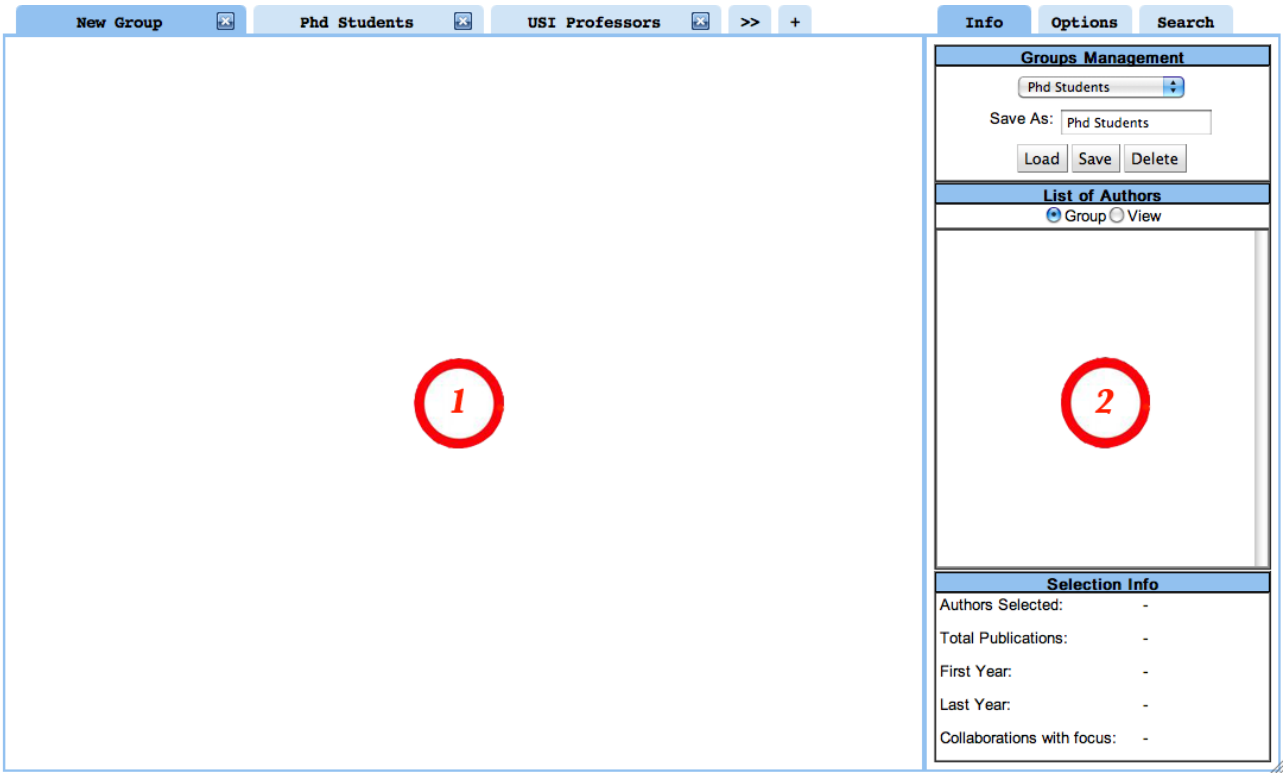


Figure 3.3. Main View of Ebony

1. Visualization Part

The visualization part is the main part of Ebony. It displays the views.

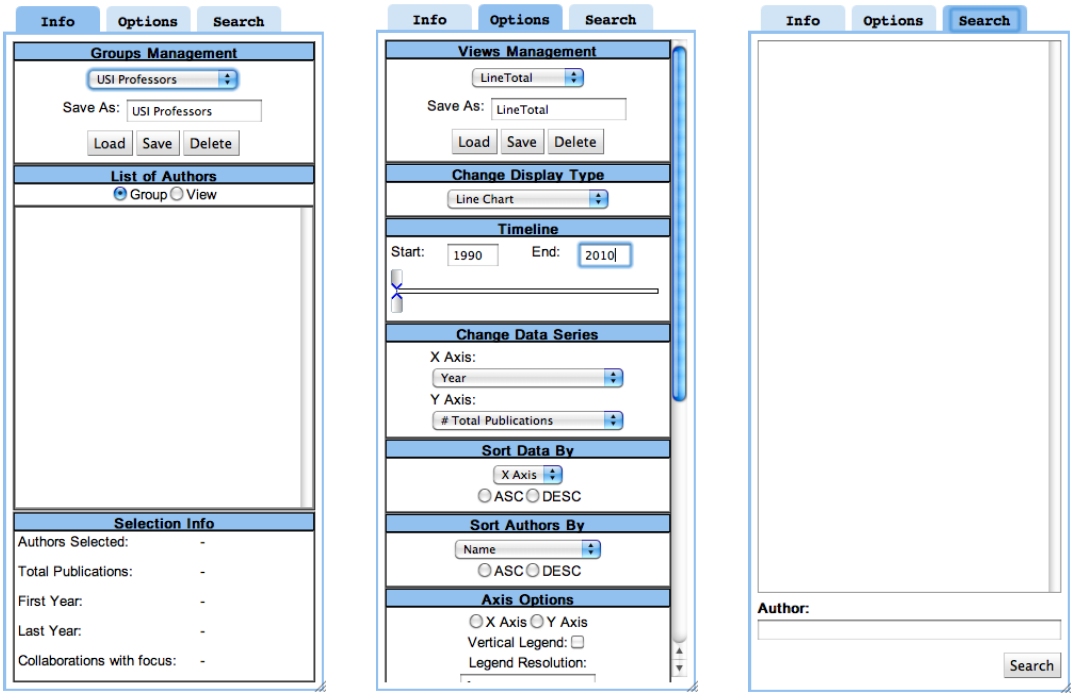
On the top of this part, there is a tab system that simulates that used in modern browsers. Thanks to this, the user can open more visualization tabs, make real-time comparisons, and there is no need to open multiple instances of Ebony, as it works, internally, as a real web browser.

2. Control Part

The control part is composed by three tabs, with all the instruments to create groups of authors, build and customize a view, and obtain information about the displayed authors.

All the tabs are structured as containers for widgets, each one offering a different functionality to the user.

These tabs adapt and update its content based on the currently selected visualization tab.



(a) Info Tab (b) Options Tab (c) Search Tab

Figure 3.4. Tabs of the Control Part

- **Info Tab** (Figure 4.6(a))
  If the user is registered, this tab offers the widget to save and load groups. Moreover, the user can modify the current group, view the authors that are currently visualized. On the bottom of the tab there is a widget that gives more detailed information about the authors that are currently selected.

- **Options Tab** (Figure 4.6(b))
  This tab gives access to all the configuration parameters of the current view; each of them is modifiable in a widget. For a registered user the first widget in the tab can be utilized to save or load views.

- **Search Tab** (Figure 3.4(c))
  This tab gives the possibility to search authors in the database and add them to the current group of authors, and, therefore, also to the current view.

3.8.3 Commands

Ebony has a simple and intuitive set of commands to perform all the basic operations to interact with the view.

Not all commands may be available for all kinds of view and or the behavior might slightly differ depending on the specific needs of the view.

Here we show a list of all the basic commands with their standard behavior:

- **Mouse Move**
  When the mouse is moved over a shape that represents an author, the name of the author appears in the canvas. This is useful to study and identify the output produced by the analysis.
- **Mouse Click - Left Button**
  When an author in the visualization, or in the list of authors in the info tab, is left-clicked it will be marked as selected. When the SHIFT key is pressed, the author is added to the selection, otherwise it becomes the only one that is selected. If an author is clicked twice, it is removed from the selection.
- **Double Click - Left Button**
  Adds the authors that are currently selected to the current group of authors.
- **Enter key**
  The Enter key is used to add the authors that are currently selected to the current group of authors. In most of the option components that need to enter text, pressing the enter key makes Ebony re-evaluate the data with the new parameters.
- **SHIFT + DEL, or SHIFT + Backspace**
  Deletes the authors that are currently selected from the current group of authors.
- **Mouse Drag**
  Dragging the mouse in the visualization translates the whole drawing by the amount of pixels for which the mouse has been dragged.
- **Mouse Drag + SHIFT**
  Rotates the drawing, if this operation is available for that kind of visualization.
- **Mouse Scroll, or Z/X**
  Zooms in (Z) or out (X) of the drawing. This is an implementation of a scaling transformation. The view point from which the zoom is performed may vary, depending on view currently adopted.
- **R key**
  Resets the drawing to the initial values. Usually such values are the scaling factor and the positions of the different shapes drawn.
- **Mouse Click - Right Button**
  A popup appears, giving the possibility to choose between different options. This is discussed deeply in Section 3.8.4.

3.8.4 Popups

A popup appears when an element is right-clicked in the canvas. The content of the popup is different, according to the kind of object that is clicked.

Some views define different popups, these special cases will be detailed in section 3.10 and 3.11.

A popup that always appears (but might have additional features) is the one that is generated when an author is right-clicked. The same popup is generated when the user clicks an entry in the authors' list in the info tab.

The popup has, at least, these features:

- Add/Remove an author to/from the current group of authors.
- Create a new group (adding the author to it).
- Add the author to any saved group, in which he is not already contained.
- Visualize the DBLP profile of the author.

3.9 Ebony Visualizations

Ebony allows the user to visualize the collected data in many different ways. In Ebony each view is based on an underlying type of visualization that inherits from one of the two primitive types of visualization: chart and graph.

As explained before (Section 3.6), the authors that are in the group to be visualized are totally separated from the concept of view. Each view is characterized by a set of options (that vary depending on the underlying visualization type) that link the two concepts, and that can be divided in two categories:

1. **Data Options**
   These options choose the relevant data or organize the data in the visualization.

2. **Graphical Options**
   These options adjust the graphical output of the visualization.

The main idea is to first setup the visualization by using the data options and, afterwards, based on the initial output, use the graphical options to improve the graphical aspect. The graphical improvement part is as important as the setup part, because having clean visualization facilitates the interpretation and helps to draw conclusions about the analysis that has been performed.

3.9.1 Global View Options

There are some options or option components that are common to all the views offered by Ebony:

- **Display Type Changer**
  This is a basic option component that allows the user to change visualization type.

- **Timeline**
  Through this widget the user can set the interval of time. The data that is collected from the authors of the group belongs only to this interval of time. This option is useful to limit the interval of time, or to focus only on a certain period. The time interval affects only the authors that belong to the group that is being analyzed, and it is not applied to the coauthors (it would have no sense to restrict the data of these based on a time interval that is not directly connected to them).
  Moreover, it is possible to set a different time interval for each author in the group. This feature can be used to increase the accuracy of the analysis. Quite often not all authors in a group have joined in the same year and some members might have already left the group.

- **Author Color Picker**
  By default, each time that a new author is loaded he receives a newly random generated color. For visual benefits it might be the case that the user wants to change the generated color. This component gives the opportunity to perform such action. The color changes coherently in all the views that have this author displayed.

3.9.2 Chart Visualization

Charts create visualizations that emphasize the quantitative measure of the analyzed data. They can be used to verify or discover trends in the data and it is also easy to compare different charts, having a visual feedback of the differences between the two representations.

The different types of charts offered by Ebony are discussed in Section 3.10.

3.9.2.1 Global Chart Options

Some options (or option components) are common to all types of charts:

- **Chart Series Selector**
  Ebony deals with 2D visualizations, therefore in each chart there are two axis and each of them needs a series of data. This option component gives to the user the possibility to choose the metrics and assign them to the different axis of the chart. Therefore the user is completely free to create any association. All the metrics that are identified by a series of data are presented in Section 3.5.1.

- **Data Sorting**
  Through this option component the user can decide the axis for sorting the order of the data. It is possible to choose between ascending and descending order.

- **Sorting of the Authors**
  Depending on the chart, it might be convenient to sort not only the data but also on the single author. This can be done choosing one of the single value data (listed in Section 3.5.3), or following the values that appear on the Y axis. The order of the sorting can be chosen between ascending and descending.

- **Axis Options**
  This option component gives the possibility to modify the appearance of the two axis. It is possible, for each of the axis, to set the orientation of the legend (vertical or horizontal), set the legend resolution (excluding the first and the last entry, all the other entries in the legend will be multiple of this number) and set the maximal number of entries in the legend. These last two options work only for a legend that has numeric values.

Moreover, if the X axis of a chart is composed by a data series (see Section 3.5.1) that is not numeric, by right-clicking on an entry on the axis the user can hide it from the chart. Right-clicking anywhere in the canvas opens a popup that provides the unhide functionality.

Charts cannot be rotated.

3.9.3 Graph Visualization

Graphs are useful to visualize connections (relationships), thus they can be used for analyzing the single elements of a group, in respect to the whole group or to position an element taking a group as reference.

Graphs can also be very useful for separating different entities that belong together, identify clusters, and visually separate elements that have no connections.

Ebony uses circles to represent the authors. Each circle might have a brighter portion, which represents the part of the career of the author which is not taken into consideration, due to the timeline settings. The lines that connect different circles represent the relationship among them. By clicking on an author (on his circle) it is possible to highlight the relationships.

To obtain the color of a coauthor, the colors of the authors in the analyzed group, that have a relationship with him, are interpolated (based on the strength of the connection).

Each type of graph is discussed in Section 3.11.

3.9.3.1 Global Graph Options

Each type of graph is built on the top of the same model (edges and vertices) and shares some common options (and option components) with the other graphs:

- **Graph Data Selector**
  In each graph before drawing, the data for the vertices and the edges have to be collected. Each vertex is identified by a single value data (see Section 3.5.3) and the edges are relationships data (listed in Section 3.5.2). This option component gives the possibility to set the data types for vertices and edges.

3.10 Charts

In this section, we analyze each view based on a chart type, giving information about its individual options and the main purposes for which it can be used.

3.10.1 Line Chart

The global explanations about charts in Ebony are given in Section 3.9.2.

In a line chart (Figure 3.5) each author is represented by a set of points, representing the available data, connected by lines.

