DNA

DYNAMIC NETWORK ANALYZER

Graphical User Interface Instruction Manual

René Wilmes

25.04.2015

Dresden University of Technology

Contents

Chapter 1

Introduction

The graphical user interface of the DNA – Dynamic Network Analyzer allows to illustrate and compare data of dynamic networks. It is equipped with numerous features, whose functioning and use will be described during this manual. The interface might be executed in two different operation modes, which will be briefly explained in the following.

1.1 Playback-Mode

The Playback-Mode is used to illustrate previously generated data. Therefore the generation process has to be completed prior to the playback start. During this mode the interface offers additional functionality like setting the playback speed or the „timeslider“, which allows the user to move back and forth through the generated data.

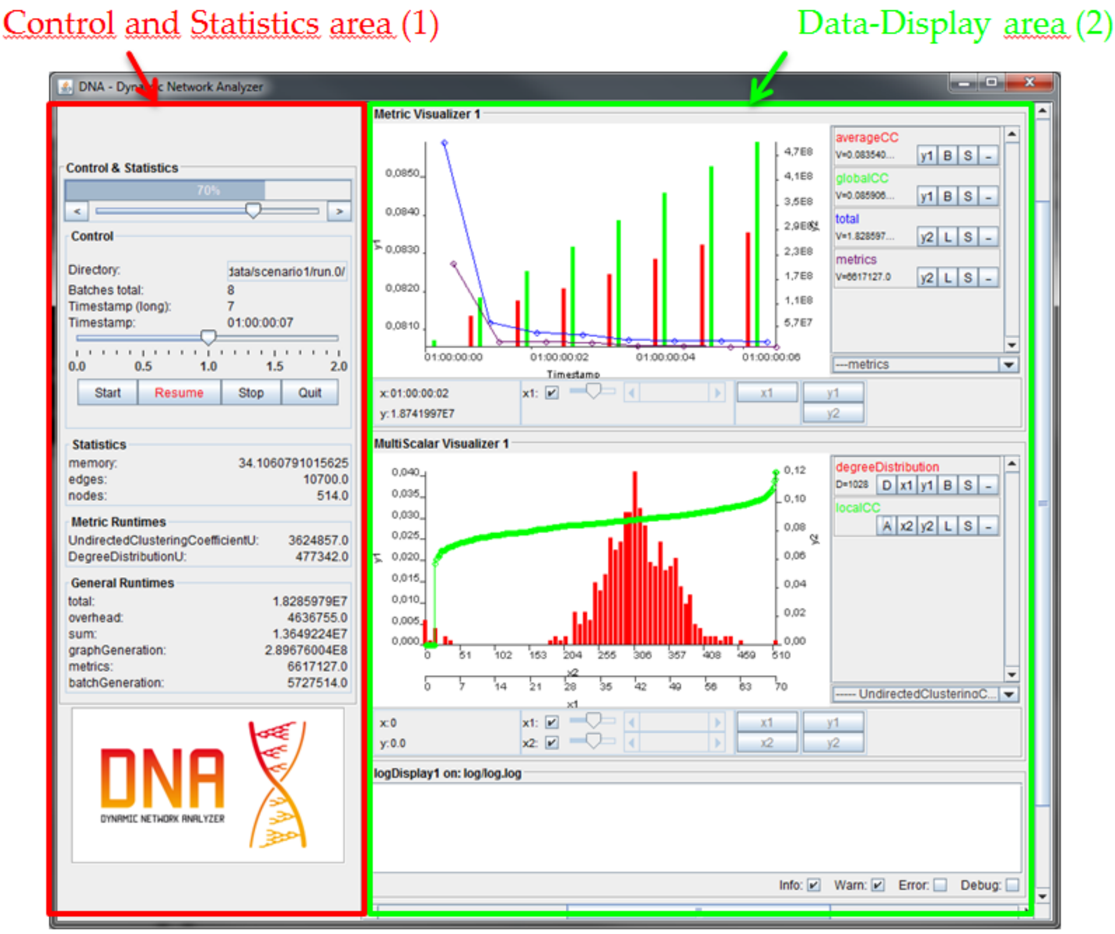
1.2 Live Display-Mode

The intend of this mode is to watch over the status and changes of a live-system. It uses a watchservice, which will automatically hand over any new batches found on the specified filesystem-path.

