**Software Requirements**

**Specification**

**for**

**Gephi**

## Version 0.92 Prepared by Konstantinos Varvoutas Aristotle University of Thessaloniki

**February 2017**

***Copyright © 1999 by Karl E. Wiegers. Permission is granted to use, modify, and distribute this document.***

# Table of Contents

[Table of Contents ii](#_TOC_250028)

[Revision History iiii](#_TOC_250027)

1. [Introduction 1](#_TOC_250026)
   1. [Purpose 1](#_TOC_250025)
   2. [Document Conventions 1](#_TOC_250024)
   3. [Intended Audience and Reading Suggestions 1](#_TOC_250023)
   4. [Product Scope 1](#_TOC_250022)
   5. [References 2](#_TOC_250021)
2. [Overall Description 3](#_TOC_250020)
   1. [Product Perspective 3](#_TOC_250019)
   2. [Product Functions 3](#_TOC_250018)
   3. [User Classes and Characteristics 7](#_TOC_250017)
   4. [Operating Environment 8](#_TOC_250016)
   5. [Design and Implementation Constraints 8](#_TOC_250015)
   6. [User Documentation 8](#_TOC_250014)
   7. [Assumptions and Dependencies 9](#_TOC_250013)
3. [External Interface Requirements 10](#_TOC_250012)
   1. [User Interfaces 10](#_TOC_250011)
   2. [Hardware Interfaces 18](#_TOC_250010)
   3. [Software Interfaces 19](#_TOC_250009)
   4. [Communications Interfaces 19](#_TOC_250008)
4. [System Features 19](#_TOC_250007)
   1. [Graph Visualization 19](#_TOC_250006)
   2. [Graph Layout 22](#_TOC_250005)
   3. [Graph Metrics 23](#_TOC_250004)
   4. [Filters 24](#_TOC_250003)
   5. [Data Table 25](#_TOC_250002)
   6. [Dynamic Graphs 28](#_TOC_250001)
   7. [Graph Export 29](#_TOC_250000)
5. **Other Nonfunctional Requirements 30**
   1. Performance Requirements 30
   2. Safety Requirements 30
   3. Security Requirements 30
   4. Software Quality Attributes 30

**Glossary 31**

# Revision History

|  |  |  |
| --- | --- | --- |
| **Name** | **Date** | **Version** |
| Gephi | 8 February  2010 | 0.70 Alpha |
| Gephi | 17 February  2010 | 0.70 Alpha2 |
| Gephi | 1 April 2010 | 0.70 Alpha3 |

|  |  |  |
| --- | --- | --- |
| Gephi | 30 April 2010 | 0.70 Alpha4 |
| Gephi | 1 October 2010 | 0.70 |
| Gephi | 8 April 2011 | 0.80 Alpha |
| Gephi | 4 October 2011 | 0.80 |
| Gephi | 31 March 2012 | 0.81 |
| Gephi | 3 Jan 2013 | 0.82 |
| Gephi | 20 December  2015 | 0.90 |
| Gephi | 14 February  2016 | 0.91 |

# Introduction

## Purpose

The purpose of this document is to present a detailed description of the open-source software Gephi. It will explain the purpose and features of the software, the interfaces of the software, what the software will do and the constraints under which it must operate. This document is intended for users of the software and also potential developers.

## Document Conventions

This Document was created based on the IEEE template for System Requirement Specification Documents.

## Intended Audience and Reading Suggestions

* Typical Users, such as students, who want to use Gephi for analyzing networks (Social networks, Social Media networks, Semantic networks etc.)
* Advanced/Professional Users, such as engineers or researchers, who want to use Gephi for more demanding graph analysis.
* Programmers who are interested in working on the project by further developing it or fix existing bugs.

## Product Scope

Gephi is a tool that people can use to analyze graphs and networks. Users can use it to represent their data as a graph and interact with them. They can alter the layout of the graph or calculate metrics of it, to understand their data in a better way and discover hidden aspects about it, such as patterns or faults during data sourcing.

This is a software for Exploratory Data Analysis, a paradigm appeared in the Visual Analytics field of research.

## References

Gephi’s website:

https://gephi.org/

Gephi’s GitHub page:

https://github.com/gephi

IEEE Template for System Requirement Specification Documents:

https://goo.gl/nsUFwy

GNU General Public License version 3: <http://www.gnu.org/licenses/gpl.html>

CDDL Common Development and Distribution License: https://opensource.org/licenses/CDDL-1.0

# Overall Description

## Product Perspective

Digital Board Marker is developed for recording lectures as animations, saving the size up to 100 times as compared to regular video lecture. It is developed for students who face problem in jotting down the lecture and cannot focus on the lecture.

It is an open source project and it has a very active developer team to support it and provide feedback to users. It is developed to run on Windows, Android and Web Browser.

## Product Functions

## Desktop and Mobile Application

File:

* Open: Loads an existing Lecture file.
* Open Recent: Loads one of the displayed, recently opened Lecture files.
* Properties: Displays some properties of the Lecture (such as the title, duration and number of topic tags) which can be edited.
* Save: Saves the edited lecture file without changing its name or directory.
* Save as: Saves the lecture file and gives the user the ability to change its name or directory.
* Exit: DBM app shuts down

Player Window:

* Resize: Resize the application player view window.
* Play/Pause: Plays or Pauses the lecture animation.
* Audio control: Controls the audio level.
* Next: Play next lecture in lecture playlist.
* Previous: Play previous lecture in lecture playlist.
* Fast Forward: Speeds up lecture animation and audio track as well.
* Slow down: Slower the animation speed and also audio track.
* Timeline: Displays the timeline tab.

Splash Screens:

* Welcome: Displays the Welcome window.

Help:

* Check for Updates: Displays the plugins that can be updated to newer versions
* About: Displays the logo of DBM, which licenses are being used, the product version and other info.

**Web Application**

Main Pages:

* Login/Registration: Displays the Login and Registration pages in which students as well as teachers or instructors can be registered and login afterwards.
* Course Section: Displays all available courses to student and instructor
* Lectures Section: Includes lecture hierarchy and lecture player

Login/Registration:

* Registration Page: Requests First name, Last name, CNIC, Degree, Email, Country, Education institution, Password from user being registered and sends a verification email.
* Login Page: Require Username and password from user.

Course Section:

* All Courses Page: Includes all available courses and display them to the respective user or instructor.
* Registered Courses: Displays the courses in which current user is registered.

Lecture Hierarchy:

* Course Content Page: Lectures and the corresponding date.
* Configuration: Preferences about how the data is presented.
* Add Lecture: Manage recently recorded unlisted lectures
* Search: Stand Search functionality.

Lecture Player:

* Resize: Resize the application player view window.
* Play/Pause: Plays or Pauses the lecture animation.
* Audio control: Controls the audio level.
* Next: Play next lecture in lecture playlist.
* Previous: Play previous lecture in lecture playlist.
* Fast Forward: Speeds up lecture animation and audio track as well.
* Slow down: Slower the animation speed and also audio track.
* Timeline: Displays the timeline tab.

Splash Screens:

* Welcome: Displays the Welcome window.

Active Learning Control Panel:

* Show controls such as on/off and other button controls under this panel.

## User Classes and Characteristics

* Typical Users, such as students, who want to use DBM for watching online lectures as well as download to view it in offline mode later after.
* Professional Users, such as Instructors or teachers, who want to use DBM for editing and annotations.

## Operating Environment

* + Windows 7
  + Windows 8
  + Windows 10
  + Android 4.0 and higher
  + Web Browser with HTML5

## Design and Implementation Constraints

DBM is developed in C#, it uses .net Core as build platform of web application. Desktop application is developed in C# as well. It is developed as windows form application in visual studio 2015. Android app is developed in Android studio that features editing and annotation of video lecture and requires name, category and course name of the currently being uploaded lecture.

## User Documentation

There is a quick start guide available on the website of DBM:

## Assumptions and Dependencies

DBM web app is developed in C# visual studio 2015. Web application requires any browser that supports HTML5 and CSS3. Supported browsers are Chrome, Safari, Firefox latest version.

DBM desktop app is developed as windows form application in visual C#. It requires some runtime libraries to be installed on client’s machine such as

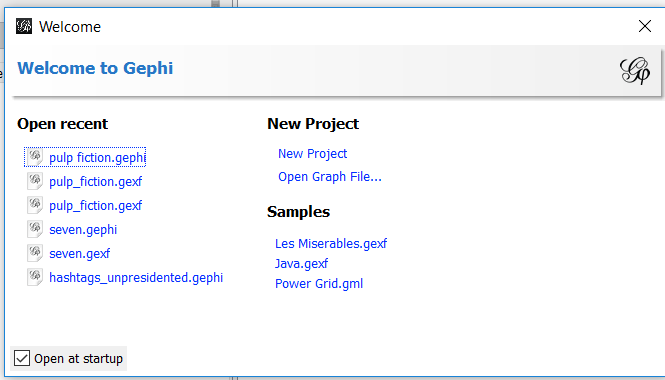
* Visual Redistributables
* .Net Framework 4.5

DBM Android app requires android version above 4.0, stable internet connectivity and QHD Display resolution in minimum.