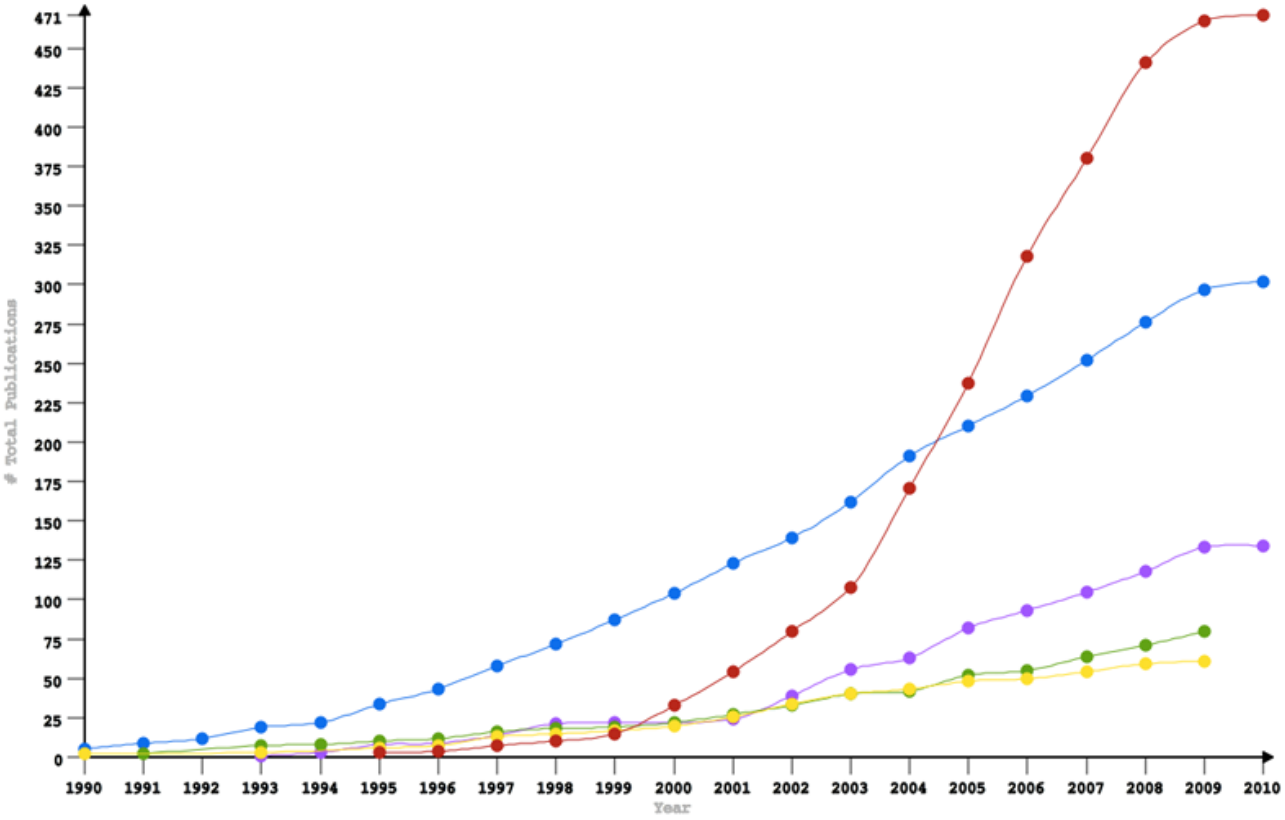


Figure 3.5. Line Chart produced by Ebony

3.10.1.1 Use of Application

Usually line charts are used to identify and visualize trends in data during a certain period of time. Thus, generally, a line chart has an X axis with numerical and chronologically ordered (ascending or descending) entries.

As each author is represented by its own set of points, it is easy to compare different authors in the group that is being visualized. The user can visually estimate the contribution of an author to a group in different periods. For example, this particular use of application can be exploited for analyzing and discovering the evolution of the entire group based on data about a single author.

3.10.1.2 Limitations

Line charts are meaningful only if the X axis is defined as a chronologically ordered (ascending or descending) and numerical series of data. If a user puts on the X axis a series of data representing not numerical data (i.e. the venues for which an author has published), the line chart becomes almost impossible to compare the different authors and their sets of data.

If multiple users share the same exact points on the graph, only one of them will be visualized (as the others will be drawn in the exact same points as before).

3.10.1.3 Options

The global options for each view are presented in Section 3.9.1, while those for charts are mentioned in Section 3.9.2.1.

Ebony defines some particular options for a line chart:

- **Year Normalizer**
  A user could be interested in analyzing the authors of a group in the first years of their career. In this case, it can be useful to compare authors having the same initial conditions (starting career) instead of constraining the time interval, which will not take into consideration the fact that some authors have already published in the past. This option component gives the opportunity to switch between the two possibilities. If the user selects the normalized version of the line chart, Ebony will calculate the following values:

  $career(author) = LastYearPublication_{author} - FirstYearPublication_{author}$
  $n = \min \forall_{author} career(author)$

  Afterwards Ebony will visualize the authors of the group taking into consideration the first n years of their career. The normalized version must only be used when the *year* metric is on the X axis.

- **Line Chart Drawing Options**
  Ebony uses a Catmull-Rom spline[1] to draw the lines between the points in the line chart. With this option component it is possible to change the precision of the drawing. Having a higher accuracy gives better graphical results, but slows down the overall performance. On the other hand, if the user wants to privilege the performance aspect, a lower drawing precision is recommended.

In the line chart it is not possible to sort the data by the Y axis. This because as each author has its own set of points and is visualized independently of all the other entities in the group, we have not found a solution for sorting the data according the Y axis. In fact, it is not possible to easily convert the usual representation to a Y axis sorted visualization.

[1] http://en.wikipedia.org/wiki/Cubic_Hermite_spline#Catmull.E2.80.93Rom_spline

3.10.2 Stacked Column Chart

The global explanations about charts in Ebony are given in Section 3.9.2.

A stacked column chart (Figure 3.6) is a type of chart in which the data for each author is represented by rectangles that map the value on the X axis to the corresponding value on the Y axis. Each column contains stacked data. The order in each column is decided by the user, through the option component to sort the data (consult Section 3.9.2.1).

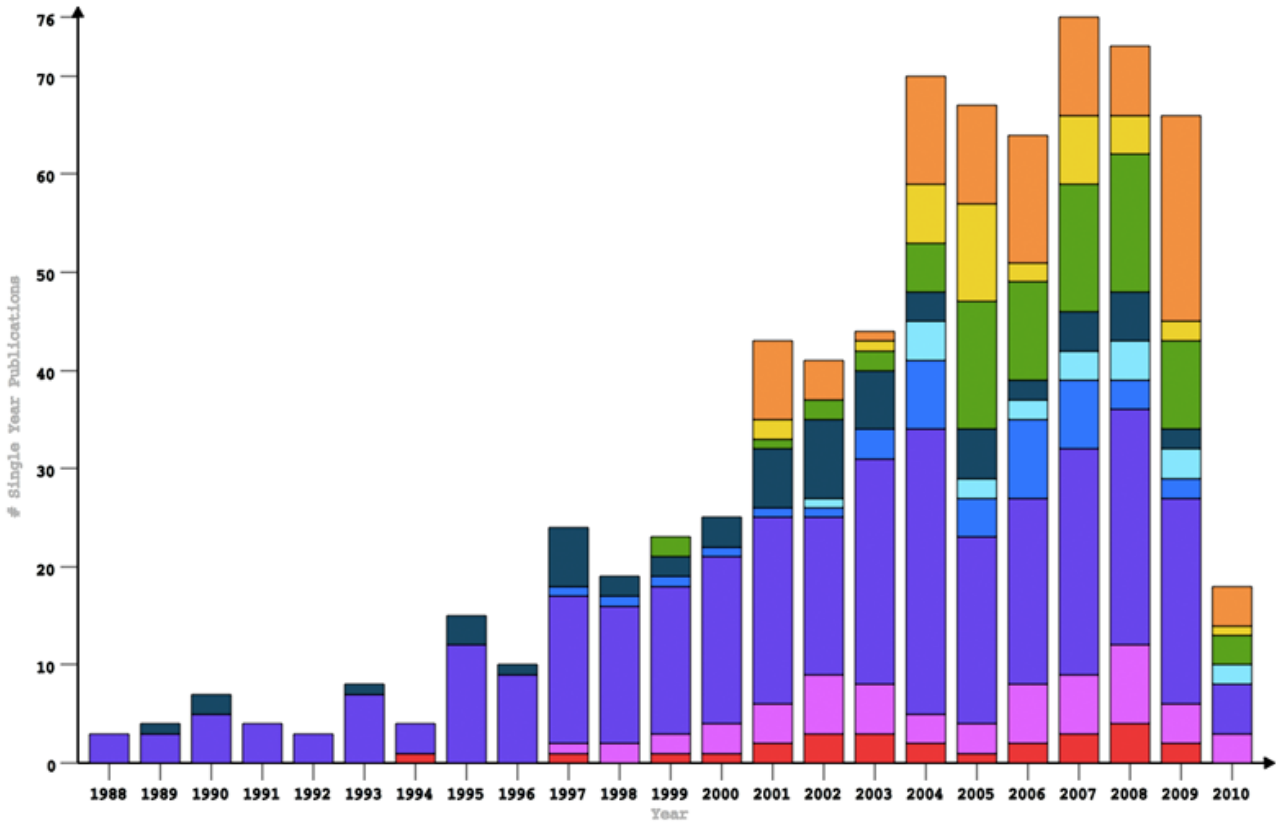


Figure 3.6. Stacked Column Chart produced by Ebony

3.10.2.1 Use of Application

Stacked column charts are well suited for analyzing the impact of an author in a specific domain (in respect to the group), represented by the X axis value of the column.

Each author is identified by a specific color and therefore it is possible to have an overview of the overall importance of the author in the analyzed data, just having a quick overview of the color distribution in the chart.

Moreover, having the data of the authors stacked one on top of the other, it is feasible to have a global overview of the group in the specific domain specified by the relationship between X and Y axis data series, without having to study the data for each author one by one.

3.10.2.2 Limitations

In stacked column charts it is not easy to quantify precisely the data belonging to an author, specially if the overall set of data that is being visualized is big. This limitation can be extended to quite any analysis done with stacked column charts, which means that studies performed with this particular type of chart are not suited for precise evaluations. An exception is given by the analysis of the group. In fact the more the analyzed entity becomes a small part of the entire chart (i.e. single author in respect to the whole group), the more the analysis becomes difficult to be done precisely.

This limitation is a direct consequence of having the data stacked one on top of the other. Therefore, it is also difficult to directly compare the authors of a group in such a chart.

3.10.2.3 Options

The global options for each view are presented in Section 3.9.1, while those for charts are mentioned in Section 3.9.2.1.

In stacked column charts the data series (see Section 3.5.1) put on the X axis is treated as being a series of strings and is never interpreted as containing numbers (but the calculations, the ordering and all the other features interpret correctly the data series). Thus, it is always possible to hide undesired columns, but the options of setting the axis resolution and the maximum entries in the axis are disabled.

Stacked column charts have not special options that modify the model or the data collected, but as a huge set of data might have a bad impact on the graphical aspect of the visualization, there are some options to modify the appearance:

- **Chart Options**
  As it is not possible to control the maximum number of entities in the legend (on the X axis) and all of them have to be visualized (except for the hidden ones), having a big data set might lead to bad visual results. Therefore this option component gives the possibility to adjust the size of the column and the space between two entries on the axis. This can help to improve the visual aspect of the chart.
  The drawback of having too many columns can be avoided by hiding the ones that are not important for that specific analysis. Otherwise the chart can be also scaled to fit the screen.

3.10.3 Stacked Bar Chart

The global explanations about charts in Ebony are given in Section 3.9.2.

The difference with usual charts is that the X axis is put vertically on the left side and the Y axis is horizontal on the bottom. In fact the axis are switched.

A stacked bar chart (Figure 3.7) is a type of chart in which the data for each author are represented by rectangles that map the value on the X axis to the corresponding value on the Y axis. Each bar contains multiple data and they are stacked one nearby the other. The order in each bar is decided by the user, trough the option component to sort the data (consult Section 3.9.2.1).

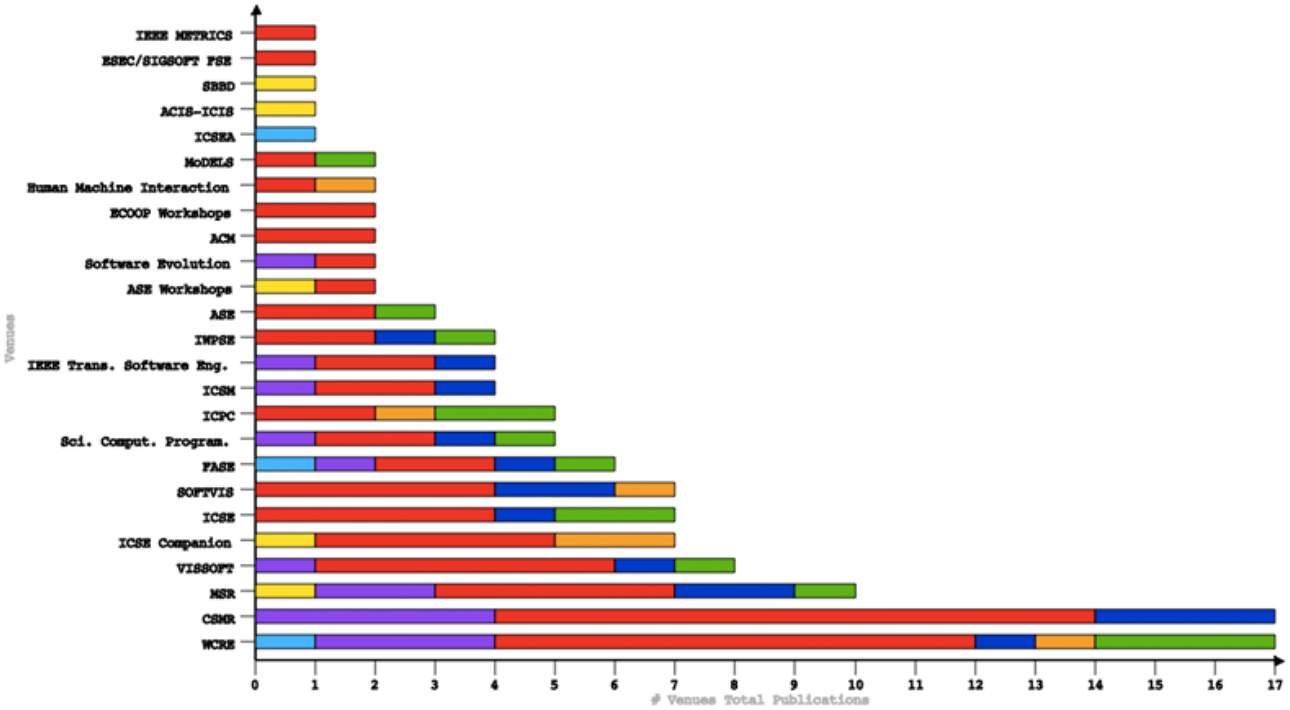


Figure 3.7. Stacked Bar Chart produced by Ebony

3.10.3.1 Stacked Bar Chart vs. Stacked Column Chart

There is no difference in the model behind stacked bar and column charts.