Chapter 2

User Interface Structure

The user interface is implemented in java swing and uses the jchart2d[jchart2d.sourceforge.net] library to display charts and plots. It is divided into two main areas, the control and statistics area (1), whose main purpose is to control the program and display selected statistics, and the data display area (2), which is used to illustrate data via charts and print events in log windows, see figure 2.1. The data display area can contain multiple instances of MetricVisualizers (2.2), MultiScalarVisualizers (2.3) and LogDisplays (2.4).



Directory: data/scenario1/run.0/
Batches total: 8
Timestamp (long): 7
Timestamp: 01:00:00:07

Statistics
memory: 34.1060791015625
edges: 10700.0
nodes: 514.0

Metric Runtimes
UndirectedClusteringCoefficientU: 3624857.0
DegreeDistributionU: 477342.0

General Runtimes
total: 1.8285979E7
overhead: 4636755.0
sum: 1.3649224E7
graphGeneration: 2.89676004E8
metrics: 6617127.0
batchGeneration: 5727514.0

V=0.083540...
V=0.085906...
V=1.828597...
V=6617127.0

x:01:00:00:02
y:1.8741997E7
x1: ☑

D=1028

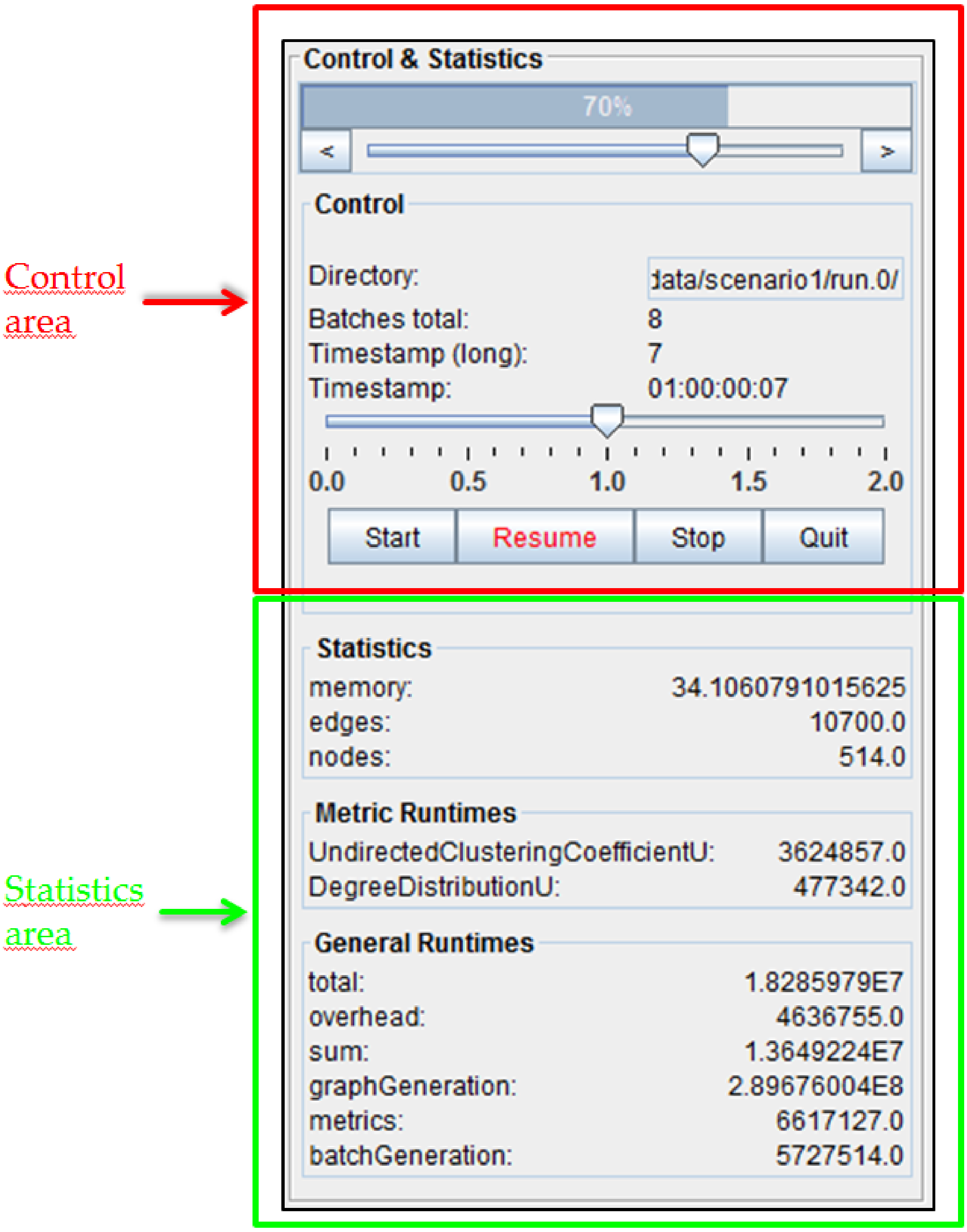
x:0
y:0.0
x1: ☑
x2: ☑

Info: ☑ Warn: ☑ Error: □ Debug: □

Figure 2.1 – Example of the two interface areas.

2.1 Control and Statistics Area

The control and statistics area is used to grant general control over the program and to display general statistics and runtimes, see figure 2.2. It can be divided into two smaller subareas, which will be discussed in the next two chapters.



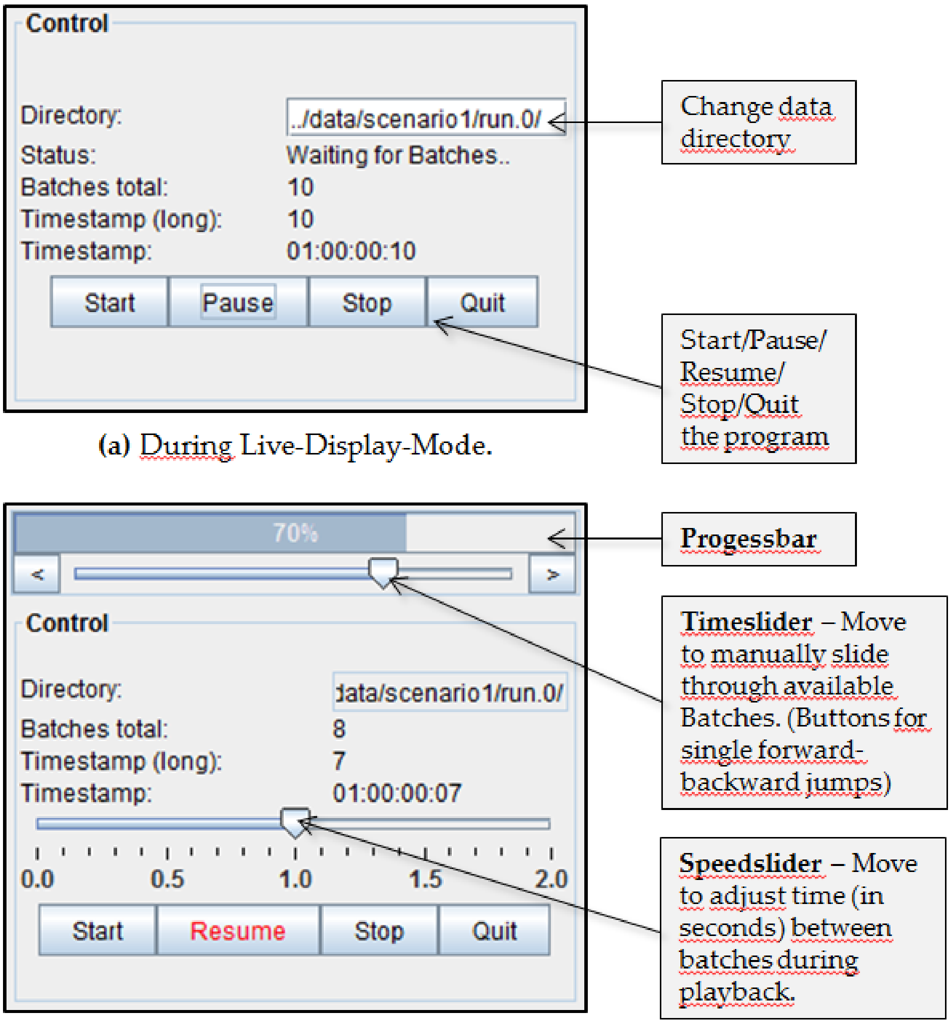
Directory: data/scenario1/run.0/
Batches total: 8
Timestamp (long): 7
Timestamp: 01:00:00:07