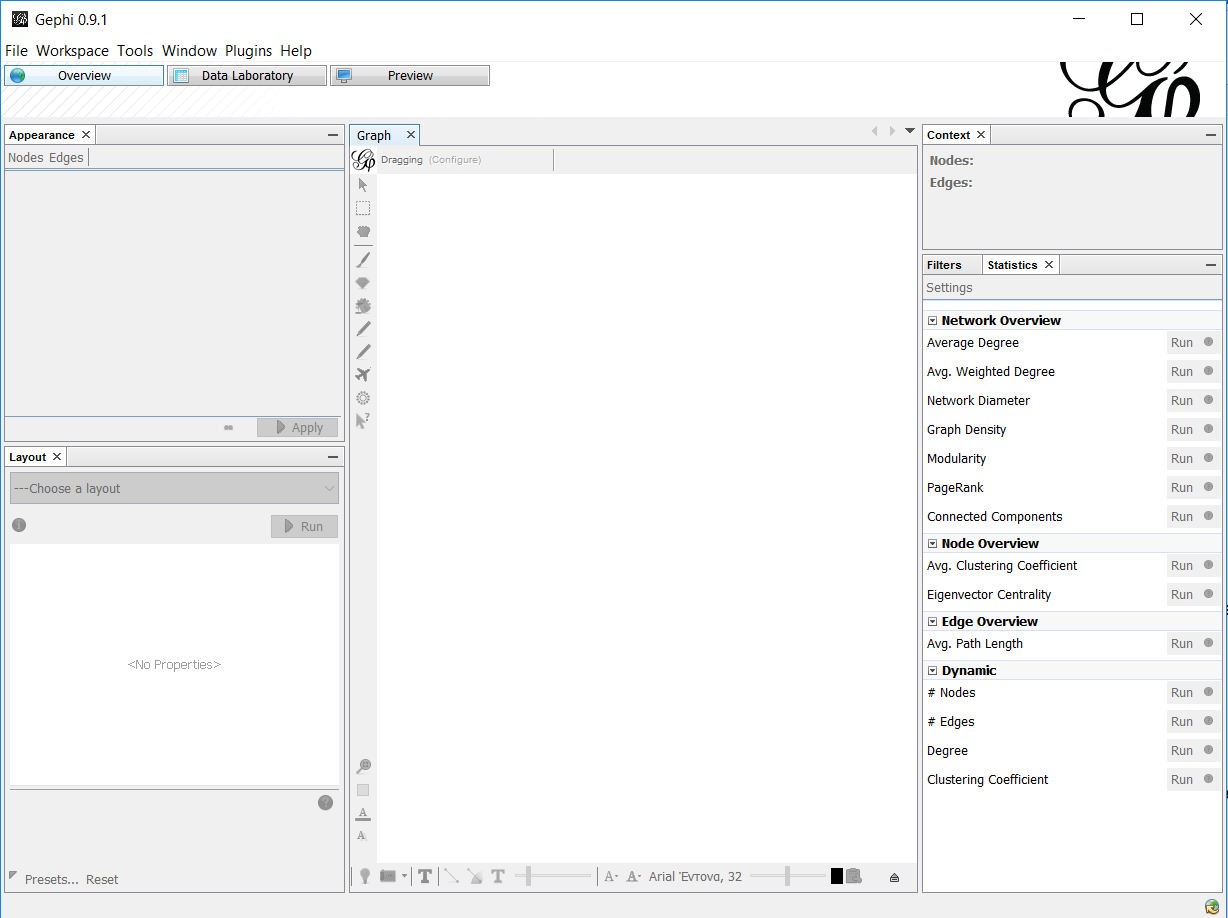
# External Interface Requirements

## User Interfaces

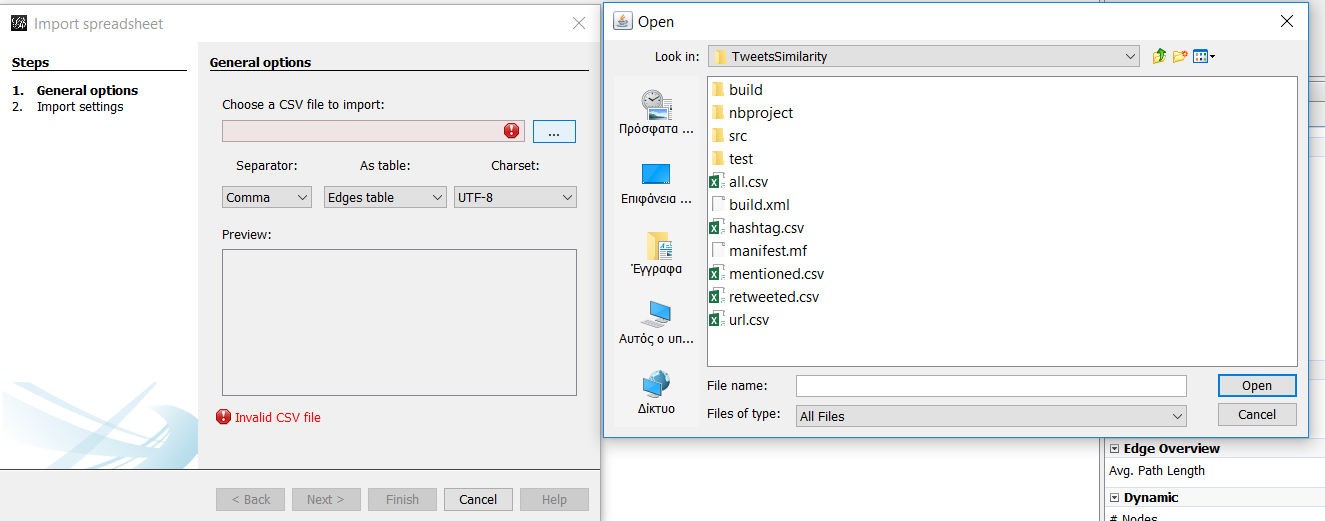
1. Gephi’s Welcome Screen:



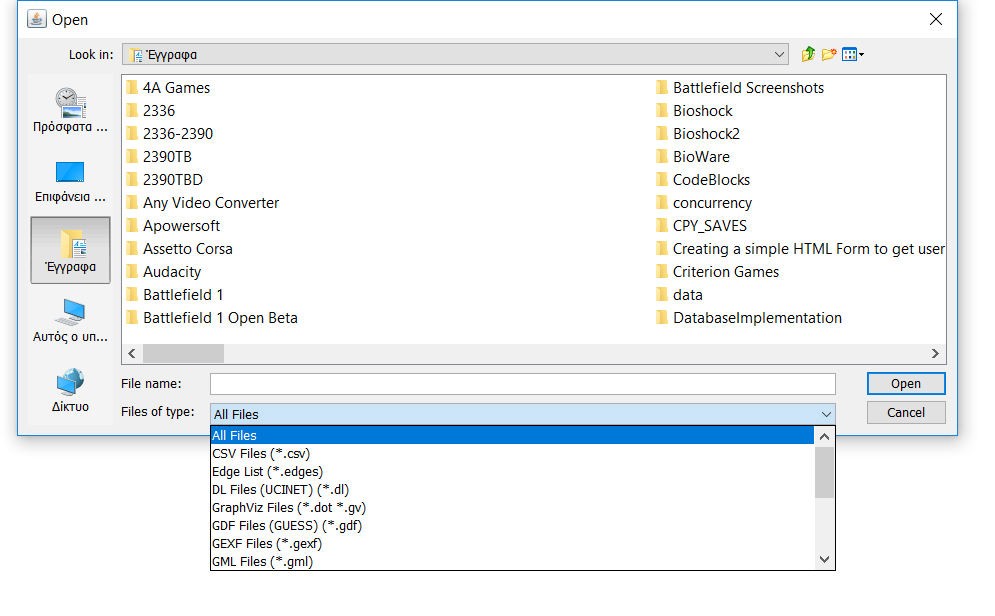
1. Gephi’s Main Screen:



1. Import Graph (CSV file)

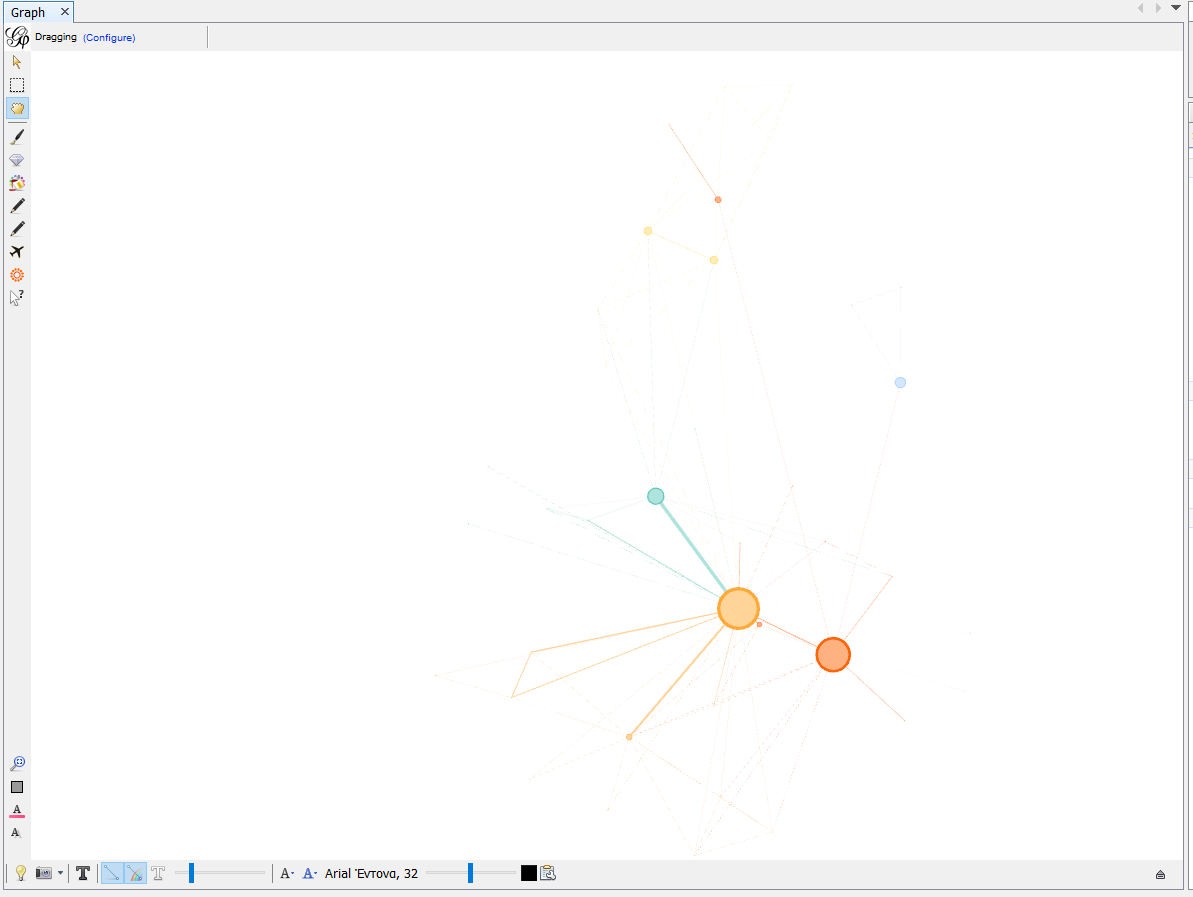


1. Open Graph (all supported file formats)



1. Overview of the Graph. Left side buttons, from top to bottom: direct selection (select only one node/edge), rectangle selection (select part of the graph ),

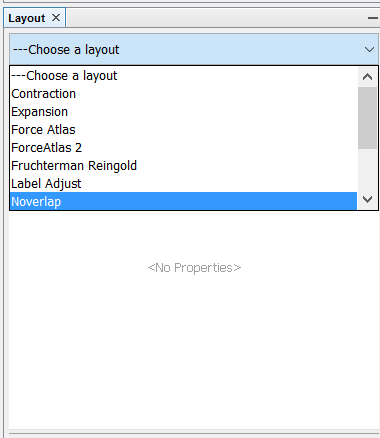
Drag ( drag a node using the mouse), Painter ( color nodes by clicking on them), Sizer ( change size of nodes by clicking on them and dragging the mouse vertically), Brush ( colors the selected node and its nearest neighbour), Node Pencil ( create new node where mouse is clicked), Edge Pencil (create new edge by clicking first on source node and then on target node), Shortest Path ( display the shortest path between two clicked nodes), Heat Map ( Set color intensity on a node neighbourhood, based on the distance from the clicked node), Edit (Click on a node to edit its attributes), Center on Graph, Reset Colors ( remove color from every edge and node), Reset Label Colors( remove color from every label), Reset Label Visible( make labels invisible).



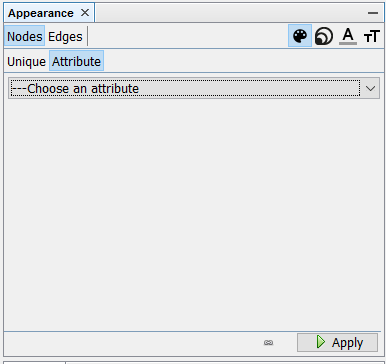
Bottom buttons, from left to right: Background color (left click to switch between black and white, right click to choose color), Take Screenshot, Show Node Labels, Show Edges, Edges have source node color, Show Edge labels, Edge Weight Scale, Size Mode( Fixed, Scaled or Node size) Color Mode( Unique, Object or Text), Font, Font

size scale, Default color, Attributes (choose which attributes are displayed on graph overview)

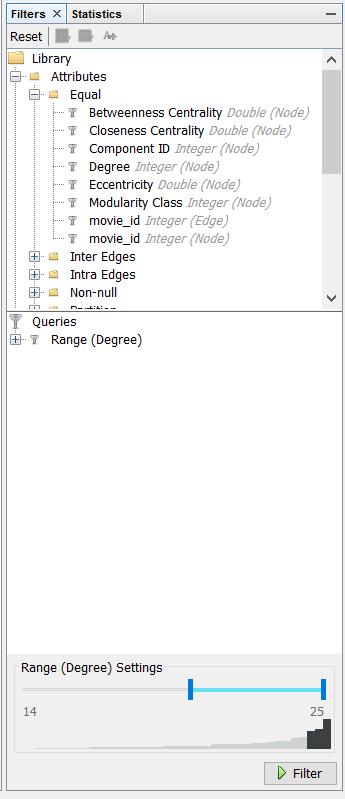
1. Change Graph Layout



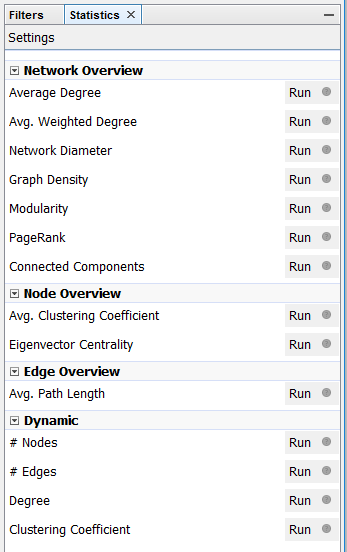
1. Change Node/Edge Colour (Unique applies the exact same colour to all nodes/edges. Attribute changes the colour of each node/edge according to its value of the selected attribute)