The only significant variation is the graphical aspect. Therefore while choosing between the two it has to be taken into consideration which graphical aspect emphasizes more the result that people has to perceive.

To obtain all the information about stacked bar chart you can consult Section 3.10.2, which presents in detail the Stacked Column Chart.

The use of application, limitations and options offered by Ebony are the same.

3.11 Graphs

In this section each view based on a graph is presented in detail, providing information concerning the individual options and the scenarios where the view should be applied.

3.11.1 Group Graph

Information regarding graphs in Ebony are provided in Section 3.9.3.

In group graph (Figure 3.8) the authors are put on the circumference of a circle to give to each one the same relative importance in the visualization. Of course the size of the circles representing the authors might differ, depending on the settings input by the user (consult Section 3.9.3.1). The order on the circumference follows the orders given by the user through the option components.

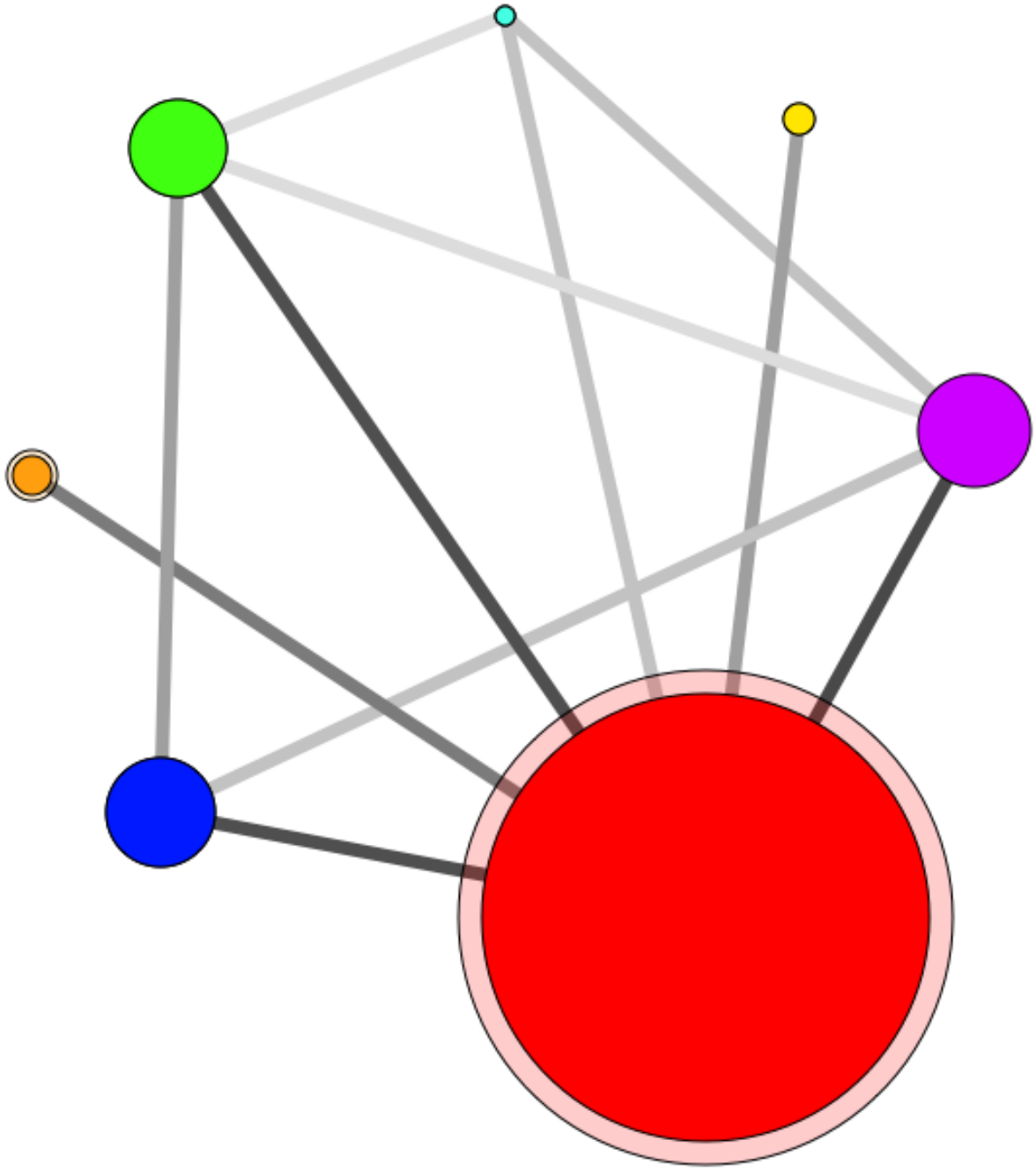


Figure 3.8. Group Graph produced by Ebony

3.11.1.1 Use of Application

This specific type of graph has been created to help the user to identify and discover relationships in a group of authors. The idea is to be able to comment about the cohesion of a group and identify possible connections (strong and weak ones).

A very concrete application of this view is the identification of a group of authors that is suitable to judge a work or a thesis. In this case the group should have the least possible number of connections, because this would increase the possibility that the members do not know each other personally and therefore no one could suspect of having the problem of favoritisms.

3.11.1.2 Limitations

The only real limitation of this view is the fact that it is difficult to estimate the real value of the connection between two authors. A graphical estimation can be performed (which is sufficient in most of the cases), but it is impossible to have the precise numeric value.

Except for this drawback, this view accomplishes all the tasks for which it has been created. The only further remark that can be done is that the usage of this particular view is restricted to very precise cases, that are however really important in a scientific environment.

3.11.1.3 Options

The global options for each view are presented in Section 3.9.1, while those for graphs are mentioned in Section 3.9.3.1.

There are special options for the Group Graph that are used to customize the graphical aspect of the view:

- **Authors Sorter**
  It can be useful to sort the authors on the circumference to have different orders and give priority to some kind of authors. The authors can be sorted using single value data (see 3.5.3) or by name.
- **Relationship Emphasizer**
  It is possible to change the graphical aspect of the lines that represent the connections between the authors. The two possible solutions are:
  1. Using a different color gradient for stronger/weaker relationships
  2. Painting larger lines for stronger connections. For this option there is also the possibility to set the maximum weight of the line
- **Graph Options**
  This option component gives the possibility to set the minimum radius of the circles that represent an author and to dispose the minimum distance between two vertex in the graph. In this case this minimum distance corresponds to the minimum space that two authors should have between them on the circumference.

3.11.2 Relationships Graph

The graph produced by this view is composed mainly by two parts: an inner part that is a Group Graph (for details about this view consult Section 3.11.1) and an external circumference on which all the coauthors of the authors in the analyzed group are placed.

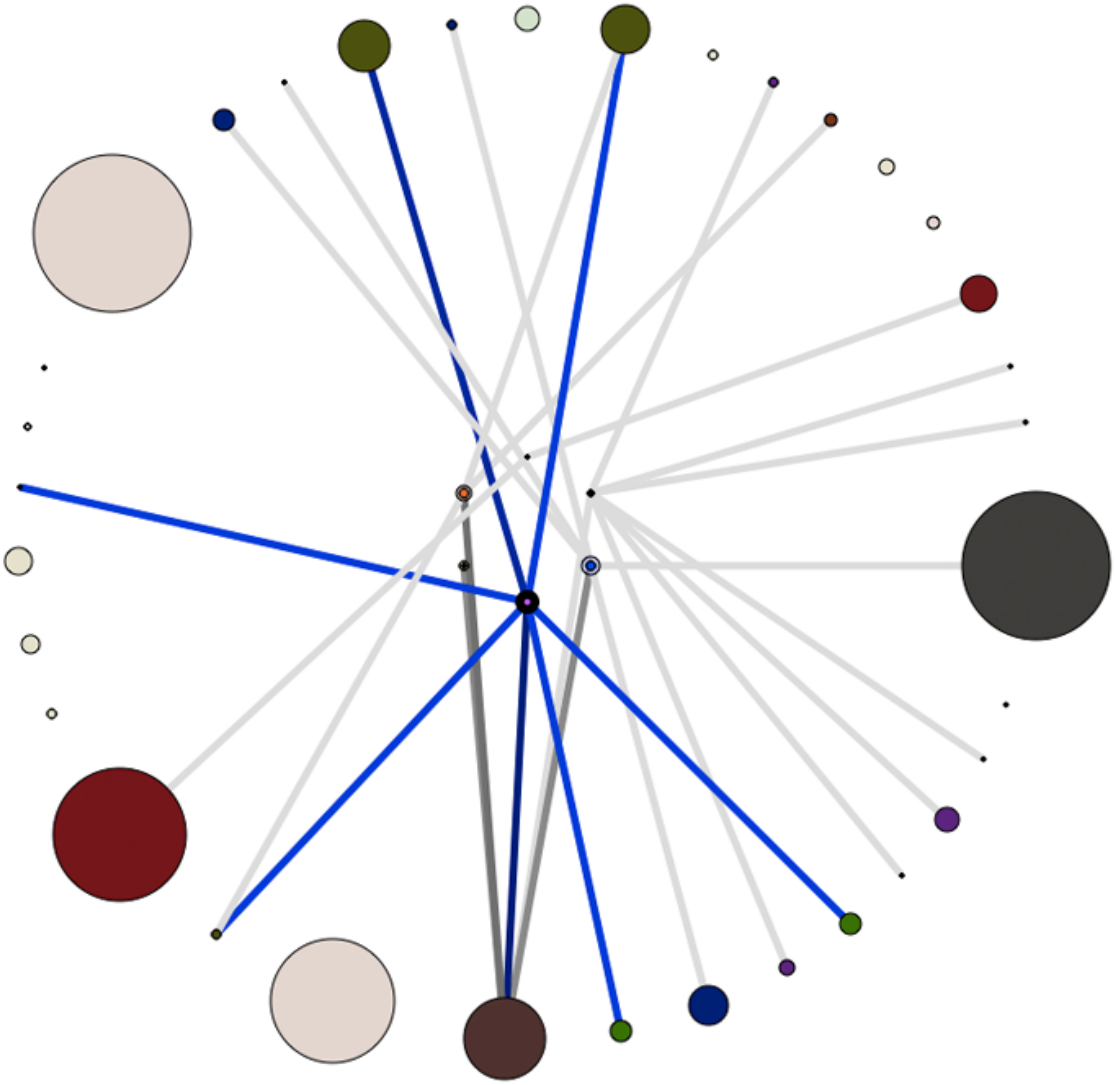


Figure 3.9. Relationships Graph produced by Ebony

3.11.2.1 Use of Application

This view should be used to have an overview of all the connections that a group has and is perfectly meaningful also if the group is composed by a single author. In this case the evolution of the career of the author can be estimated, by playing with the timeline and isolating different periods.

It can be of major importance identifying the quantity of relationships of a group, as it can be used as a measure for estimating the importance in the scientific world of the group itself. Moreover analyzing the authors that have a connection to the group it is possible to discover a relationship to a specific research area or research group. As the connections are of different strength also the robustness of the different affiliations can be estimated.

This view can also be used to discern the relationships that are common for different authors in the group. Such an analysis can be helpful to have more data to judge more the connection between the authors in the group.

3.11.2.2 Limitations

The view does not give the exact number of overall relationships and also the exact value of each single connection.

We think that a visual analysis is sufficient to extract the necessary data that this view has to provide to the user for absolving the tasks for which it has been designed.

When the group that is being analyzed has many relationships, it might happen that the visualization becomes less clear and harder to analyze.

Sometimes it can be difficult to discover the common relationships of many authors, if the number of total coauthors is significant.

In this case it can be really useful to use very different colors for the authors in the group to obtain, for coauthors with multiple connections with the group, interpolated colors which can easily be distinguished.

3.11.2.3 Options

The global options for each view are presented in Section 3.9.1, while those for graphs are mentioned in Section 3.9.3.1.

The graphical aspect of this view can be customized using the following options (and option components):

- **Authors Sorter**
  With many coauthors it can be convenient to sort the authors according to some parameters to easily be able to focus only on the important ones. The authors can be sorted using single value data (see 3.5.3) or by name.
- **Relationship Emphasizer**
  It is possible to change the graphical aspect of the lines that represent the connections between the authors. The two possible solutions are:
  1. Using a different color gradient for stronger/weaker relationships
  2. Painting larger lines for stronger connections. For this option there is also the possibility to set the maximum weight of the line
- **Graph Options**
  This option component gives the possibility to set the minimum radius of the circles that represent an author and to give the minimum distance between two vertex in the graph. In this case this minimum distance corresponds to the minimum space that two authors should have between them on the circumference. This option is used on both, inner and outer, circumferences.

3.11.3 MDS Graph

This particular type of graph (Figure 3.10) uses the set of related statistical techniques called Multi Dimensional Scaling (MDS) to visualize an unconnected graph that tries to preserve in 2D the similarities (and dissimilarities) between the vectors of n components (previously constructed) that contain information about the author in the group.

The next sections will give more information about MDS and explain how it is used in Ebony.