memory:	34.1060791015625
edges:	10700.0
nodes:	514.0
UndirectedClusteringCoefficientU:	3624857.0
DegreeDistributionU:	477342.0
total:	1.8285979E7
overhead:	4636755.0
sum:	1.3649224E7
graphGeneration:	2.89676004E8
metrics:	6617127.0
batchGeneration:	5727514.0

Figure 2.2 – Example of the Control and Statistics area during playback mode.

2.1.1 Control Area

Depending on the used mode, Live-Display or Playback, this area offers different functionalities to control the programs behaviour, see figure 2.3 (a) and (b).



Directory: ../data/scenario1/run.0/
Status: Waiting for Batches..
Batches total: 10
Timestamp (long): 10
Timestamp: 01:00:00:10

Change data directory

Start/Pause/ Resume/ Stop/Quit the program

(a) During Live-Display-Mode.

Directory: data/scenario1/run.0/
Batches total: 8
Timestamp (long): 7
Timestamp: 01:00:00:07

Progessbar

Timeslider – Move to manually slide through available Batches. (Buttons for single forward-backward jumps)

Speedslider – Move to adjust time (in seconds) between batches during playback.

(b) During Playback-Mode.

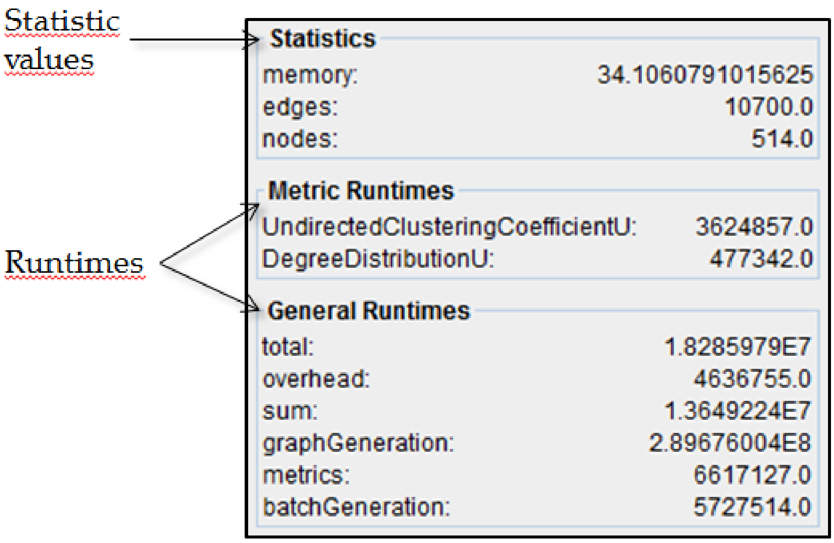
Figure 2.3 – Examples of the Control area during different modes.

2.1.2 Statistics Area

The statistics area is used to display statistics of the current batch. It can display statistic values aswell as metric and general runtimes. Which values will be displayed can be freely configured. For example, in Figure 2.4 the statistics memory, edges and nodes are being displayed, out of 21 possible values. This is done by defining the desired values in the configuration file and setting the ShowDefinedValues-Flag as true, which will display only the defined values. Setting it to false will result in hiding all defined values and showing all others that are available. See Figure 2.5 on the next page for the configuration code that produced the example in Figure 2.4.