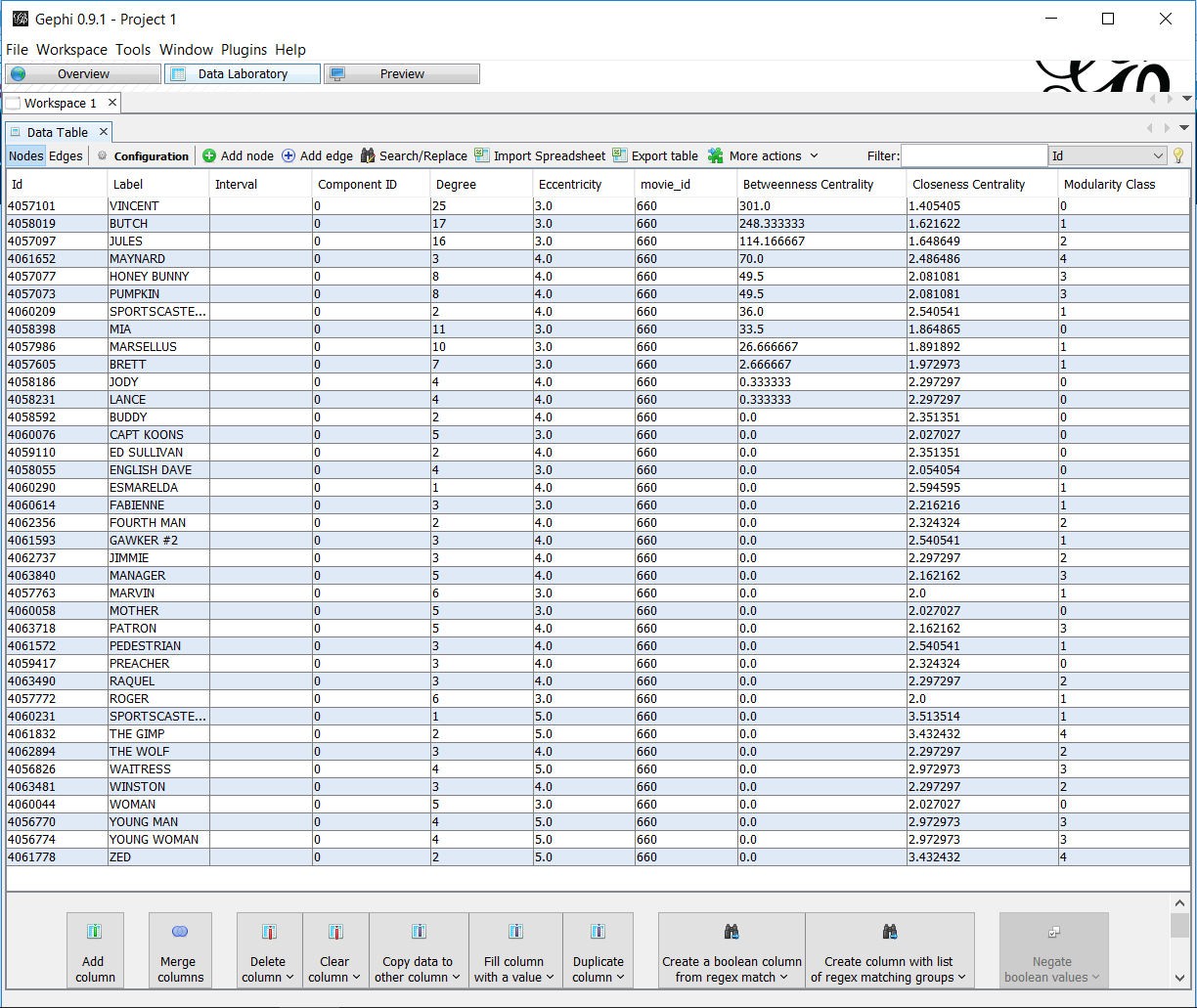
1. Apply Filters Create filters that can hide nodes and edges ( that do not comply with the filter’s settings) To apply a filter drag it from the “Filters” module to the “Queries” module and press the “Filter” button on the bottom right side. Filter settings can be modified.



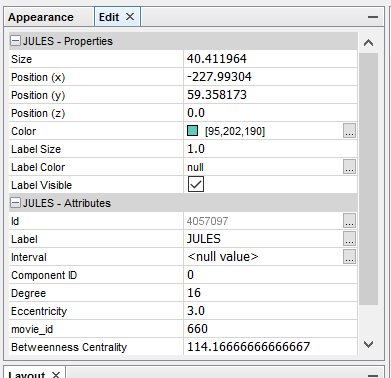
1. Calculate Graph Metrics



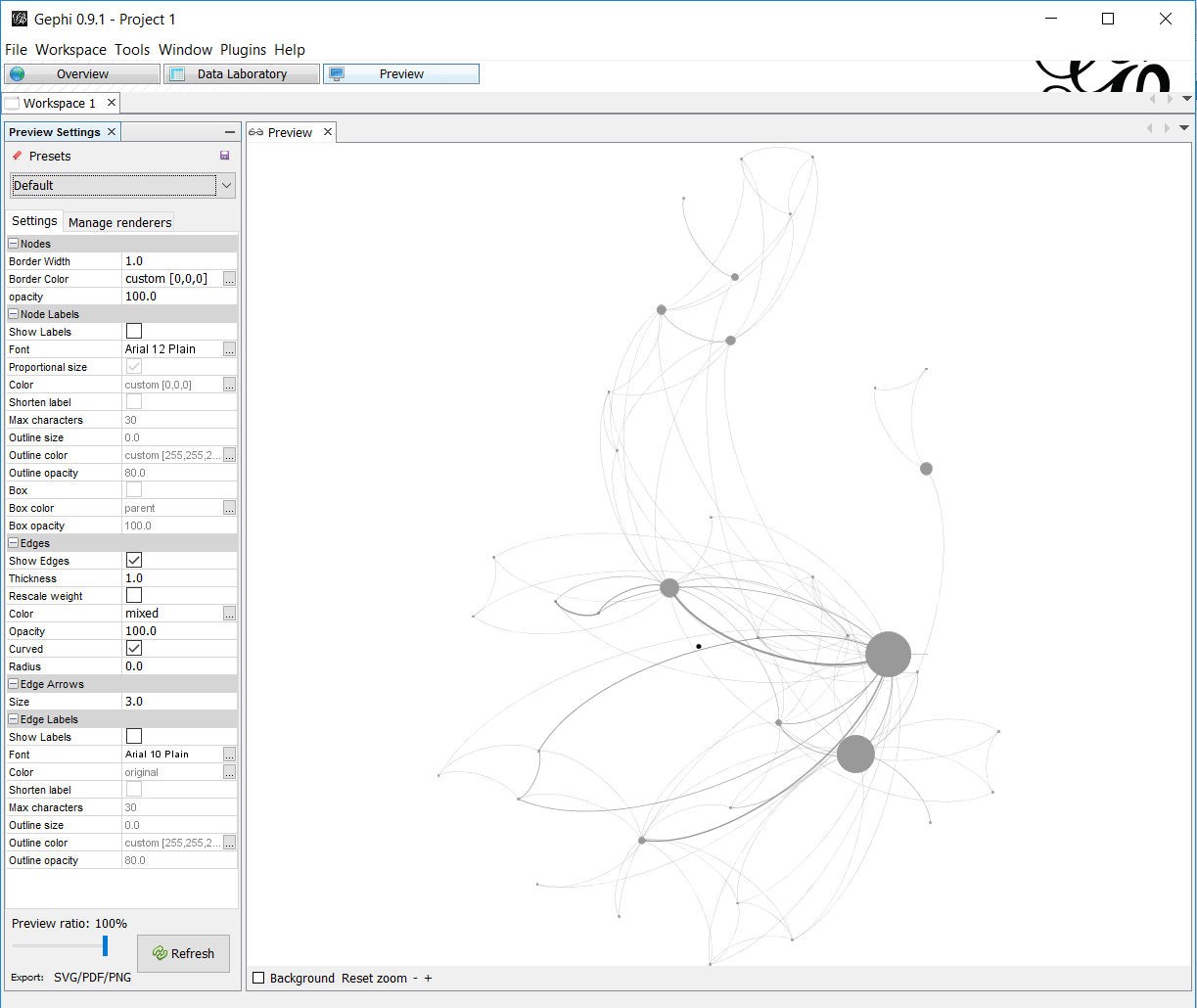
1. Data Laboratory



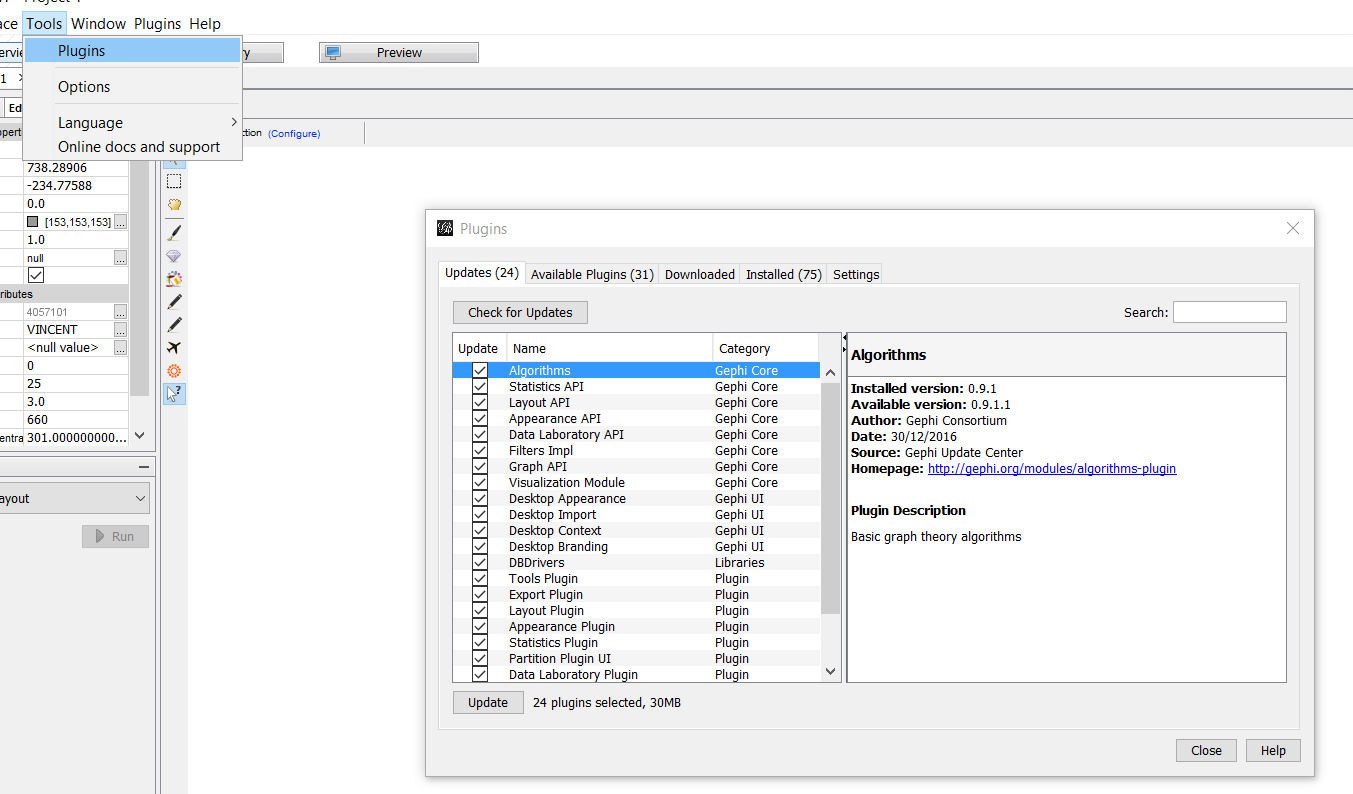
1. Edit Node (either use the “Edit” button and click on the Node in the Overview screen or left click on the Node in the Data Laboratory screen)