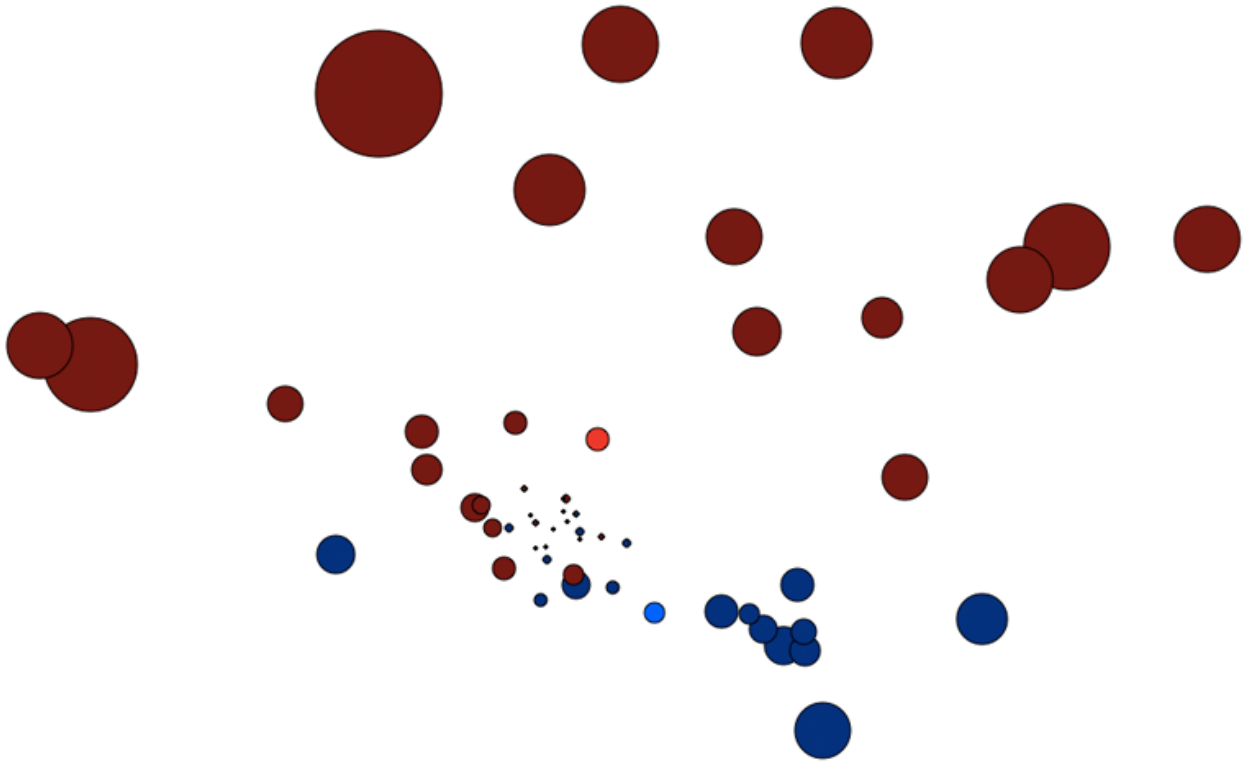


Figure 3.10. MDS Graph produced by Ebony

3.11.3.1 MDS - Multi Dimensional Scaling

MDS[2] is a set of related statistical techniques that are used in information visualization for discover, study or analyze similarities (and dissimilarities) in data.

The usage of MDS and the problem that is found and that has to be solved can be summarized in these steps:

1. Collect the K objects that represent the data set that is going to be analyzed.

2. Apply a distance function to the objects, to obtain the distance (the dissimilarity) between each pair of entries.

3. Construct a dissimilarity matrix Δ of size $K \times K$ in which all the distances are stored.
   Therefore a generic entry in the matrix is: $\delta_{i,j}$ = distance between i^{th} and j^{th} objects.

4. Choose the number of dimensions N to which the procedure is going to scale the original data set.
   Higher dimensions give better precision but create difficulties while interpreting the results. In most cases N is chosen to be 2 or 3, to be able to represent the data using 2 or 3D visualizations.

5. The goal of MDS is now to find K vectors $x_1, \ldots, x_K \in \mathbb{R}^N$ such that: $\|x_i - x_j\| \approx \delta_{i,j}$ for all $i, j \in K$ where $\|\cdot\|$ is a a vector norm (usually the Euclidian distance is used).

6. The last step deals with the decision on how to solve the problem and find the vectors. Usually MDS is rephrased as an optimization problem where the set of vectors x_1 to x_K is the minimizer of some cost function.

[2] http://en.wikipedia.org/wiki/Multidimensional_scaling

3.11.3.2 MDS in Ebony

The idea of implementing a graph that uses MDS for placing the authors is completely experimental, and based only on some readings [7] that commented about the results achieved applying this technique in other domains.

Ebony applies the technique of MDS to obtain the MDS Graph. First of all the user has to chose two metrics to construct for each author a vector that contains information (feature vector). The author is going to be represented in the data set as this vector.

The metrics are two data series (for more information consult Section 3.5.1).

The first series is considered to contain the keys that are used to obtain for all authors the entries that have to be present in the vector.

The second series contains the values that are going to be inserted in the vector and considered for calculating the differences between the vectors. This series can contain only numeric values.

Obviously as this is a one-to-one (key to value) relationship, if the two data series are not chosen correctly the result is something completely unreliable.

After the creation of the vector the dissimilarity matrix is obtained calculating the distance between the vectors using the method chosen by the user (see the options in Section 3.11.3.6).

Afterwards the matrix is passed to an external implementation of MDS [10] that uses the simulated annealing[3] to find the solution. As required by Ebony this implementation scales the initial data set to 2D vectors.

It has to be noticed that running the simulation multiple times does not guarantee that the obtained graph is always the same although the initial vectors for the authors are identical. This is a consequence of the fact that the solution method is an heuristic (and therefore is not guaranteed to give an exact solution) that involves some randomness and that applying some transformations to the vectors does not change the pairwise distances, which is fine for some types of norm (for example the Euclidean norm that is used).

3.11.3.3 Distance Algorithms

The distance algorithm is used to calculate the dissimilarity between the feature vectors of the authors and therefore create the dissimilarity matrix. Currently the following algorithms are available in Ebony:

- **Euclidean Distance**
  This algorithm simply computes for each pair of vectors $\mathbf{p}$ and $\mathbf{q}$ of size K their distance using the following formula: $\mathrm{d}(\mathbf{p},\mathbf{q}) = \sum_{i=1}^{K}(p_i - q_i)^2$.

- **Cosine Similarity**
  The cosine similarity simply calculates the cosine of the angle between two vectors $\mathbf{p}$ and $\mathbf{q}$ of size K and uses this measure for calculating the similarity: $\text{similarity}(\mathbf{p},\mathbf{q}) = \cos(\theta) = \frac{\boldsymbol{p}\cdot\boldsymbol{q}}{\|\boldsymbol{p}\|\|\boldsymbol{q}\|} = \frac{\sum_{i=1}^{K} p_i q_i}{\sqrt[2]{\sum_{i=1}^{K} p_i^2}\sqrt[2]{\sum_{i=1}^{K} q_i^2}}$. As we need a measure of dissimilarity it has been chosen to calculate this as: $\text{dissimilarity}(\mathbf{p},\mathbf{q}) = \frac{1}{similarity(\boldsymbol{p},\boldsymbol{q})}$.

3.11.3.4 Use of Application

The MDS Graph can be used to estimate the distances in a group, based on the chosen metrics. Changing the metrics can help to analyze the group using different perspectives and collecting all the data can give the possibility to own a complete data set, useful to make strong assessments about the group and to perform a complete and well-documented study.

An interesting study that can be performed on a group is observing the evolution, using the timeline, of the cohesion of the group (meaning the increasing or decreasing similarity) and observe the impact of this change on the publications of the group. Obviously not all the groups or the single situations are appropriate for such an analysis.

This view can also be used to judge the distance between a single author and a group known to be strongly related and also evaluate the similarity between two groups.

Another possibility given by this view is to search authors that work on a specific topic, seeking for authors that have a strong similarity, and therefore a minimal distance, to a group of authors known to work on that topic.

[3] http://en.wikipedia.org/wiki/Simulated_annealing

3.11.3.5 Limitations

The main limitation of the MDS Graph is that the obtained visualization is never really guaranteed to be really meaningful; it all depends on the information of the authors in the analyzed group.

For example an author that has a strong dissimilarity with all the other members is going to deform the visualization too much and if all the authors have a minimal distance this will lead to having all the authors near a point (which can be actually fine for some analysis).

Having too many or too few authors displayed is going to make the graph practically unanalyzable and therefore useless.

A final limitation that can be remarked is the fact that the information about the authors that are collected in the vectors are independent of the group visualized. This can lead to situations in which an author has connections with all the other members, but has also so many other connections (that the other authors in the group do not have) that he will be put far away from the others. In out view this is the correct behavior but sometimes this might not be the desired result.

3.11.3.6 Options

The global options for each view are presented in Section 3.9.1, while those for graphs are mentioned in Section 3.9.3.1.

There are other several options used to control the simulation and to change the data taken into consideration for the visualization:

- **Simulation Controller**
  Using this option component it is possible to set the maximal number of iterations that the algorithm for solving the MDS problem can perform. Reached the maximum number of iterations the algorithm stops and returns the actual solution. Obviously a very small number leads to very inaccurate solutions, but on the other hand speeds up the visualization process.

  As the space for the visualization is limited, it can be useful to scale the circles representing the authors to a size that makes the graph more understandable. Thus the maximum circle radius is a really important option, because having circles that are too big leads to situations in which all the circles seem to be really near but the centers are not and thus, in reality, they are quite distant. Therefore it is important to find a tradeoff to emphasize the importance of an author (having quite big circles) without damaging the shape of the graph.

  It has been chosen that only some options are able to restart the simulation automatically. In most of the cases after changing some options that modify the data set, it is strongly recommended to restart the simulation manually.

- **Author Vector Constructor**
  This option component is used to set up the data series used for creating the feature vectors that represent the authors. This key-value mapping should be meaningful, otherwise the graph produced will not be trustable. A deeper explanation on how these data series are used is given in Section 3.11.3.2. The simulation should be restarted after changing these options.

- **Distance Algorithm Chooser**
  Using this widget it is possible to switch between the different distance algorithms presented in Section 3.11.3.3. As the dissimilarity matrix is going to change it is highly recommended to restart the simulation after changing this option.

- **Load Coauthors Option**
  Sometimes analyzing only the authors in the group is not sufficient for producing a very meaningful graph, therefore it might be a good idea to include also the coauthors of each member of the studied group. The visualized authors will maybe be a lot (maybe too many), but sometimes having more information and having authors that should be close to the studied ones gives a better, more understandable, result.
  Changing this option automatically restarts the simulation.

By right-clicking on an author it is possible to hide him. All the hidden authors are not taken into consideration while performing the computations. There is also the unhide functionality.

3.11.4 Force-Directed Graph

The force-directed graph (FDG, Figure 3.11) is a view that uses a force-directed algorithm to position the authors in the 2D space.

The main idea is to characterize each author by a certain metric, give to each relationship a value and use an algorithm that takes into consideration the forces of attraction and repulsion between the different nodes in the graph to produce a visualization that reflects the input data.

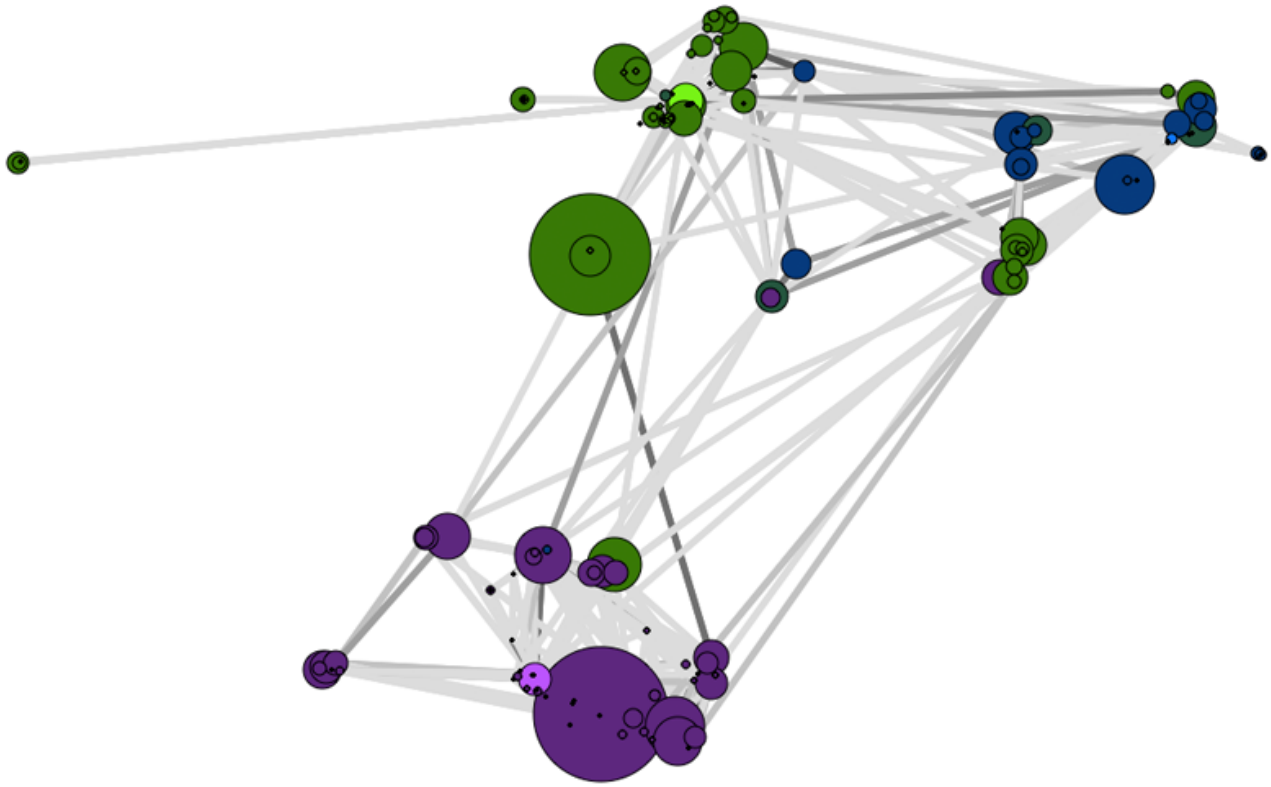


Figure 3.11. Force-Directed Graph produced by Ebony

3.11.4.1 Force-Directed Algorithms

The primary concept behind force-directed algorithms[4] is to position the nodes of a graph in 2D or 3D so that all the edges are of equal length and there are as few crossing edges as possible.