2.2 Data-Display Area

The data display area is a flexible part of the interface, which allows the user to monitor the networks properities. It is freely configurable and can contain any amount



Statistics	
memory:	34.1060791015625
edges:	10700.0
nodes:	514.0
Metric Runtimes	
UndirectedClusteringCoefficientU:	3624857.0
DegreeDistributionU:	477342.0
General Runtimes	
total:	1.8285979E7
overhead:	4636755.0
sum:	1.3649224E7
graphGeneration:	2.89676004E8
metrics:	6617127.0
batchGeneration:	5727514.0

Figure 2.4 – Example of a statistics area.

of Metric Visualizers, MultiScalar Visualizers and LogDisplays, whose purpose will be discussed in the following chapters. Figure 2.6 shows an example with one object of each kind.

2.2.1 Metric Visualizer

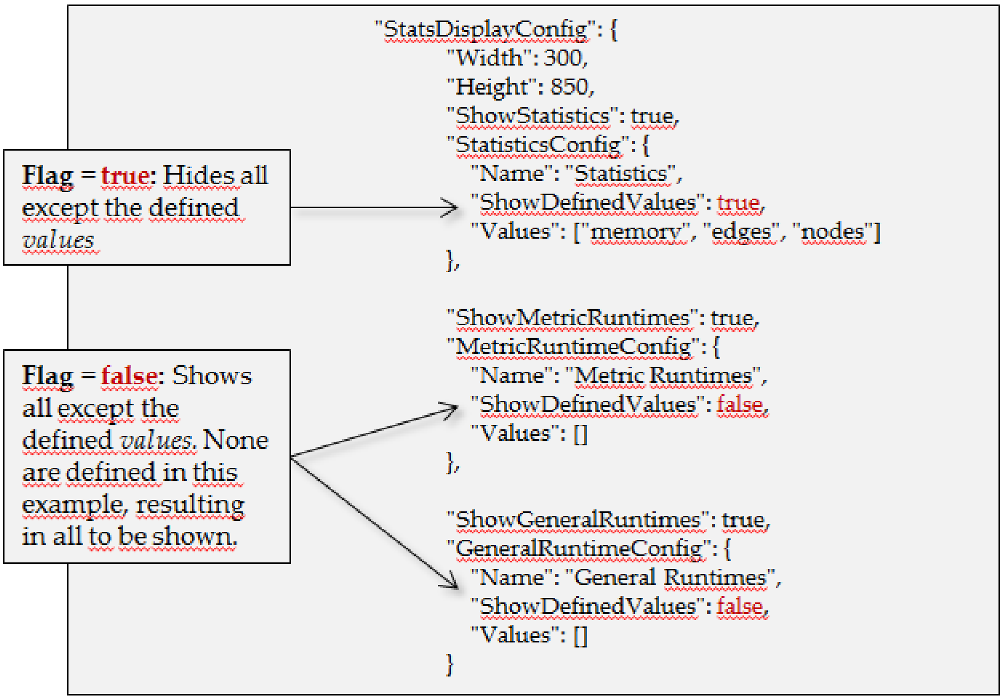
The MetricVisualizer is used to plot metric and statistic data in a chart. All currently shown values are printed in the legend and hold several display features like linespoint-/barplot or switching between y-axis. For example in figure 2.7 the value „averageCC“ is plottet as a bar on the y1-axis. New values can be added via the dropdown menu below the legend. The MenuBar allows for additional features for the chart, e.g. show grids for the respective axis or to zoom in on specific intervals of the plot. Which values will be displayed on default can be configured in the configuration file.

2.2.2 MultiScalar Visualizer

The MultiScalarVisualizer is very similar to the MetricVisualizer. It is used to plot multi scalar values like distributions or nodevaluelists. Additional features are the second x-axis and sorting of the plotted values. For example in figure 2.8, the nodevaluelist „localCC“ is sorted in ascending order, while the distribution „degreeDistribution“ is plotted as a regular distribution. By clicking on the „D“, it could also be displayed as a CDF.

2.2.3 Log Display

The LogDisplay components are used to tail log files and print relevant informations written to the log. They can filter between four different log-level messages: Info, Warn, Error and Debug, see figure 2.9.