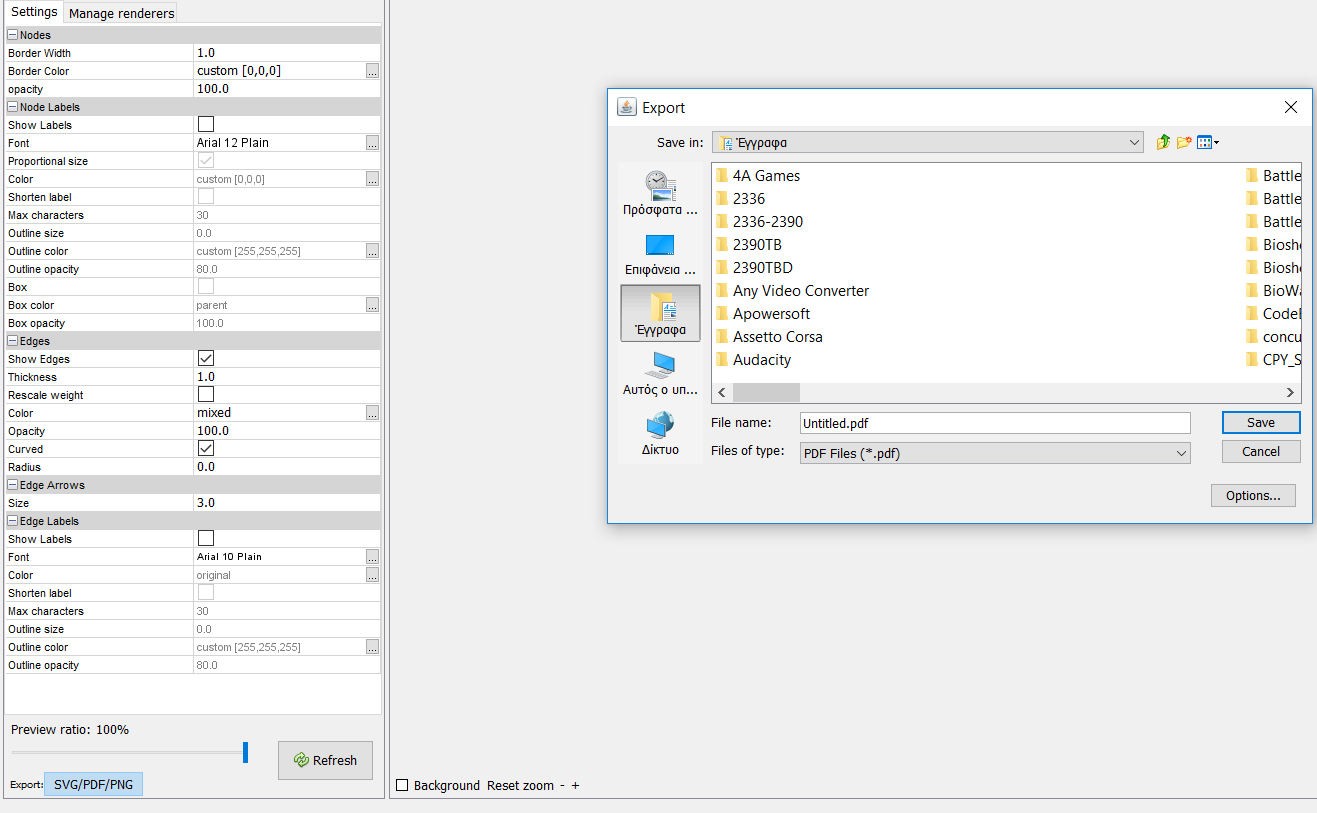
1. Preview Graph



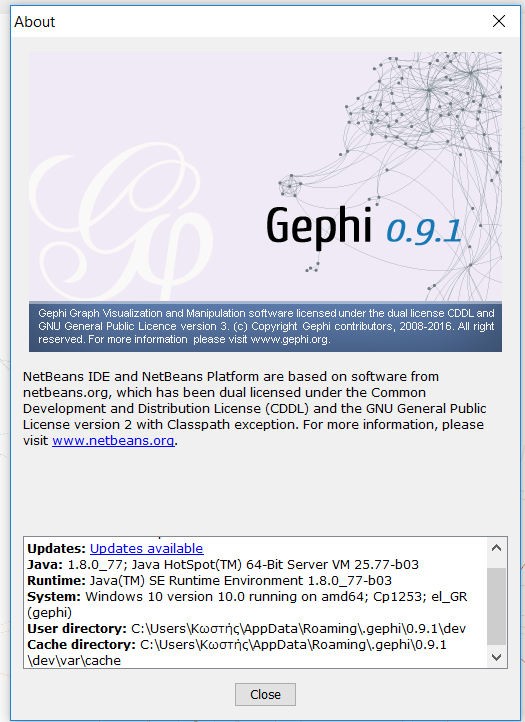
1. Manage Plugins



1. Export Graph



1. Help -> About



## Hardware Interfaces

The minimum hardware requirements of Gephi are a 500 Megahertz CPU and 128 megabytes of RAM. Also, because Gephi uses an OpenGL 3D engine to speed up graph visualization, a compatible graphics card is required. A system with these specifications can handle a Network of approximately 1000 edges and nodes. For bigger networks, additional memory is required (https://gephi.org/users/requirements/).

## Software Interfaces

Gephi requires Java to be installed on the system, more specifically Java version 7 or 8 for its latest release. Additional information can be found on section 2.7 of this document.

Gephi can be connected with a MySQL, SQLite or PostgreSQL database to import a graph edge list.

## Communications Interfaces

Gephi requires an internet connection to install new plugins, update already installed ones and update some of its components (APIs, modules etc.).

# System Features

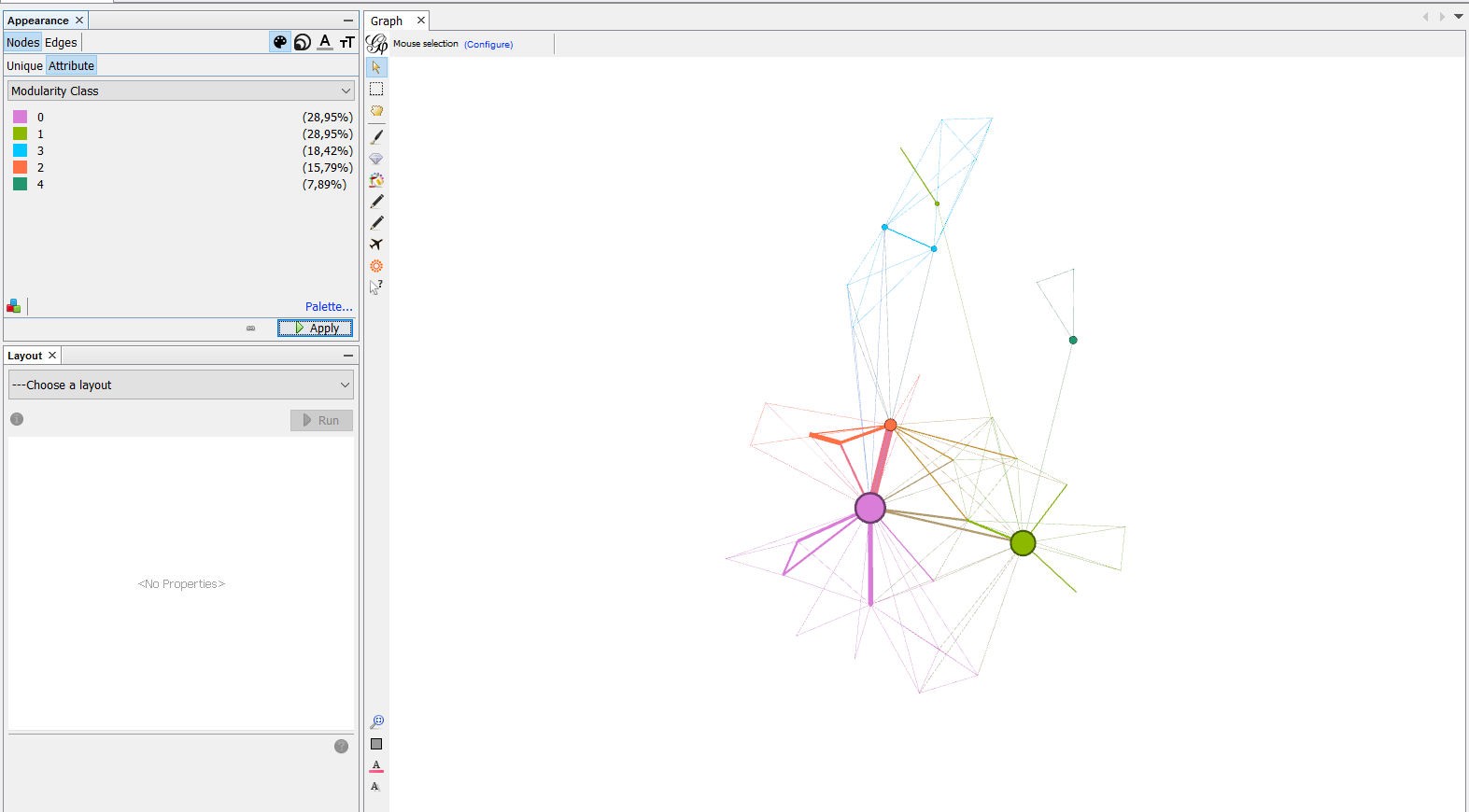
This section demonstrates Gephi’s most prominent features and explains how they can be used and the results they will give back to the user.

## Graph Visualization

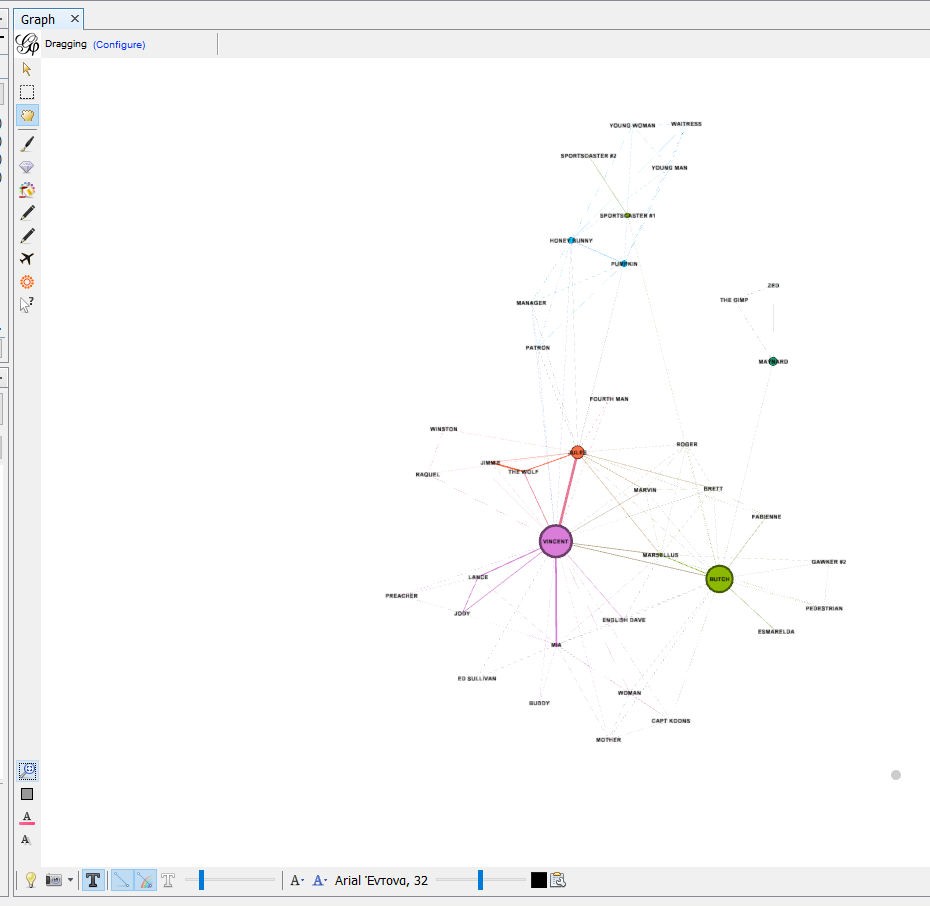
Users can directly interact with the graph by clicking on nodes to drag them, edit their attributes or color, add new nodes and edges, display shortest paths between nodes, make node labels visible/invisible etc.

Users can also use the appearance module to change the color of nodes. They can use the same color for the entire graph or use the values of a specific attribute as a reference point.