Usually this goal is achieved by assigning forces to all the nodes and the edges in the graph. Normally the edges are treated as springs, therefore they attract the nodes that are connected to the edge. On the other hand the nodes are interpreted as electrically charged particles and therefore they repeal each other.

The simulation terminates if the graph enters in a state of equilibrium (the positions do not change anymore from one iteration to the other) or if the maximum number of iterations is reached.

Usually the graphs drawn with these algorithms look good and the main advantage is that we can take advantage of very well known and surely correct physical laws to solve a complex problem like graph drawing.

A further remark is that there are many variations of these algorithms and they can be come also quite complex. There is no real method to predict the drawing that will be produced by these algorithms, therefore choosing between them is a matter of comparing the different graphs and select the best for the purposes it has been created for.

[4] http://en.wikipedia.org/wiki/Force-based_algorithms_(graph_drawing)

3.11.4.2 FDG in Ebony

The decision of inserting a graph that uses a force-directed algorithm has been taken when the Ebony project started. We were curious to see which benefits could a graph, produced by these algorithms, give to the analysis of a group of authors taken from the DBLP database.

After some initial research [1] and implementation of some force-directed methods that were not completely satisfying our wishes, we encountered another really interesting reading [9] that gave us supplementary information about graph visualization and force directed algorithms.

After having analyzed the mentioned publication we found an implementation of the force-directed mentioned realized by the same author [8]. As there were examples of application comparable to our case of application (the authors of the DBLP database), we decided to use it and the tests executed were successful.

Linloglayout's algorithm is based on three main forces: the attraction force created by the edges between nodes, the repulsion between the nodes in the graph and the gravity force.

We developed a converter used to adapt our data to the ones needed by lingloglayout and we use the result produced by the library to draw the graph.

3.11.4.3 MDS Graph vs FDG

Although the two techniques for realizing the graphs are completely different, it can appear that the two visualizations have the same purposes.

It is true that they are mainly used to identify strong similarities (relationships) between authors and discover sub groups (clusters) in the analyzed set of authors. However, they use different data to reach the goal.

The MDS graph (consult Section 3.11.3) uses a feature vector which is filled with the global data about the author; in other words the calculations do not change if we add new authors to the group. The distance between two authors remain always the same (keeping the time window in which the authors are analyzed fixed). This does not mean that the resulting graph is not going to change but internally in the dissimilarity matrix the values between two authors A an B is always going to be the same.

The FDG uses the information on the relationships only with the other visualized authors (and discards all the others), therefore changing the authors that are taken into consideration adds or removes information that are used to realize the graph.

A simplified explanation can be that the MDS graph uses global information, without really taking into consideration the composition of the group of visualized authors, and the FDG uses only local information that are based on the visualized authors.

3.11.4.4 Use of Application

The FDG can be used to identify in the group of authors how the relationships make that some authors are closer.

As this graph is strongly based on the relationships between the authors, it is easy to identify clusters of authors, which usually are strongly connected, maybe as they are members of the same research group or they work on the same domain.

Discovering clusters of authors can be not only a goal of an analysis but also a starting point, because it is then possible to analyze the specific clusters. Many iterations of these studies can be performed, focusing always on smaller groups, being able to identify sub-domains and connections that are not visible at a first glance.

3.11.4.5 Limitations

The main disadvantage of this view is directly connected with the usage of force-directed method. Normally each force-directed algorithm has complexity $O(V^3)$ where V is the number of nodes in the graph. Therefore with a considerable number of nodes the running time of the algorithm becomes really elevate.

Having many authors visualized can produce graphs that are difficult to interpret. The parameters used to adjust the algorithm play an important role, as small changes in the value can correspond to huge changes in the final result.

3.11.4.6 Options

The global options for each view are presented in Section 3.9.1, while those for graphs are mentioned in Section 3.9.3.1.

There are other several options used to control the simulation, change the composition of the group of authors that are going to be visualized and adjust the parameters used in the force-directed algorithm:

- **Simulation Controller**
  Using this option component it is possible to set the maximal number of iterations after which the force-directed algorithm has to stop. Doing few iterations leads to an incomplete result as the algorithm has not enough time to find the point of equilibrium.

  As the space for the visualization is limited, it can be useful to scale the circles representing the authors to a size that makes the graph more understandable. Thus the maximum circle radius is a really important option, because having circles that are too big leads to situations in which all the circles seem to be really near but in reality the centers (the distance has to be measured from center to center) are distant. Therefore it is important to find a tradeoff to emphasize the importance of an author (having quite big circles) without damaging the shape of the graph.

  It has been chosen that only some options are able to restart the simulation automatically. In most of the cases after changing some options that modify the data set it is strongly recommended to restart the simulation manually.

- **FDG Options**
  This option component can be used to set the parameters used by the force-directed algorithm:

  1. Repulsion Exponent: Used to give more power to the repulsion force.
  2. Attraction Exponent: Used to modify the importance of the attraction force between nodes.
  3. Gravity Factor: Used to modify the importance given to the gravity in the calculations.

  It has to be remarked that the attraction exponent has always to be greater than the repulsion exponent and the gravity factor should be usually a number between 0 and 1.

  It is also possible to decide if the edges between the nodes have to be hidden or not.
  The simulation should be restarted after changing these options.

- **Relationship Emphasizer**
  It is possible to change the graphical aspect of the lines that represent the connections between the authors. The two possible solutions are:

  1. Using a different color gradient for stronger/weaker relationships.
  2. Painting larger lines for stronger connections. For this option there is also the possibility to set the maximum weight of the line.

- **Load Coauthors Option**
  It can be decided if the coauthors have to be included in the graph or not. This choice is useful, depending on the analysis as sometimes including the coauthors helps to better outline the clusters of authors. Changing this option automatically restarts the simulation.

By right-clicking on an author it is possible to hide him. All the hidden authors are not taken into consideration while performing the computations. There is also the unhide functionality.

3.12 Server-Side

The server-side of Ebony is implemented using Java Servlets and is composed by four key elements:

- The *requester processor*, which takes the requests.
- The *database handler*, which retrieves the data from the database.
- The *cache*, which contains frequently used data.
- The *collector*, which contains a queue of authors that have to be loaded and automatically retrieves data and puts them in the cache.

A key concept is that an author is said to be *complete* when all the information about him and his coauthors have been retrieved from the database. Not all the views use the *complete* version of an author, therefore, when sending a request to the server, the client specifies if there is the need to have the authors loaded completely or not.

If the data available in the cache are already complete, they are returned as they are, without caring if the client needs them to be complete or not. This because if the client will need them complete for another analysis it will not be necessary to contact the server again.

When a request is sent to the server the following steps are performed:

1. The request is processed by the request processor which checks if the data requested is available in the cache. If the cache does not contain the desired information, the database handler takes care of retrieving the necessary information. Otherwise the data is retrieved directly from the cache.
2. If the authors that have been taken from the database are not *complete* they are added to queue of the collector.
3. The data is returned to the client.

It has to be remarked that when a user logs in, all the saved views and groups are loaded and returned to the client. All the authors in the groups that are not *complete* are added to the collector.

3.12.1 Server-side Cache and Data Collector

Two elements that needs some further explanation are the *cache* and the *data collector*.

The cache implemented in the server side is similar to the one present in the client-side (see Section 3.4.1), but it is able to contain a bigger number of entries. Moreover there is also another cache in which all the publications are stored, thus loading authors becomes much more rapid.

The data collector is used to anticipate the requests of the user. When a registered user logs in it is possible that it will use the saved groups, therefore all authors are added to the collector's queue, ready to be loaded. The same reasoning can be done for all the authors that are requested during the analysis. It is highly probable that it will be necessary to load them completely.

Loading authors completely (loading his information and also the data regarding his coauthors) might take a long time, therefore being able to do it in advance is a big improvement in terms of performance and usability.

The more coauthors an author have the more it is expensive to load his *complete* version, therefore the collector orders the authors, giving priority to the ones that have many coauthors. So it is probable that authors that are more expensive to load will be already ready to be sent to the client when a request will be performed.

Chapter 4

Validation

In this section, we want to validate the software by presenting a real case study, in which we conduct all the analysis required to achieve meaningful conclusions.

4.1 Case Study

We decided to analyze the evolution of the Faculty of Informatics at the University of Lugano (Switzerland)[1]. The faculty is born in 2004 and has expanded significantly in the following years, counting in 2010 more than 20 professors.

All the professors' information is taken into consideration only from the year in which they joined the faculty. Moreover the single authors will never be mentioned, because this analysis wants to study the faculty as a single entity, without giving importance to single professors.

This study aims to:

1. Discover the impact of hiring new professors on the productivity of the faculty.
2. Learn about the connections among the professors in the faculty.
3. Comment about the different domains in which the professors work. In particular estimate the presence of different domains, and whether there is predominance of a specific research area.

4.2 Preliminary Work

For executing the analysis first of all some preliminary works had to be done:

1. Collect all the information about the professors that worked at the faculty. Finding the year when they started (and ended if they leaved).
2. Create the group of authors, changing for each of them the timeline settings (according to the collected data) and setting up a coloring schema that makes it easy to distinguish between the authors in the group and also between the different coauthors.
3. Set up all the views necessary for this analysis.

In the next sections we are going to show the obtained visualizations and comment them, remarking the elements that are emphasized by the drawing. Afterwards, summarizing the obtained facts, we hope to have reached the goals we explained in the previous section.

[1] http://www.inf.usi.ch/

4.3 Global Overview

For having a global overview we decided to visualize the group using a line chart (Figure 4.1), putting on the X axis the years of publication and on the Y axis the total number of publications.

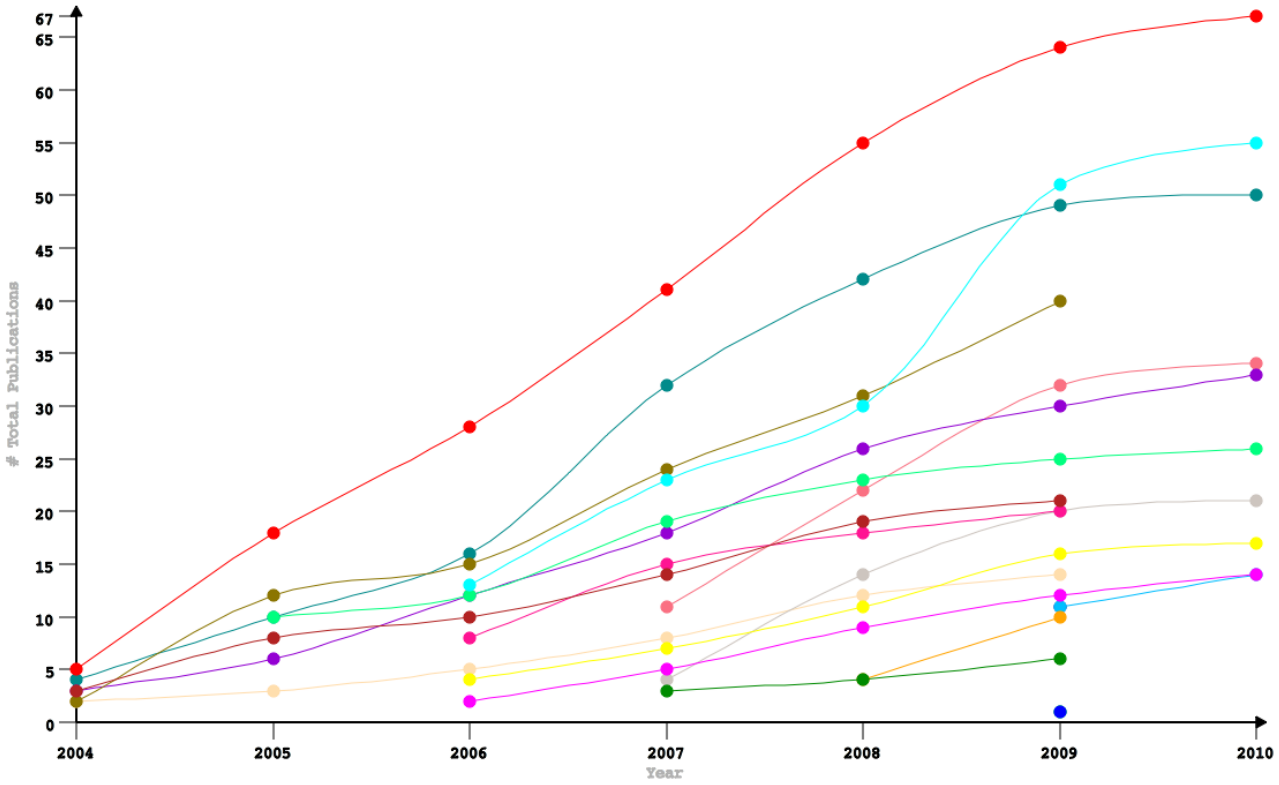


Figure 4.1. Overview of the group composed by the professors of the faculty

As we can notice the group is pretty heterogenous, the professors joined the faculty in different years and the number of publications varies a lot. Of course the number of publications is only related to the period in which the professors have worked at the faculty, therefore it does not mean that an author which results having more publications than another in this graph has more in absolute.

This is an important remark as many authors that have joined the faculty in the last years (2009 and 2010) have a small number of publications on the graph.

Summarizing we can see that the professors have always increased, from year to year, their number of publications. This means that they actively contributed to the development of the faculty with new publications.

This fact underlines how the faculty of informatics of the University of Lugano, which is quite young, is very dynamic and constantly growing.

4.4 Analysis of the Publications by Venue

An interesting analysis that can be performed over a group of highly specialized people (like the professors of a university), is to determine the venues that are the ones for which the professors have published most.

As the venues for which the professors have published are really a lot we have chosen to take into consideration only the ones for which at least 4 publications have been published.

The result can be seen in Figure 4.2.

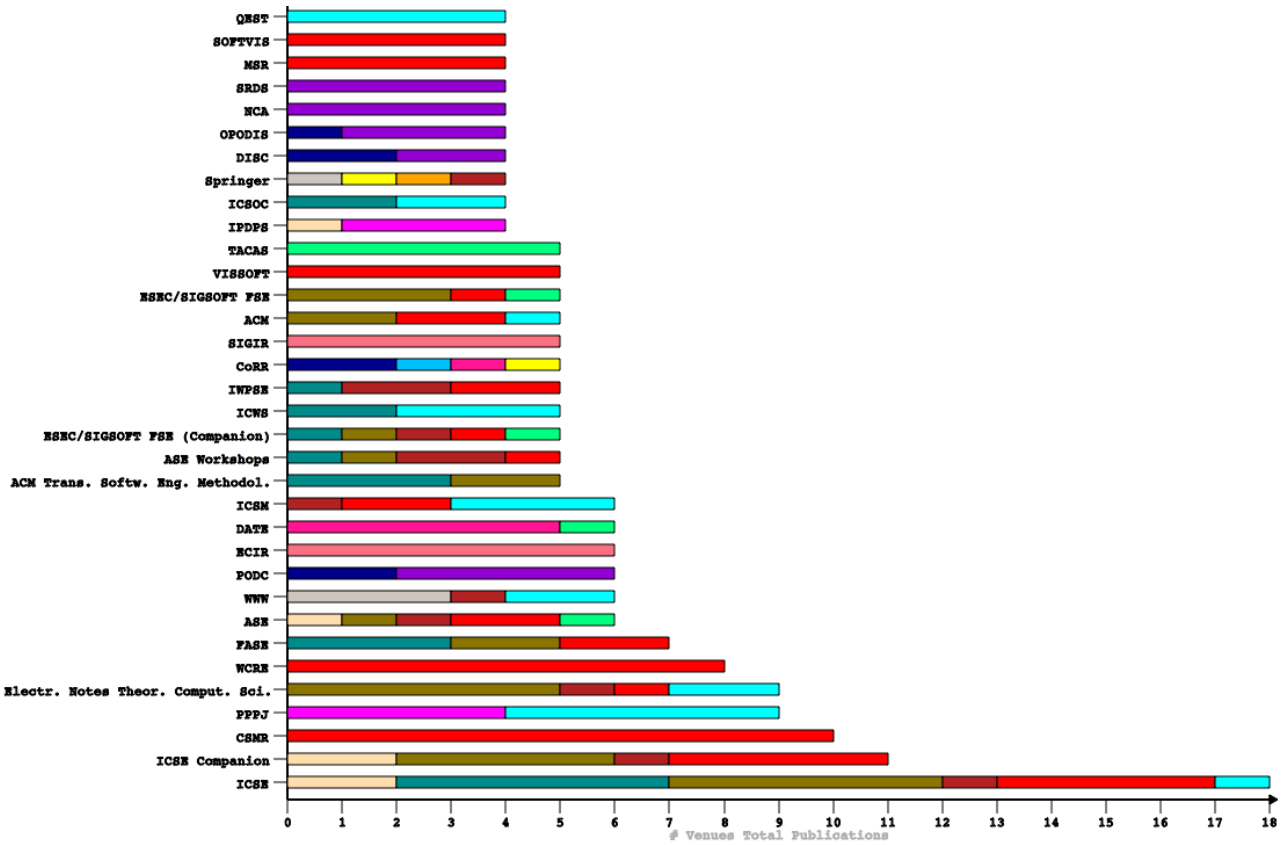


Figure 4.2. Venues for which the professors have published more

Using this chart we can identify the venues for which the professors have published more. This study can be very useful for an internal analysis of the faculty and also to identify the domains in which the faculty is more active.

We think that the professors, that have published for the same venues, work on domains that are related. Another fact that supports this thesis is that, taking a close look to the chart, many different venues for which always the same authors have published can be identified.

4.5 Analysis by Single Year

To study the relationship between the employment of the professors and the productivity of the faculty we have chosen to represent in a stacked column chart the single year publications in each year (Figure 4.3).

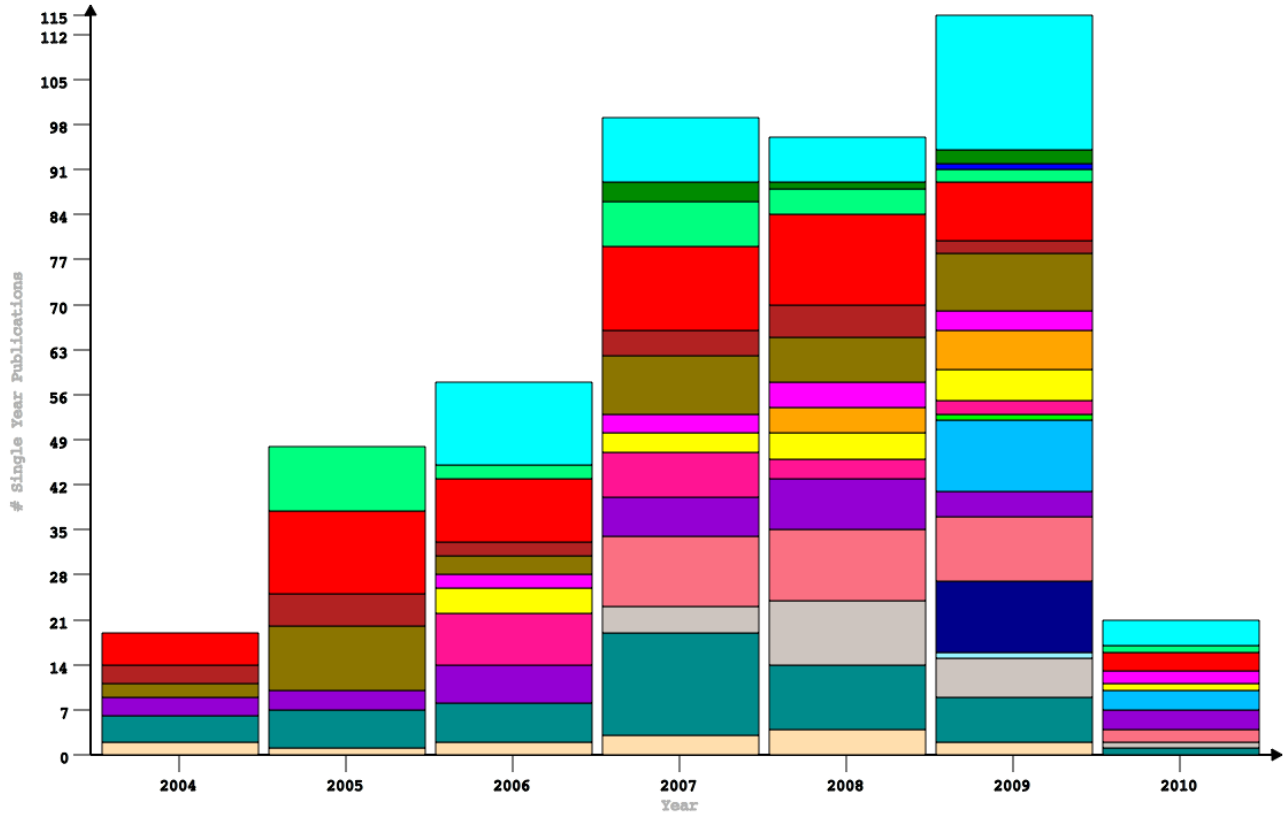


Figure 4.3. Publications done each year by the professors of the faculty

A first remark that has to be done is that as the current year (2010) is not finished yet and DBLP is not always updated immediately, the year 2010 is not really relevant for this analysis, but it has been included in the chart for completeness (as some information is already available).

We can observe that the productivity of the faculty has increased a lot over the years. Observing the different columns we can also identify the moment in which the faculty hired the new professors and notice how the important increases in the number of publications correspond to the years in which there were new employments.

On the other hand we can notice that in the years in which there were no employments (i.e. 2008 to 2009) the total number of publications of the professors remained constant.

This means that the overall productivity seems to remain constant and therefore to increase the productivity it seems necessary to hire new professors. This can be reasonable if we assume that all the professors always work at the maximum of their potential.

A direct consequence of these two considerations is that the productivity of each professor stays more or less constant regardless on new employments. This means also that the productivity of different professors are not related, which could be a symptom of weak collaboration between the professors.

4.6 Analysis of the Relationships

We decided also to analyze the inner structure of the faculty by analyzing the relationships between the professors. For the visual result consult Figure 4.4.

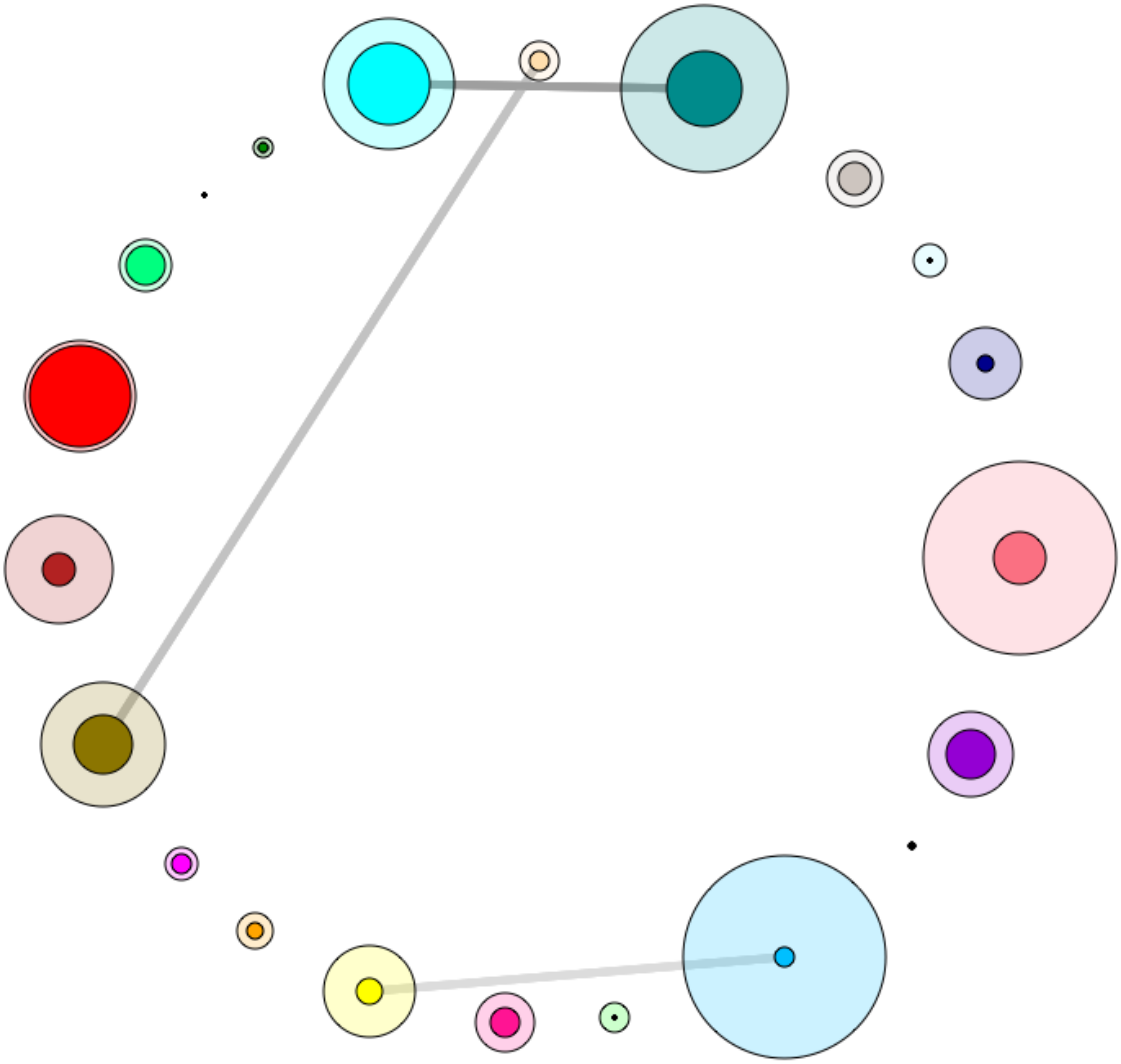


Figure 4.4. Graph of the group of professors present at the faculty

This graph confirms the idea that the professors do not collaborate much (theory explained in Section 4.5). In fact there are very few edges in the graph.

First of all we can remark the fact that the circles representing the authors are of different size, which means that at the faculty are present professors which are active since many years and others that probably have just started their career. Therefore the group of professors is quite heterogenous.

Moreover we can also notice that there are some professors that have published a lot (look at the outer semi-transparent circle), but the publications done while being professors at the University of Lugano are only a small fraction (the inner, full colored, circle) of the overall number.

4.7 Analysis of the Research Areas

We also want to determine if there is a predominant domain on which the faculty is focused on. We decided to first use a MDS Graph (see Section 3.11.3 for reaching this goal (Figure 4.5.

We have filled the feature vectors of the professors with the information about the venues for which each professor has published and decided to not include the coauthors in the analysis. We suppose that if professors have published for the same venues they most probably work on the same domain.