Example of changing the color of nodes based on one of their attributes (modularity class):



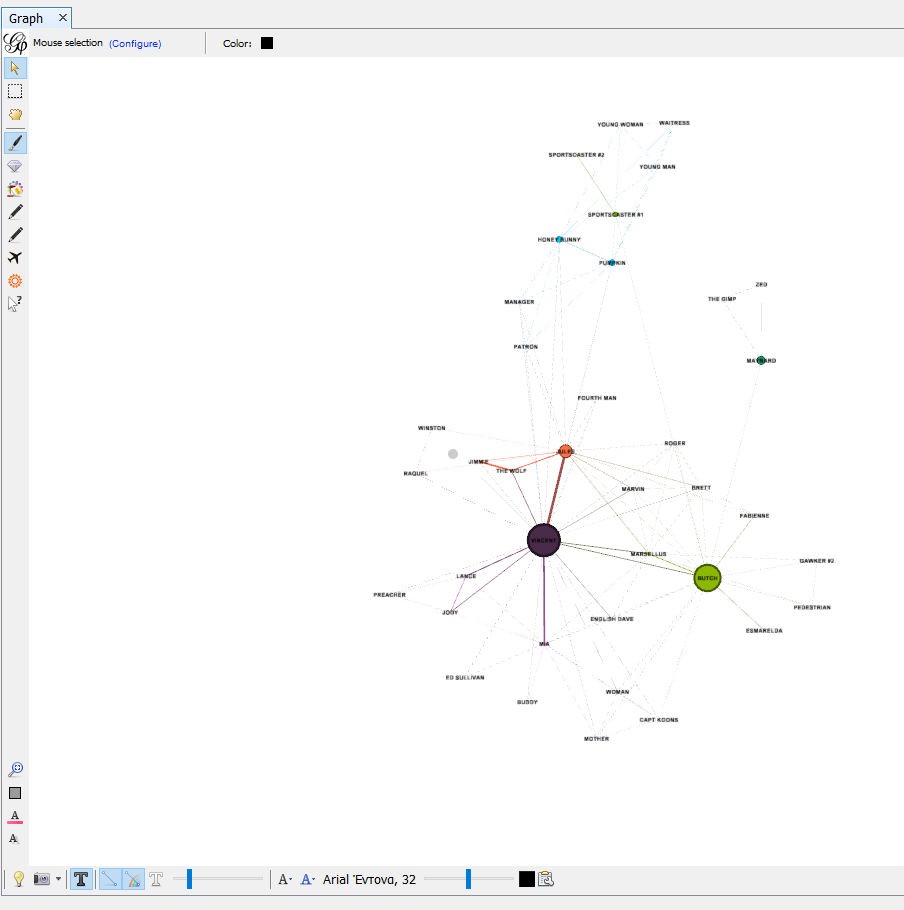
Example of making node labels visible (using the Show Node Labels – T button ):



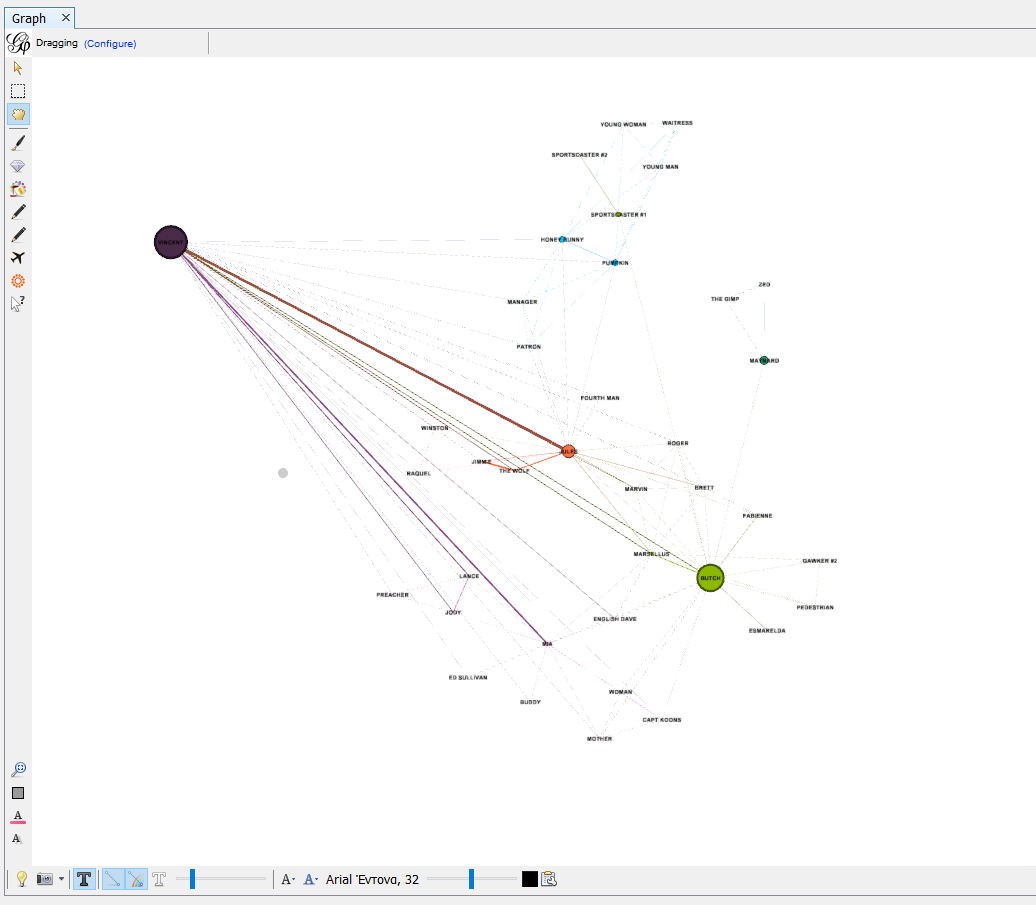
Example of chancing the color of only one node( Vincent) using mouse selection:

(The user has to press the “Painter” button first, on the left side of the screen, then choose

color, on the top of the screen, and finally click on the desired node)



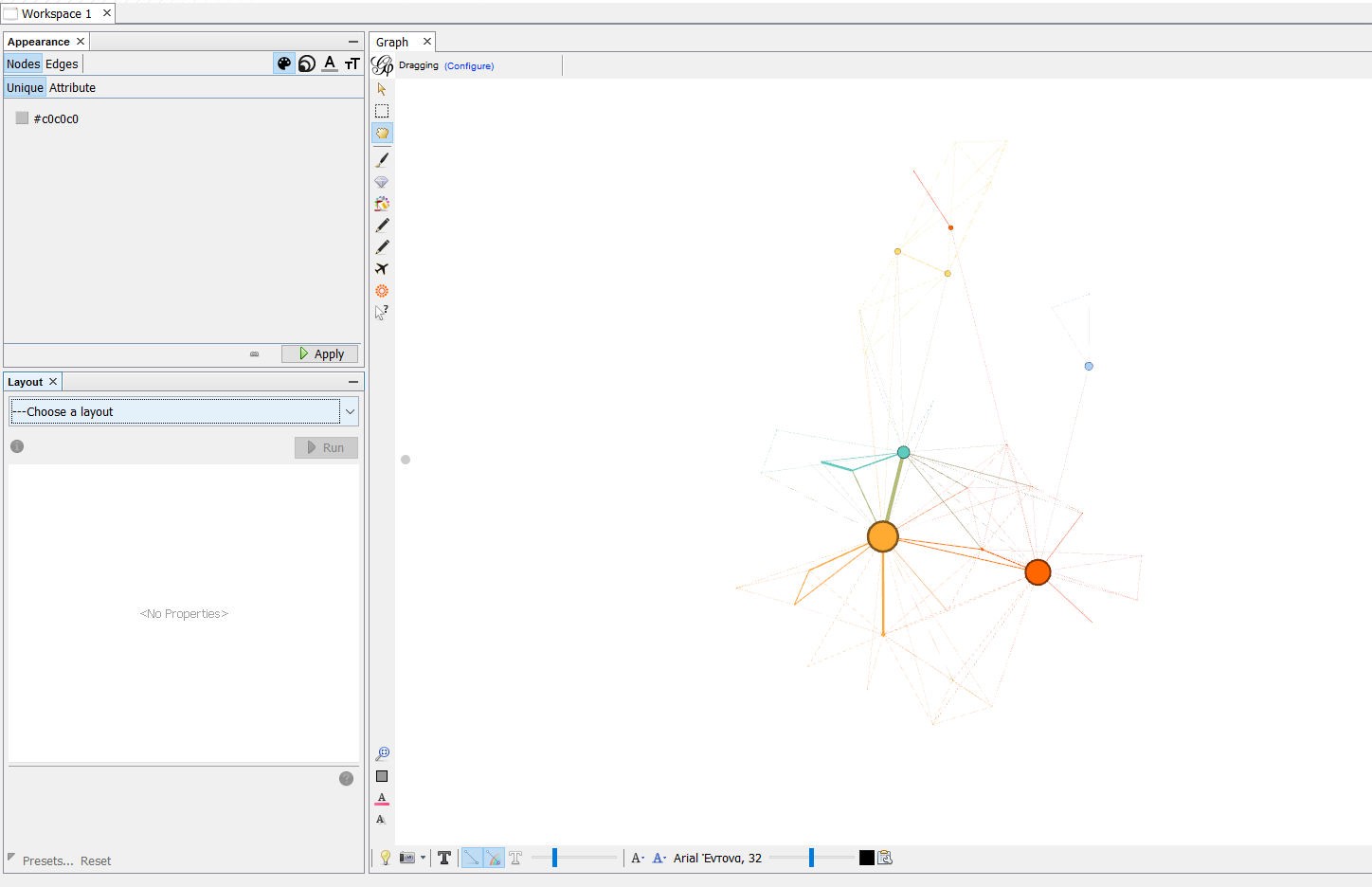
Example of Node Dragging (by pressing the ”Drag” button on the left side of the screen) :



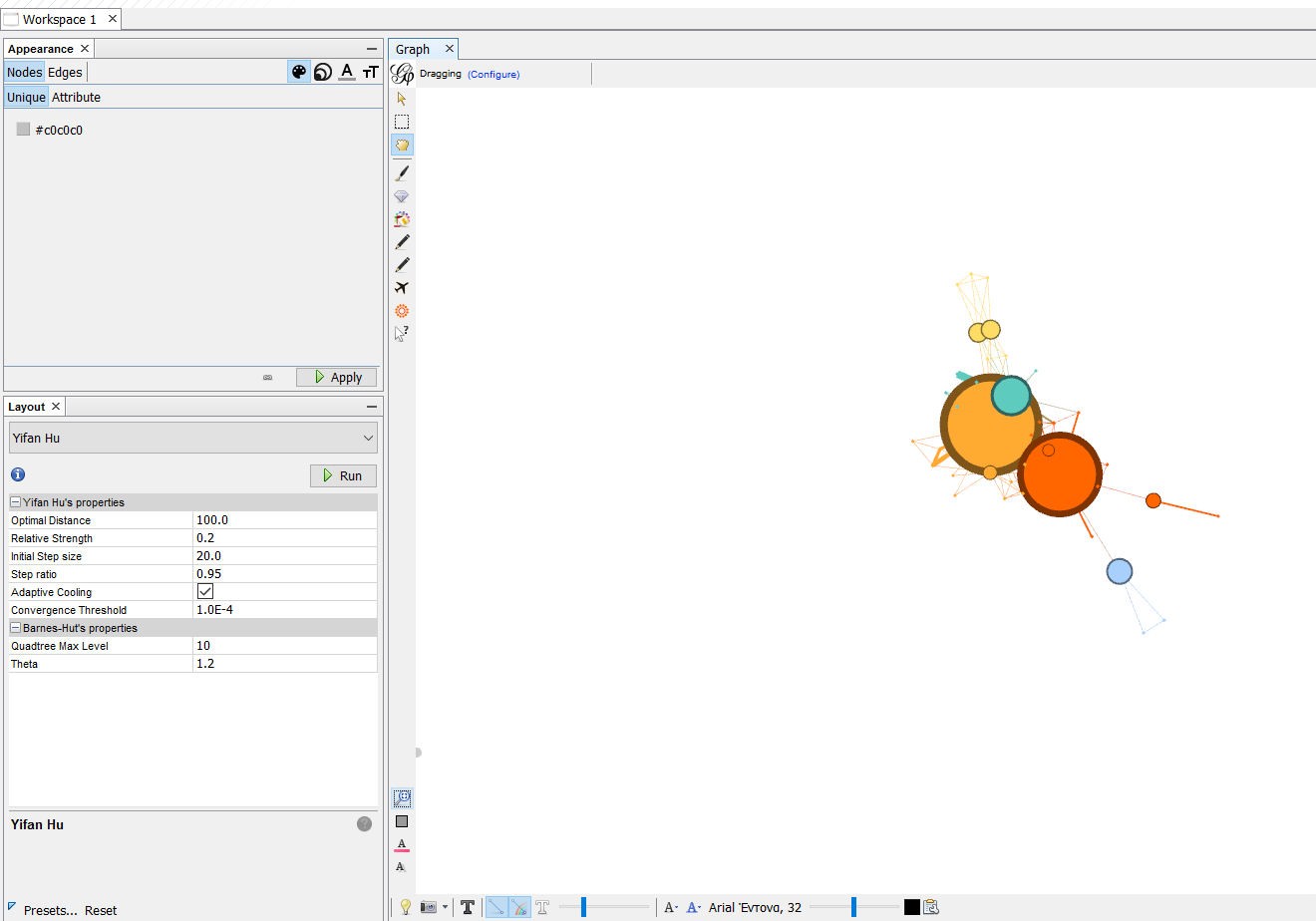
## Graph Layout

Gephi provides the following layout algorithms: Contraction, Expansion, Force Atlas, Force Atlas 2, Fruchterman Reingold, Label Adjust, OpenOrd, Random Layout, Rotate, Yifan Hu and Yifan Hu Proportional.