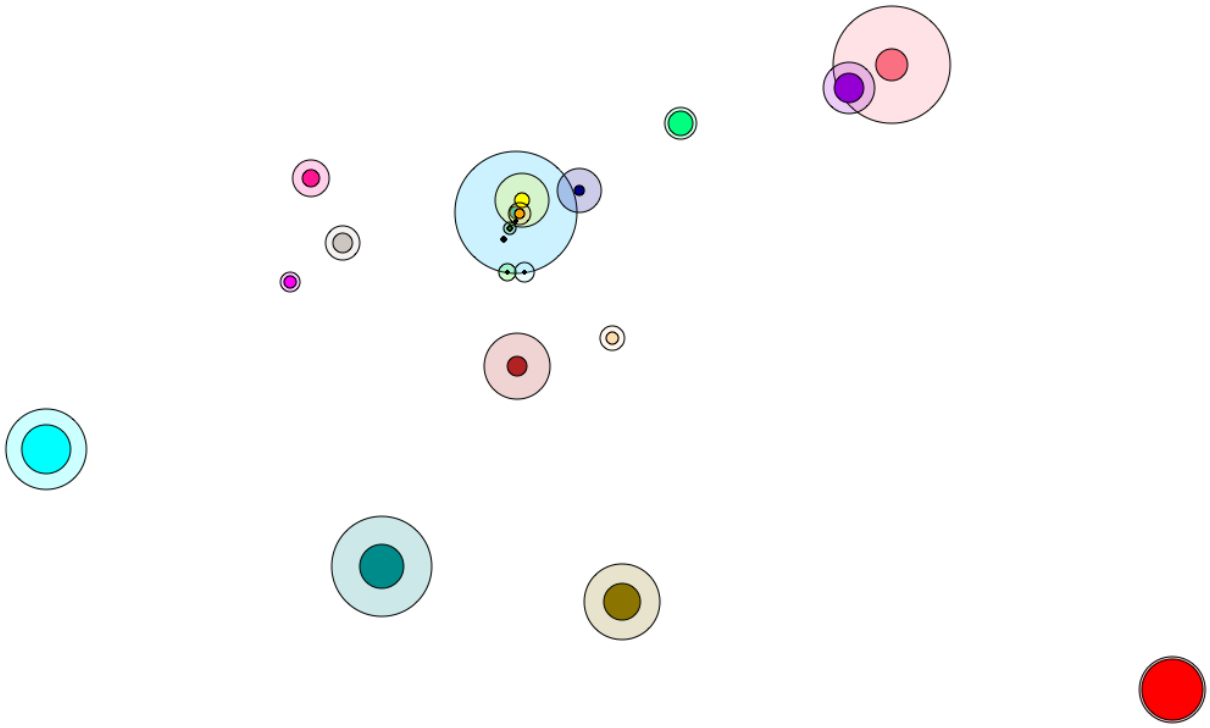


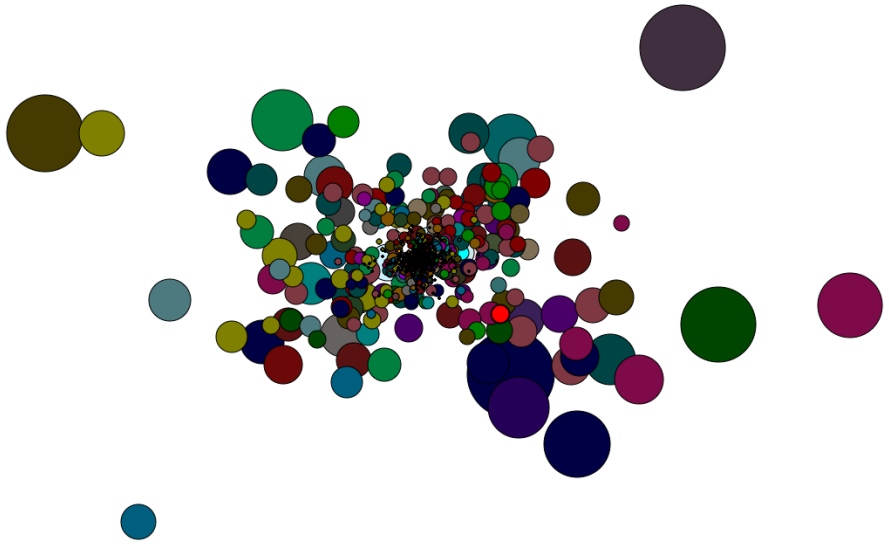
Figure 4.5. MDS Graph based on the information about the venues

The result confirms that there are some professors working on related domains, as some circles representing the authors are close.

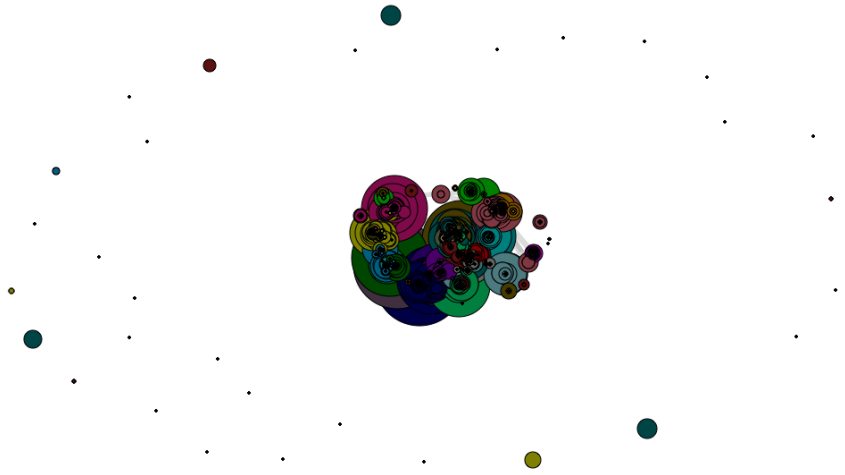
However, it is not easy to really identify clusters in the graph, therefore there are not few, really distinct, domains, but most probably the professors work in many different research areas and some of them are someway related.

4.7.1 Analysis considering Coauthors

We decided to remake the analysis using the same MDS Graph as before and the force-directed graph (see Section 3.11.4) this time including in both the coauthors of the professors present at the faculty. The graphs can be observed in Figure 4.6.



(a) MDS Graph



(b) Force-directed Graph

Figure 4.6. MDS Graph and FDG of the professors and their coauthors

We can notice that both graphs seem to put the authors all very close and they form a big cloud. This is symptomatic of the fact that all the domains of these authors are connected. We know that having a sufficient large number of domains these are all connected and this is what happens in these visualizations.

In the previous step of the analysis we already supposed that the professors work on different domains and as we only added the coauthors of the professors to amplify the effect, we can confirm this theory assessing also that there is no real dominant domain (otherwise the cloud effect would not be possible).

4.8 Final Remarks and Conclusions

With this example of a real analysis we have seen that it is possible to use Ebony to analyze a pretty large group and anyway obtain clear, intuitive and very interesting visualizations that can help to study or understand situations that, without the help of a visualization, would really be hard to explain (and prove).

Although this analysis is not the most complex one that a user might image, without the help of a visualization it would be almost impossible to reach the conclusions we have discussed in the previous and in this section.

We were able to deeply analyze the faculty of informatics of the University of Lugano by producing, with very simple operations, different visualizations that showed us the group of professors from different perspectives.

After all the analysis of the produced visualizations that have executed in the previous sections, we are now able to draw some conclusions about the evolution of the faculty, the work done there and its professors:

1. The employment of new professors increased the faculty productivity.
2. Hiring new professors do not change the productivity of the other professors.
3. There are very few collaborations between the professors in the faculty.
4. The professors work on many different domains.

Chapter 5

Conclusions

In this document, we have presented the main aspects that make Ebony a solid software for any real world analysis that deals with the data in the DBLP database.

It is possible to take into consideration the desired aspects, looking at the authors from a personally chosen point of view, and, moreover, the user can visualize the information in many different ways.

Chapter 3 specifically presented these and many other features of Ebony, explaining in detail all the possibilities that the software gives to the user to visualize the data and to fully customize the output.

In the simple cases, these customization features will most probably be ignored, but with more complicated analysis, the possibility to modify the output, also to emphasize some particular author or group, becomes a key feature.

In Chapter 4 we presented a real case study showing how Ebony can effectively be used in real world examples.

Moreover it has been shown that Ebony is able to deal with any kind of group of authors, of any size, always producing the desired visualization.

Using a real situation to test Ebony, we have also underlined the importance of visualizing data, concept presented in Chapter 2.

In conclusion, we reached the goal of building a software that satisfies the three criteria mentioned in the introduction (consult Chapter 1), in fact Ebony:

1. Gives complete access to the information contained in the database.
2. Gives to the user the possibility to visualize the information in different and meaningful ways.
3. Provides easy and fast instruments to manipulate the data and to extract relationships or metrics of interest.

Moreover Ebony tried to encapsulate these features in a nice looking and intuitive user interface, providing a set of intuitive commands.

We designed the application not only to be usable, but also to have a good performance and avoid any possible scalability issue.

The communications with the server part are reduced to the minimum.

The features implemented so far guarantee a comfortable user experience and give the possibility to the user to adopt Ebony in real world cases.

The database used by Ebony will always be updated with the new data contained in the DBLP database to guarantee to the user the best service.

The complete application can be found and used at: **http://ebony.inf.usi.ch**.

5.1 Limitations & Future Works

The features already implemented are sufficient to do a large number of analysis, moreover we have decided not to stop but to continue developing the project and improve the system.

While testing the software the developers pointed out some limitations that will be addressed in the next releases by implementing the following improvements:

- Try to decrease the time necessary to produce complex graphs (like MDS graph and force-directed graph), although the problem is a drawback of the algorithms used to produce these visualizations. Therefore, having many authors to be visualized, it will also take a considerable amount of time to produce them. Of course having a really powerful client machine helps to drastically decrease the time.
- Make improvements to reduce to the minimum the number of times that the visualization is redrawn, as redrawing is really costly and might slow down the application if there are many authors in the visualization.

Furthermore, there are also other parts of the software that will me modified in the next releases and, as a guideline, the developers decided that each new version should focus, beyond fixing eventual bugs and doing improvements, on the following points:

- Adapt the software to eventual changes in the DBLP database or extract new data if they are signaled to be useful.
- Implement new visualization types.
- Make the visualizations as flexible as possible.

Appendix A

GWTDisplay Library

In this appendix we are going to present shortly the GWTDisplay, a library that has been developed while creating Ebony and that has been used in the visualization process.

There will be no deep explanation as this document intends only to give the necessary information to understand how GWTDisplay can be used and extended.

A.1 Main Idea

GWTDisplay has been created with the idea of developing a graphical library for GWT [5] with the following key features:

1. Have different and highly customizable visualizations, to visualize the data in many ways.
2. Abstract the visualization from the model, providing only an interface to be used for connecting the model to the view.
3. Be easily extensible.

GWTDisplay was inspired under some aspects by EyeSee [6].

The library uses the components of the GWT-Graphics library [4] for the normal drawings and other external libraries (LinLogLayout [8] and part of Taxionomy [10]) as support for the implementation of part of some of the visualizations available in the library.

A.2 Implementation and Usage

As we can see from Figure A.1 the implementation of the library is pretty complex, but the design is very well done and this makes the library easily usable and extendable.

It has to be noticed that all the classes which directly refers to drawing entities (*Strokeable*, *Shape*, *DrawingArea*, etc.) are part of the GWT-Graphics library.

To use the library, another application has to be modified as follows:

- Each element that has to be visualized has to implement the *Visualizable* interface.
- For each different metric that the elements contain, a new type of data, that inherits from the DataType class, has to be created.
- For customizing the appearance of each element a custom class that implements the DisplayInfo interface should be used. An instance of this class should be returned by the element when the getDisplayInfo() method is called.

Instead, to extend the library and add new visualizations there are two main options:

1. Subclass one of the two existing types of visualizations (chart and graph) or subclass directly one of the visualizations.
2. Create a new visualization type from scratch, implementing the *Visualization* interface.

The idea is to expand GWTDisplay in every project in which it is used, by adding new visualization types. The final goal is to have library that can be used to visualize anything and which is easily includable in every project.

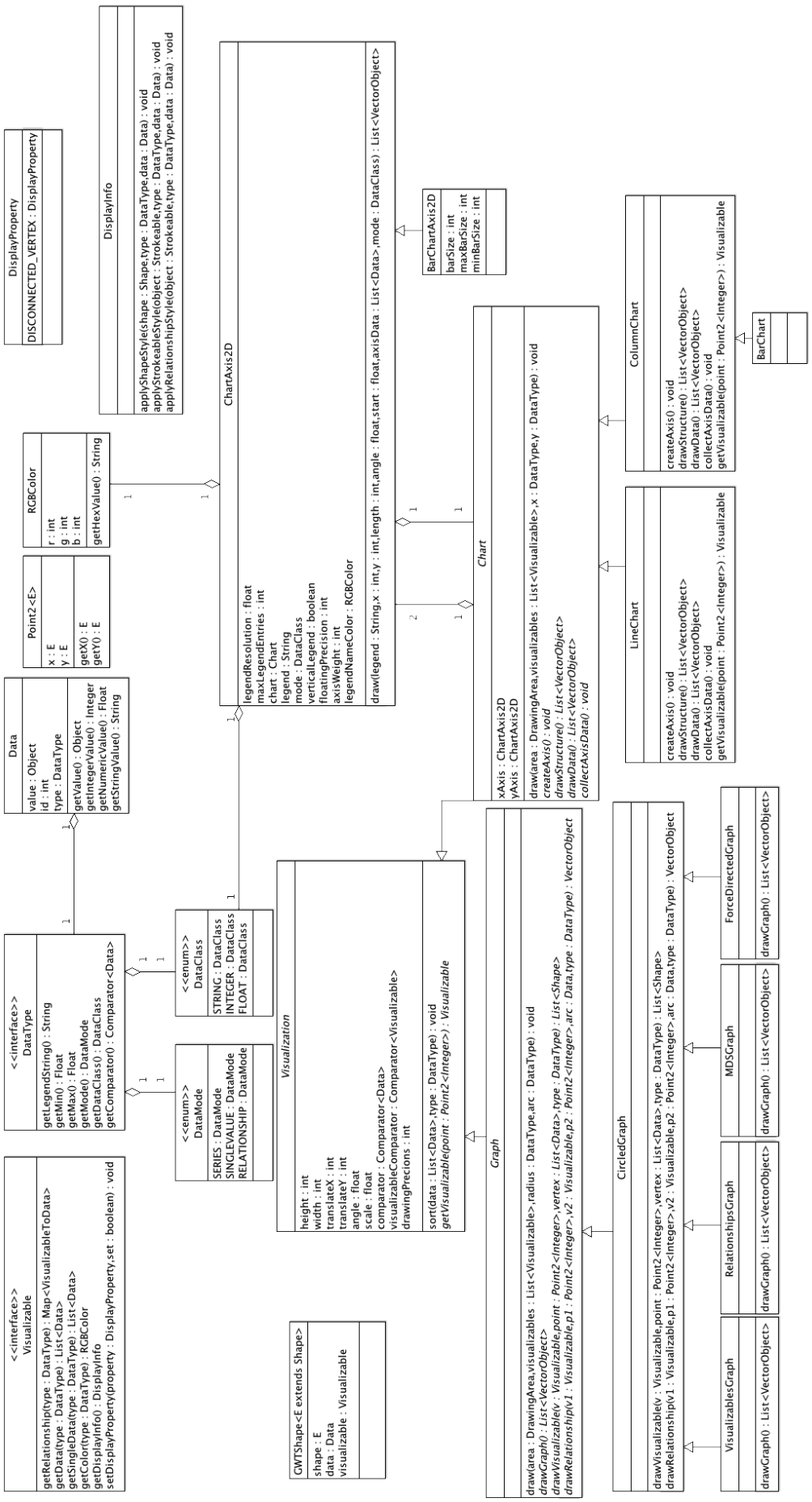


Figure A.1. UML Diagram of the GWTDisplay Library

Appendix B

Implementation Details

In this section some details about the implementation of Ebony are presented by commenting simplified versions of the class diagrams of some parts of the application.