Example of applying a Layout algorithm to a graph: The graph, before changing the Layout:



The graph, after applying the layout “Yifan Hu”:



## Graph Metrics

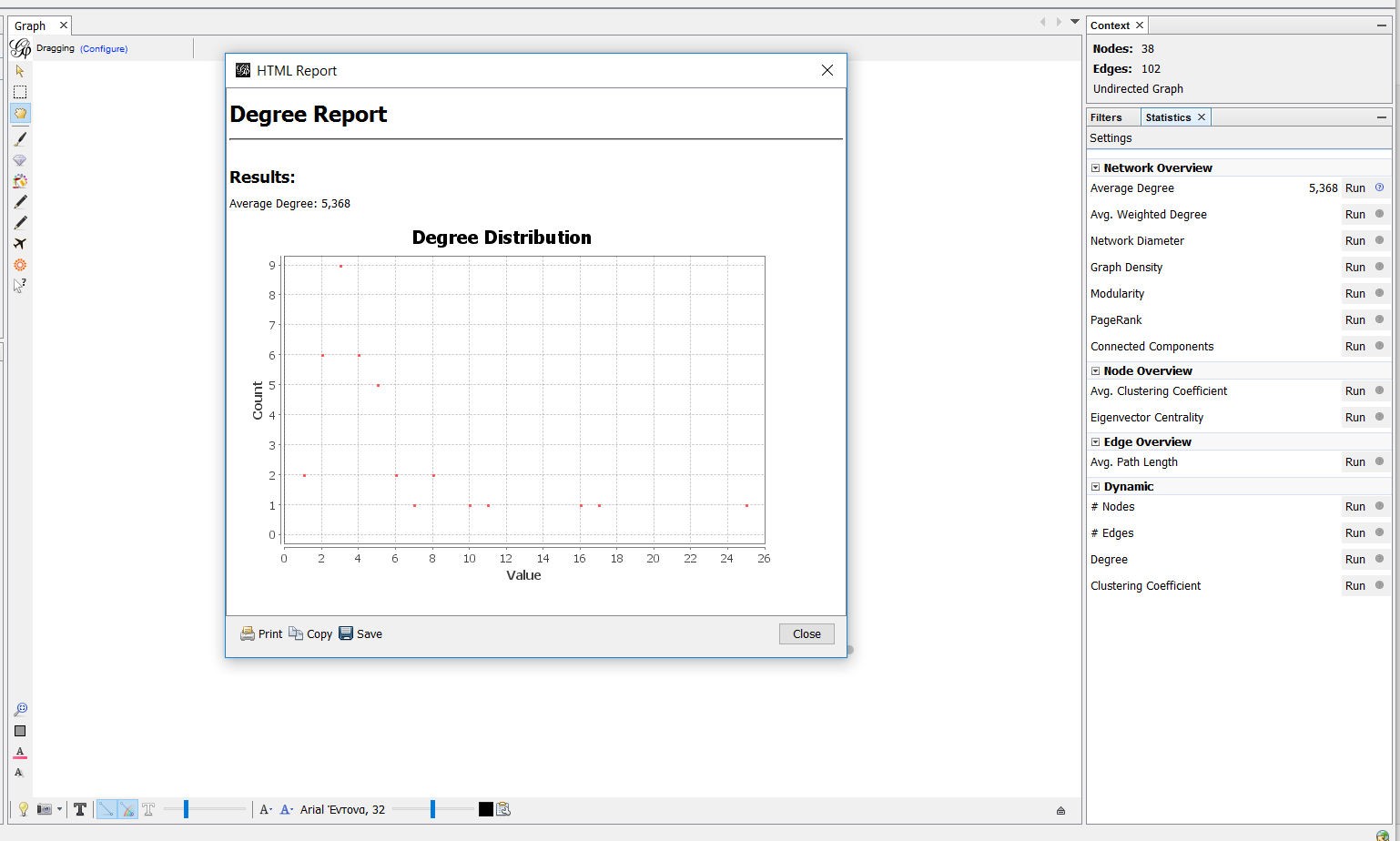
Gephi provides the following Graph metrics:

Network Overview: Average Degree, Average Weighted Degree, Network Diameter, Graph Density, Modularity, PageRank, Connected Components

Node Overview: Average Clustering Coefficient, Eigenvector Centrality Edge Overview: Average Path Length

Dynamic: #Nodes, #Edges, Degree, Clustering Coefficient

Example of calculating the “Average Degree” of the graph:



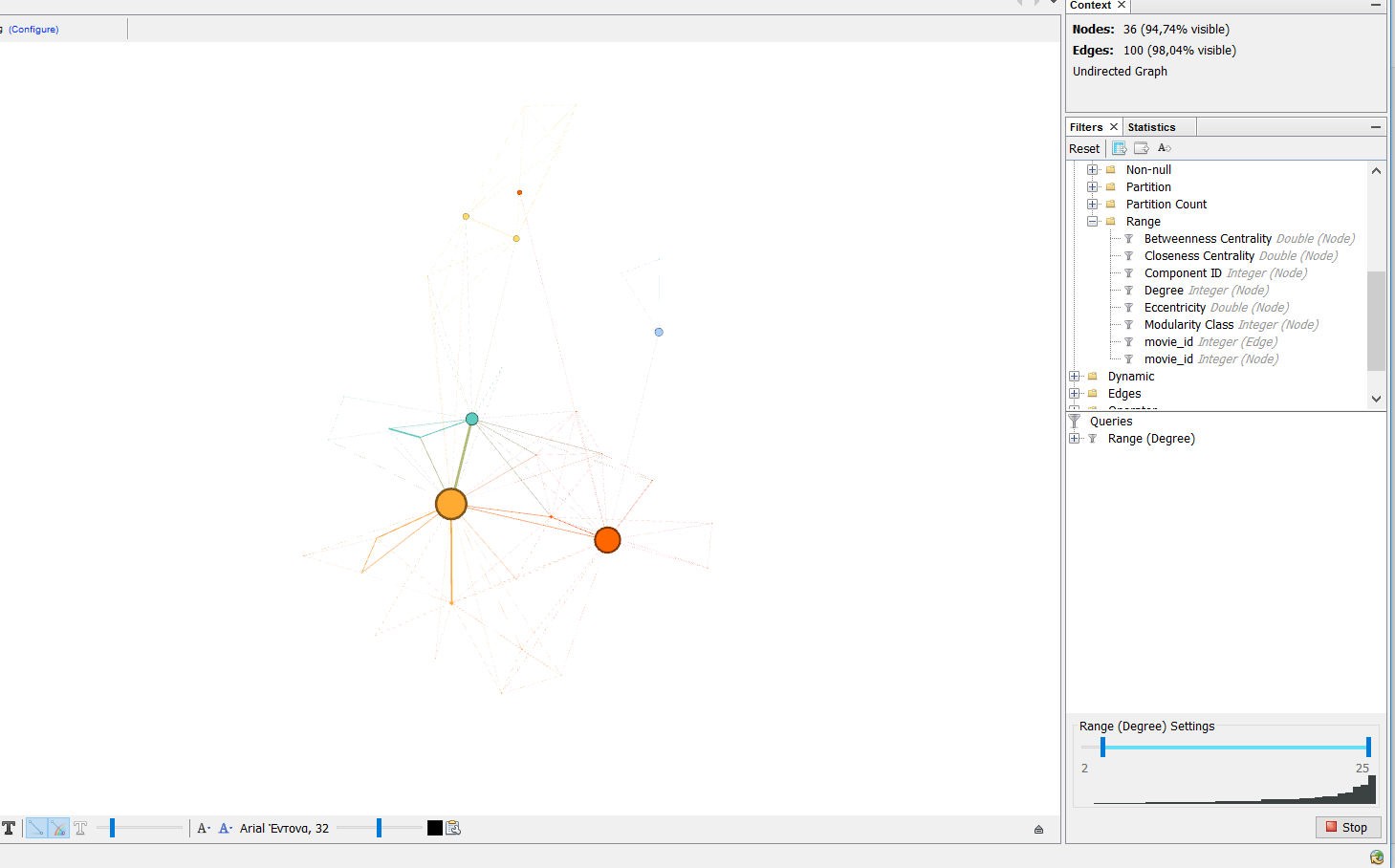
The graph which represents the distribution of the Average Degree can be printed,copied or saved.

## Filters

Το apply a filter the user must select it and drag it to the “Queries” module.

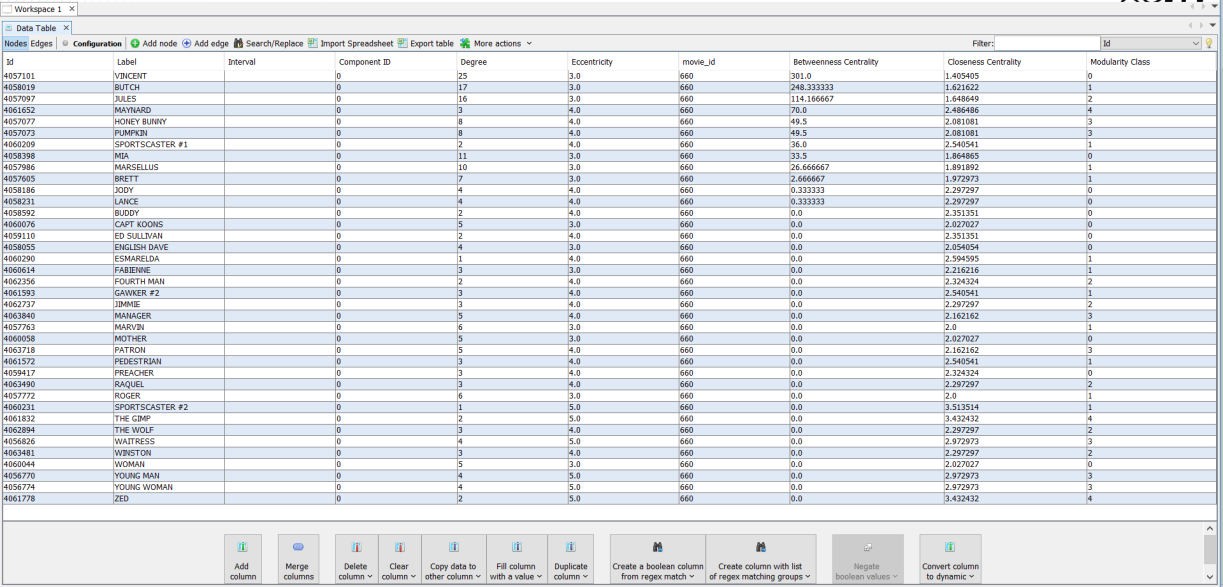
Example of applying a filter which removes every node that has a degree outside of the range [2,25], in other words it removes every leaf node.