B.1 Client Model

The diagram presented in Figure B.1 is an extended version of the one presented in Section 3.1 (Figure 3.2).

The publications contained in the DBLP database are converted in authors (class *Author*) that have a list of publications (class *Publication*) they have produced.

To enable a simple access only to the part of data relative to a specific time interval we have divided the career of the author in years (class *AuthorYear*). Each year contains the publications done in that annum and also the coauthors (class *CoauthorEntry*) that have worked with the author, including the exact number of collaborations.

Finally the model provides an interface that can be implemented if a class wants to communicate with the server for retrieving the authors from the database.

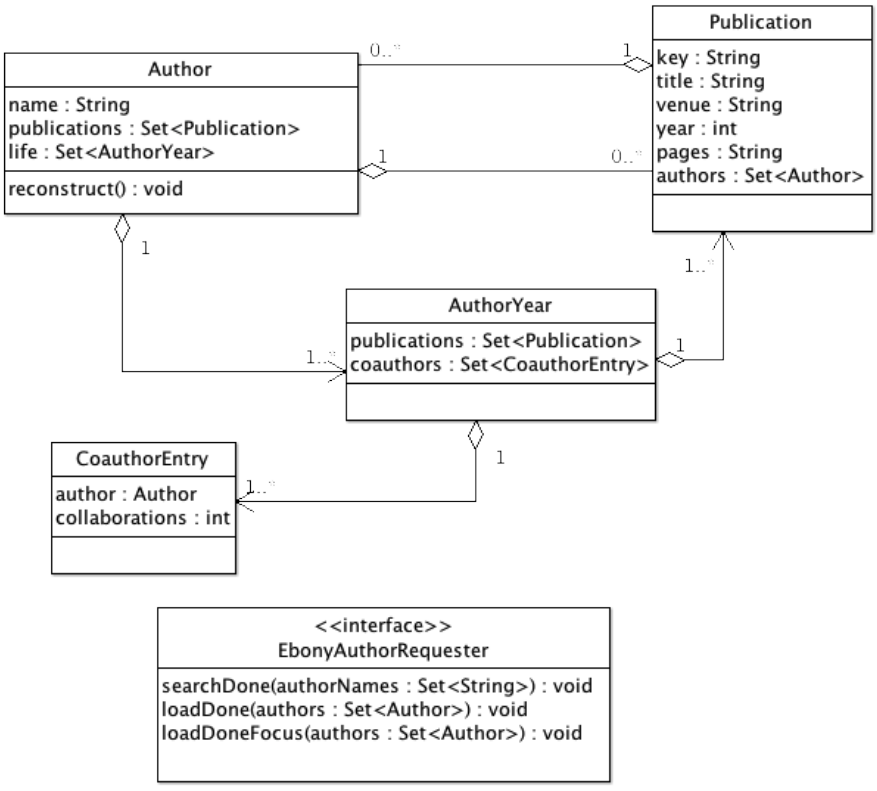


Figure B.1. UML Diagram of the model behind the data in Ebony

B.2 Client Visualization

Figure B.2 shows a simplified structure of how the visualization process is internally managed in Ebony. It presents also the connection between Ebony and GWT [5] for the client side part.

Class *EntryPoint* is an interface that is part of GWT and needs to be implemented by the class that is responsible for rendering the whole web page.

In fact class *Ebony* is instantiated each time that a client connects to the application. Each instance has its own cache (the client-side cache, class *EbonyCache*) and has a requester (class *EbonyRPCRequester*) to communicate with the server and retrieve (or save) data.

The visualization process is based on the *EbonyView* class, which is a subclass of the *DrawingArea* class (part of GWT-Graphics [4]) and which is responsible for creating the code to be used in the HTML5 Canvas to display the drawings.

The viewing process uses the GWTDisplay library (see Appendix A) to actually produce the visualization. In fact each instance of *AuthorView* is a *Visualizable* that is used by the library to extract the necessary data for realizing the required visualization.

Each *EbonyView* instance has its own set of data (class *EbonyViewData*), which are mostly related to he authors present in the visualization.

Moreover each instance has a drawer (class *EbonyViewDrawer* and subclasses), which determines what kind of visualization is going to be produced. The drawer is actually directly connected to GWTDisplay and uses the library to obtain the elements (shapes, etc.) that have to be visualized.

These elements are then processed by the *EbonyView* instance and converted into code that the HTML5 canvas can understand.

Each drawer is specialized to utilize only one of the visualizations offered by GWTDisplay. This design choice allows us to easily react to changes in the GWTDisplay library: if a new visualization is offered, a new drawer is created; if a visualization is deleted, the drawer is removed.

To customize the visualization each view instance has an options getter (class *EbonyViewOptionsGetter*, in which all the data about options and customizations are stored. The getter contains a list of option components (class *EbonyViewOptionComponent*) which have a state that directly influences the view. The option components that are used to modify the visualization depends on the getter, which is imposed by the drawer.

It has to remarked the importance of the class *AuthorView*. Each instance encapsulates an author and moreover has the information regarding the time interval on which each *EbonyView* that has been created focus on.

The idea is to have only one *AuthorView* instance for each author in every *Ebony* run. In fact the client-side cache contains *AuthorView* instances.

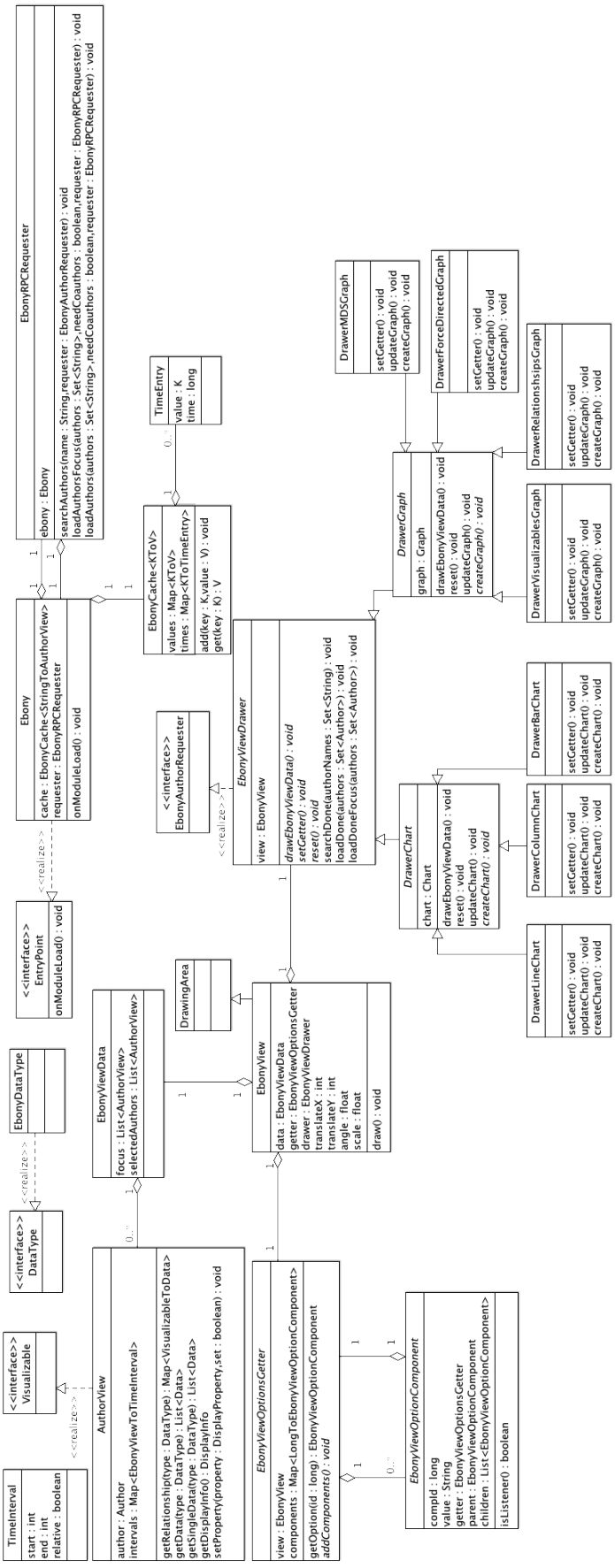


Figure B.2. UML Diagram of the viewing process in Ebony

B.3 Server

The server side (Figure B.3 shows a simplified class diagram) of Ebony is focused mainly on offering to the clients an efficient way for retrieving and saving the desired data.

A first remark to be done is that *RemoteServiceServlet* and *RemoteService* are part of GWT [5] and allows us to easily create a server side for out project. In our case the servlet is not only the actual server side to which the clients connect but is also the furnisher of all the data.

The structure of the server side is not really complex and there are few key points to be understood:

- The servlet delegates all the work related to searching or loading data to the *EbonyDataServer* instance that is created when the server is started.
- The data server directly owns the cache and contacts the database only if the data requested are not available in the cache. The database is queried using an *EbonyDB* instance.
- If it is necessary, for example an author has not been loaded completely (i.e. load his information and also the ones regarding his coauthors), the data server creates a new task (class *EbonyTask*) that signals the necessity of loading new data.
- The *DataCollector* instance runs in another thread and executes one after the other the tasks created by the data server. If there are no tasks to be executed, the data collector suspends his execution until a task is created. Therefore the collector is efficient and does not waste any resource.

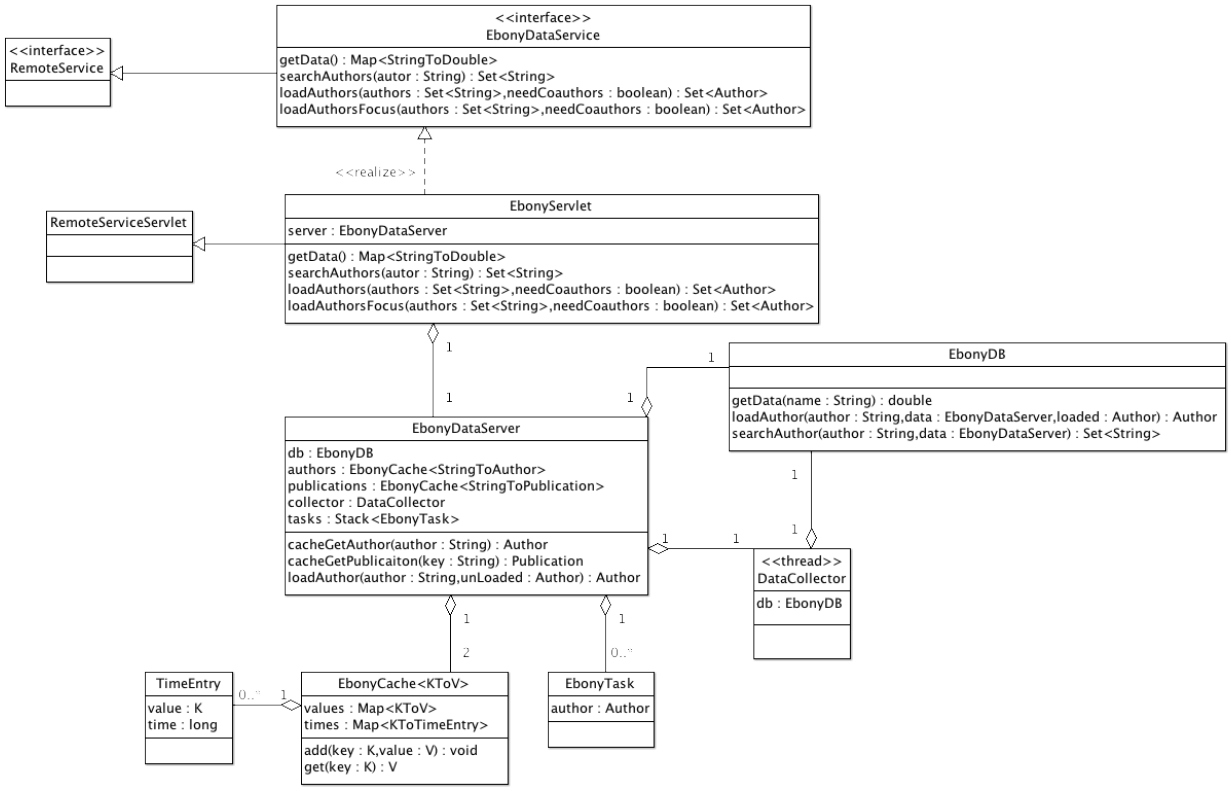


Figure B.3. UML Diagram of the server side of Ebony

Bibliography

[1] G. D. Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, 1999.

[2] DBL-Browser Project: http://dbls.uni-trier.de/DBL-Browser/.

[3] DBLPVis Project: http://dblpvis.uni-trier.de.

[4] GWT-Graphics Project: http://gwt-graphics.googlecode.com.

[5] GWT Homepage: http://code.google.com/webtoolkit/.

[6] M. Junker and M. Hofstetter. *Scripting Diagrams with EyeSee*. 2007.

[7] A. Kuhn, P. Loretan, and O. Nierstrasz. *Consistent Layout for Thematic Software Maps*. 2008.

[8] LinLogLayout Project: http://linloglayout.googlecode.com.

[9] A. Noack. *Unified Quality Measures for Clusterings, Layouts, and Orderings of Graphs, and Their Application as Software Design Criteria*. 2007.

[10] Taxionomy MDS Implementation: https://trac.v2.nl/browser/taxionomy/trunk/src/nl/v2/taxionomy/MDS.java.