On the bottom right side of the screen there is the Stop/Filter button which removes/applies the selected Filters. In the context module, we can see the number and percentage of the nodes and edges that did not get removed by the filter.



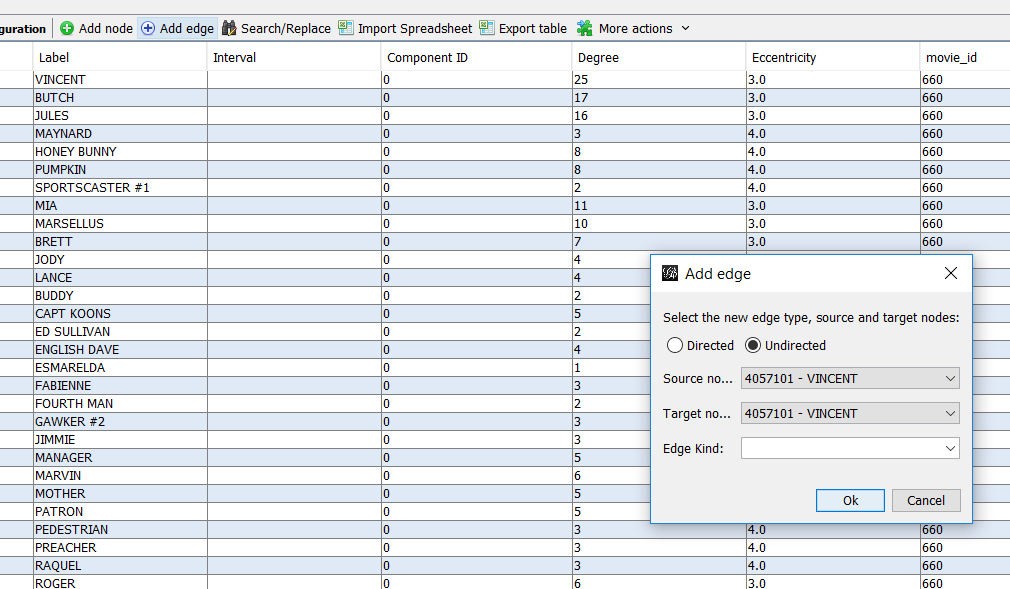
## Data Table

Here the data of the graph is presented in two tables, one for nodes and one for edges and can be edited by the user. This is the Nodes table, the user can change to

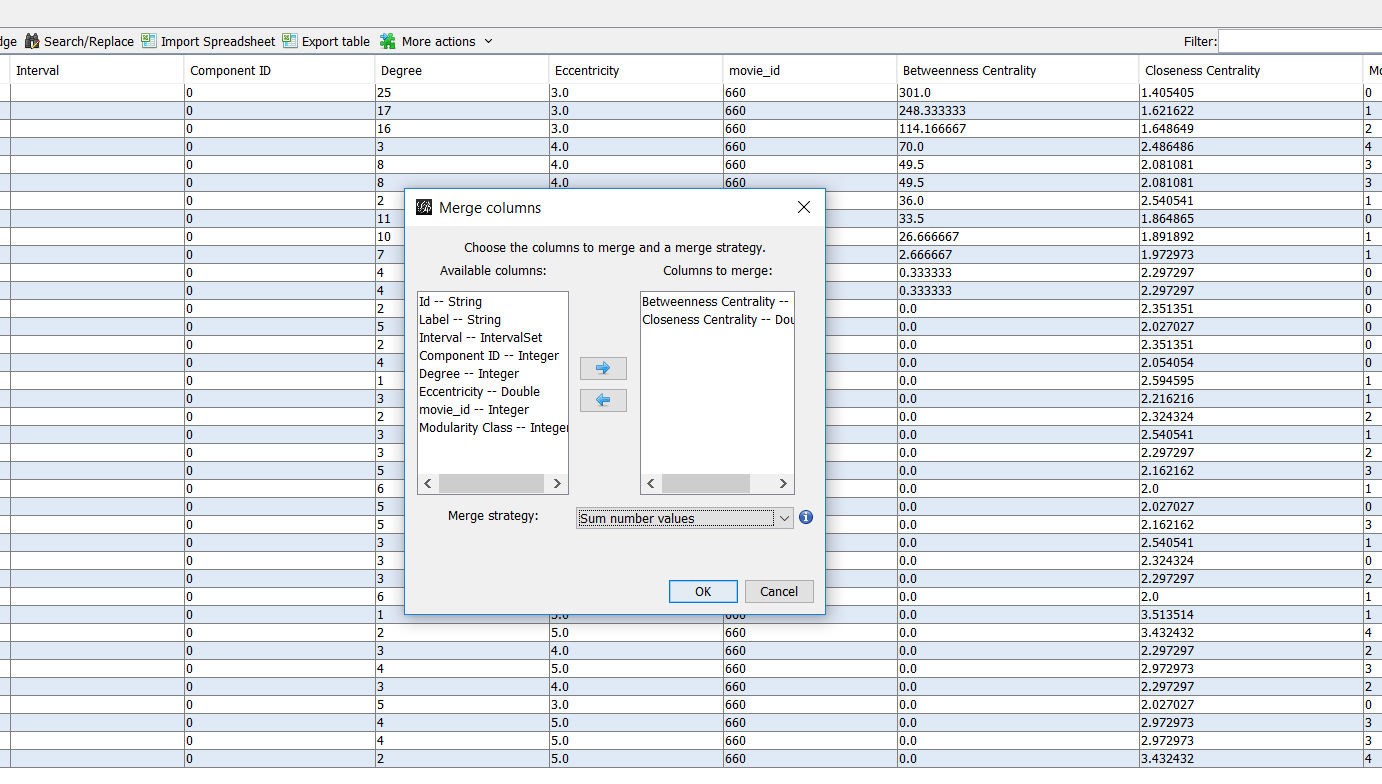


the edges table by pressing the “Edges” button on the top left hand side of the screen. The user can add new nodes/edges, edit existing ones, merge columns, create duplicates of columns etc.

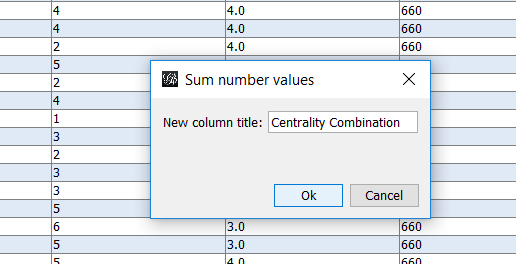
Example of adding a new Edge to the graph (by pressing the “Add edge” button):



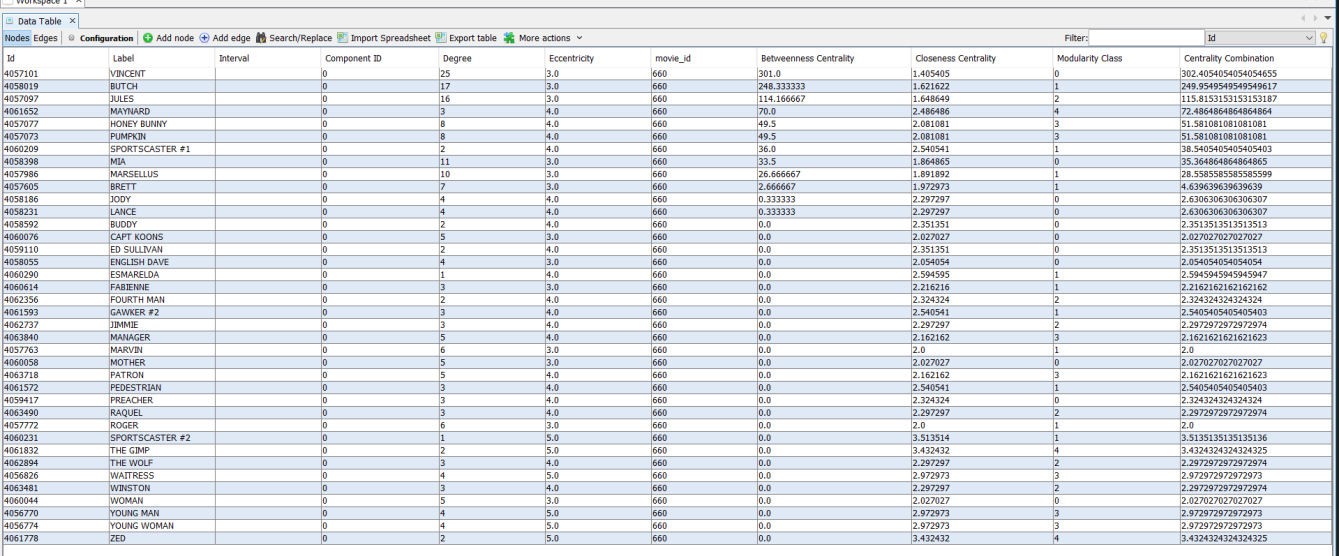
Example of merging two columns into one:



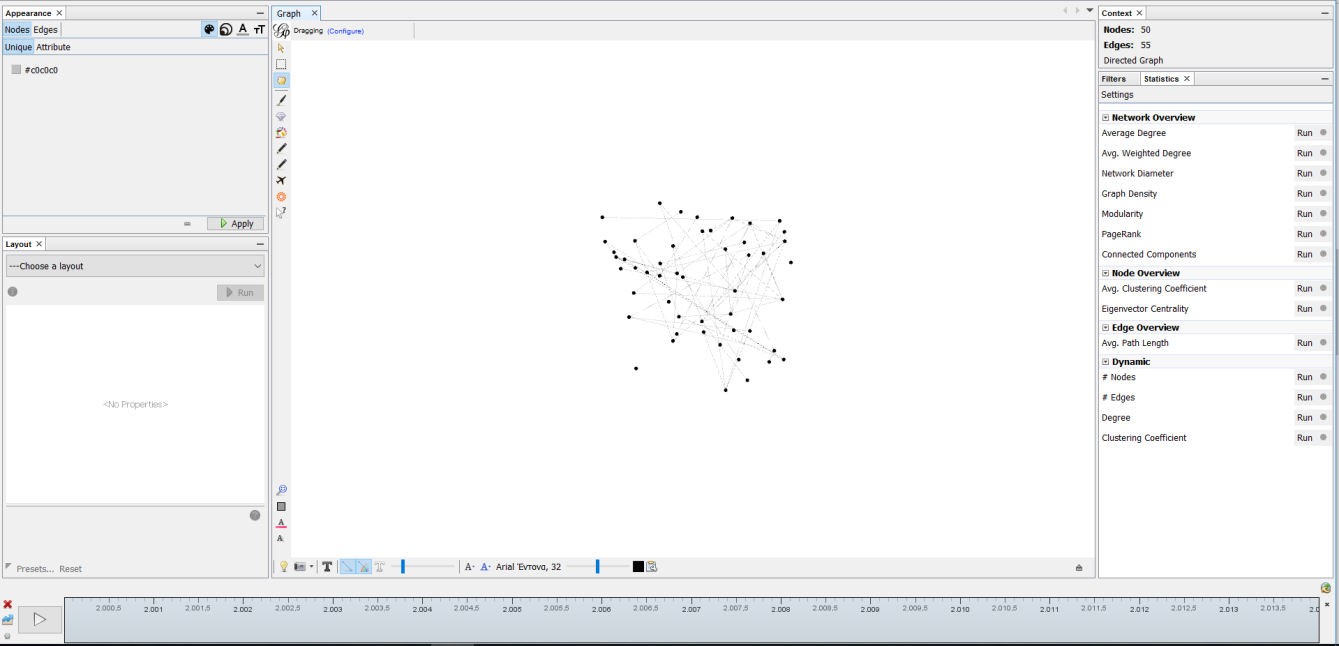
Choosing the new column’s title:



The new column (last one from the left) in the table:

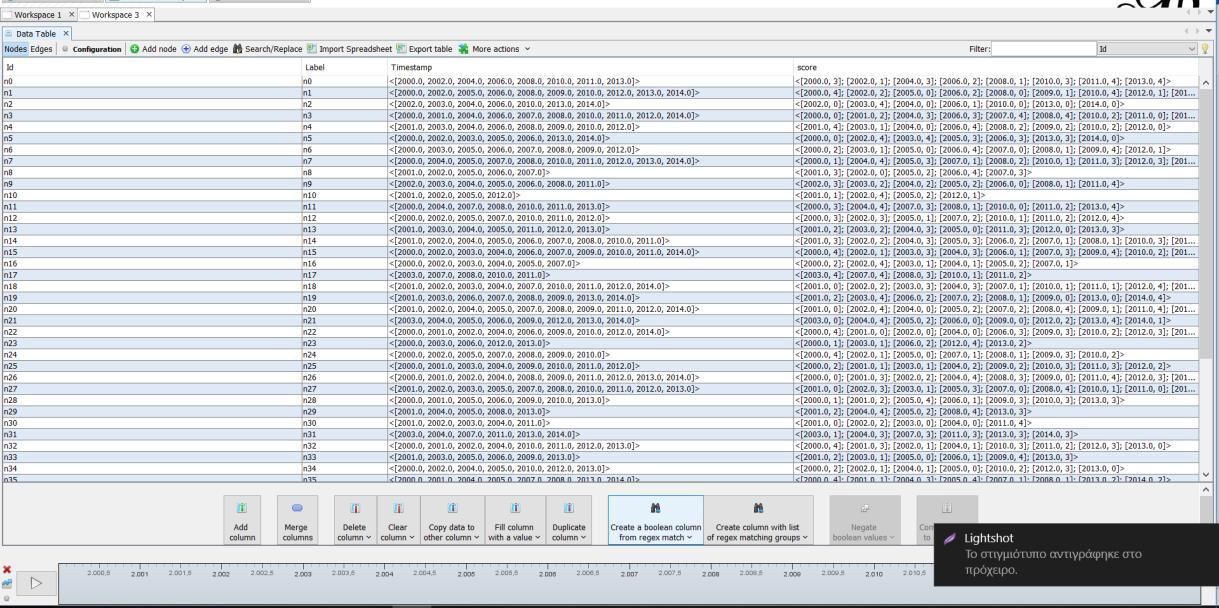


## Dynamic Graphs



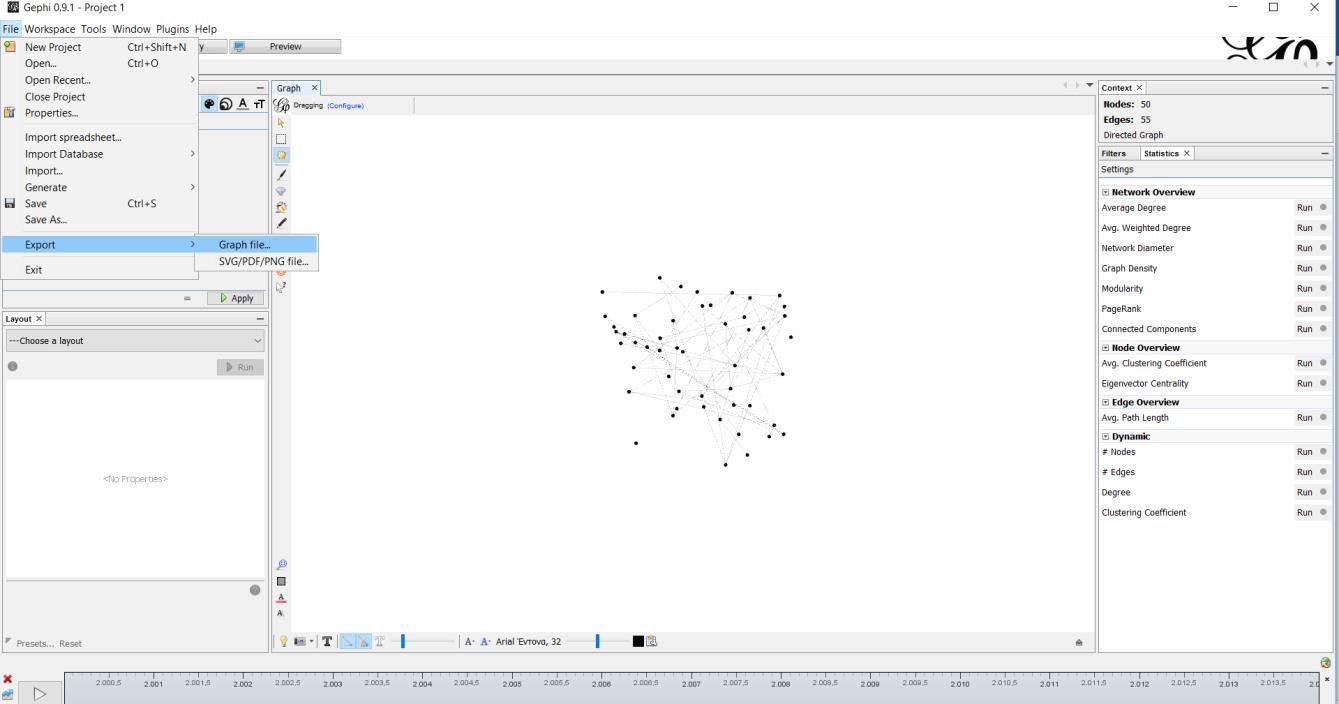
On the bottom side of the page we have the timeline bar, which can be used to demonstrate how the graph has changed at each timestamp. The user can select custom time intervals.

We can see in the data laboratory the appearance of the attribute Timestamp, which represents the time, and also the different values of the attribute score for every timestamp. Gephi also provides special metrics for dynamic graphs (for more information refer to section 4.3)

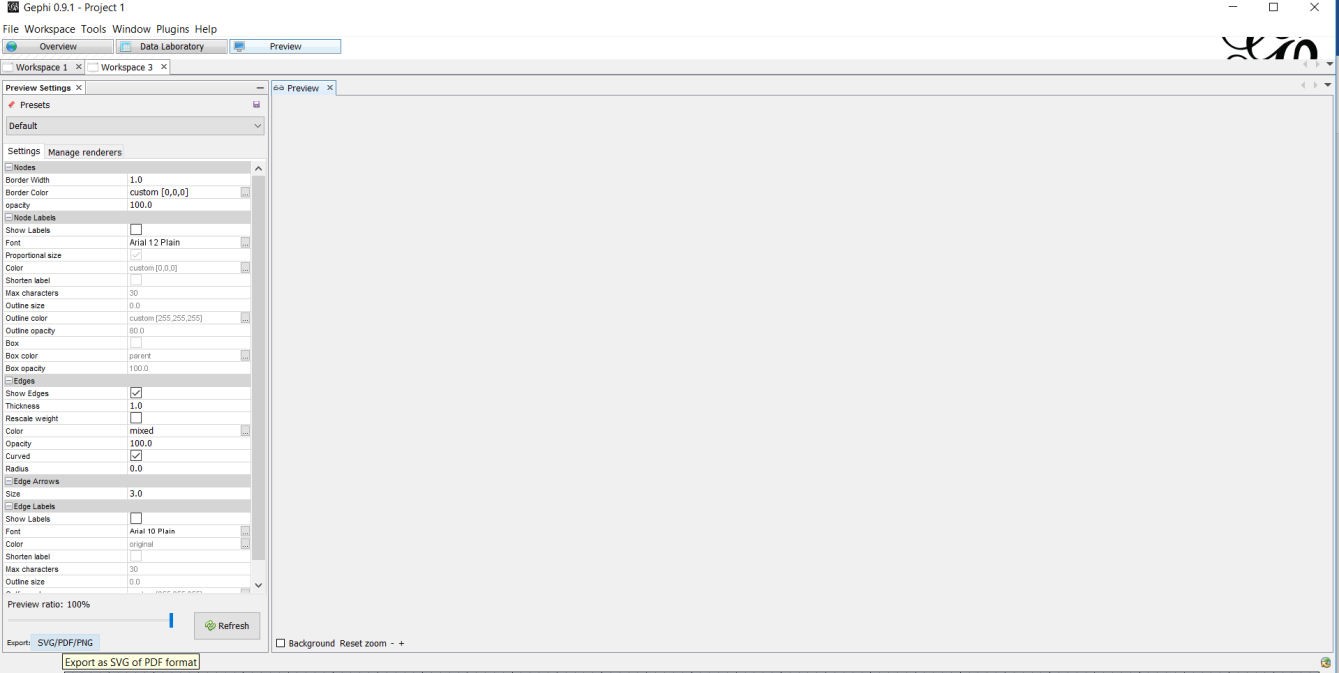


## Graph Export

The user can export the graph, either via the main toolbar



Or via the preview page:



The available output file formats are: SVG PDF and PNG (just for the graph it self) and also every supported graph format (such as gexf , which also includes the graph data)

# Other Nonfunctional Requirements

## Performance Requirements

Gephi requires a system with at least a 500 megahertz CPU and 128 megabytes of RAM and an OPENGL 1.2 compatible graphics card. However, these requirements can support effectively graphs of up to 1000 nodes and edges. Performance depends on the graph size and as a result, the system requirements for bigger graphs are more demanding. (precise figures: https://gephi.org/users/requirements/)

## Safety Requirements

To ensure that no one of Gephi’s users loses any data while using Gephi (due to a crash or a bug of some kind) the developer team updates Gephi regularly. There is a bug tracker available where users can report any bugs they have encountered so that the developers can fix it in the next release.

## Security Requirements

Gephi does not have any security requirements and thus any type of user can use it without any additional privileges.

## Software Quality Attributes

Gephi provides the users with both simple and advanced features. Due to its well designed and easy to use interface it can be used by both experts and typical users. However, users must already have a basic knowledge of graphs before using it.

# Glossary

References: **https://en.wikipedia.org/wiki/Main\_Page**

* + - Graph: a system of nodes connected in pairs by edges. Often subdivided

into directed graphs or undirected graphs according to whether the edges have an orientation or not. Mixed graphs include both types of edges.

* + - Node: A node is (together with edges) one of the two basic units out of which graphs are constructed. Nodes of graphs are often considered to be atomic objects, with no internal structure.
    - Edge: An edge is (together with vertices) one of the two basic units out of which graphs are constructed. Each edge has two (or in hypergraphs, more) vertices to which it is attached, called its endpoints. Edges may be directed or undirected; undirected edges are also called lines and directed edges are also called arcs or arrows. In an undirected simple graph, an edge may be represented as the set of its vertices, and in a directed simple graph it may be represented as an ordered pair of its vertices. An edge that connects

vertices *x* and *y* is sometimes written *xy*.

* + - Degree: the degree of a node of a graph is the number of edges incident to the node.
    - Weight: A graph structure can be extended by assigning a weight to each edge of the graph. Graphs with weights, or weighted graphs, are used to represent structures in which pairwise connections have some numerical values. For example, if a graph represents a road network, the weights could represent the length of each road.
    - Network(graph) Diameter: The diameter of a graph is the length of the shortest path between the most distanced nodes.
    - Graph Density: graph density is the ratio of the number of edges and the number of possible edges.
    - Modularity: Modularity is one measure of the structure of networks or graphs. It was designed to measure the strength of division of a network into modules (also called groups, clusters or communities).
    - PageRank is an algorithm used by Google Search to rank websites in their search engine results. Because the web is represented as a graph when PageRank is applied to it, it can be applied to every graph.
    - Connected component: a connected component (or just component) of

an undirected graph is a subgraph in which any two vertices are connected to each other by paths, and which is connected to no additional vertices in the supergraph.

* + - Clustering coefficient: clustering coefficient is a measure of the degree to which nodes in a graph tend to cluster together.
    - Centrality: In graph theory and network analysis, indicators of centrality identify the most important vertices within a graph.
    - Closeness Centrality: In a connected graph, the closeness

centrality (or closeness) of a node is a measure of centrality in a network, calculated as the sum of the length of the shortest paths between the node and all other nodes in the graph. Thus, the more central a node is, the closer it is to all other nodes.

* + - Betweenness Centrality: In graph theory, betweenness centrality is a measure of centrality in a graph based on shortest paths. For every pair of vertices in a graph, there exists a shortest path between the vertices such that either the number of edges that the path passes through (for undirected graphs) or the sum of the weights of the edges (for directed graphs) is minimized. The betweenness centrality for each vertex is the number of these shortest paths that pass through the vertex.
    - Dynamic Graph: An update on a graph is an operation that inserts or deletes edges or vertices of the graph or changes attributes associated with edges or vertices, such as cost or color. A dynamic graph is a graph that is subject to a sequence of updates.