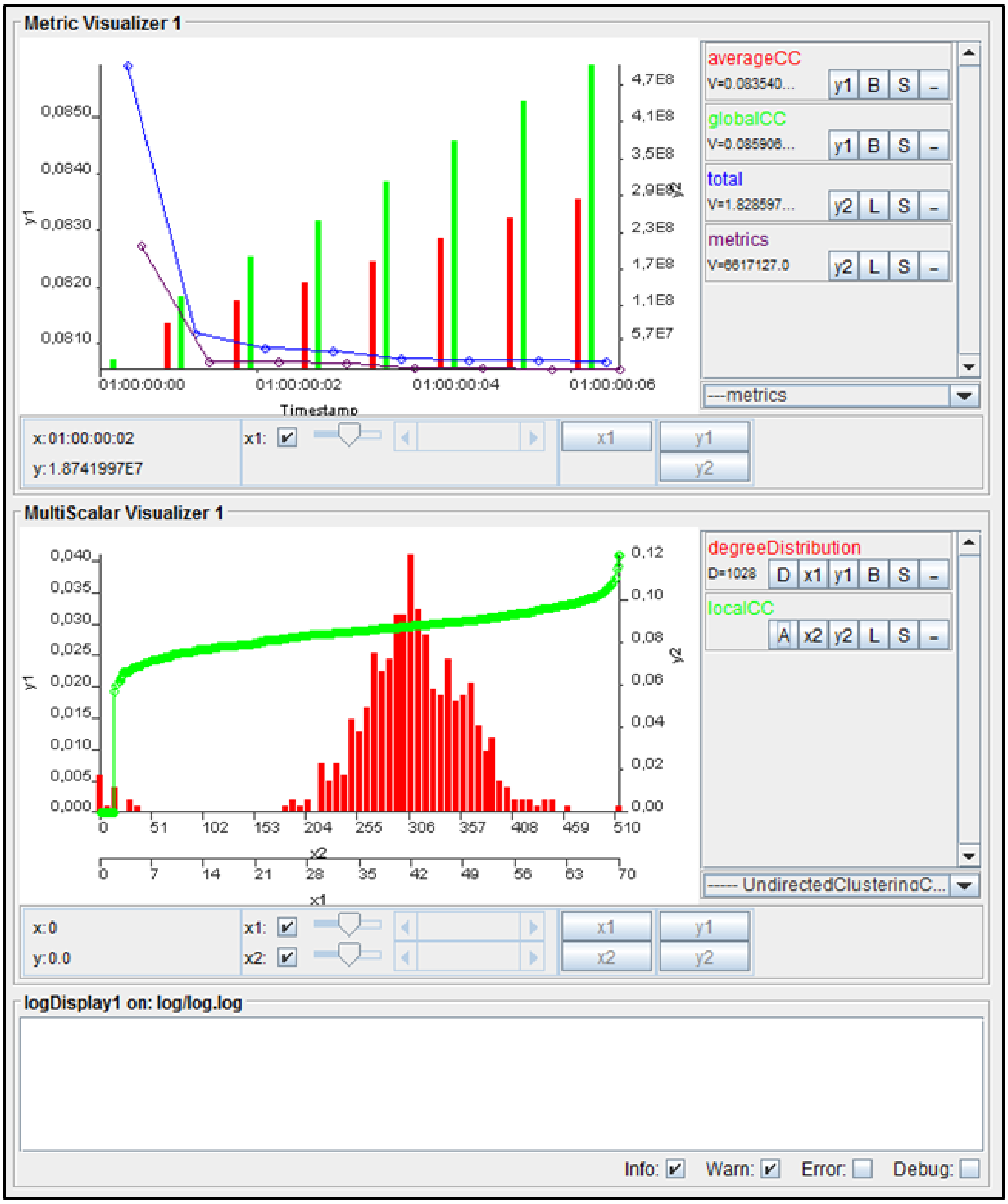


```
"StatsDisplayConfig": {
        "Width": 300,
        "Height": 850,
        "ShowStatistics": true,
        "StatisticsConfig": {
          "Name": "Statistics",
          "ShowDefinedValues": true,
          "Values": ["memory", "edges", "nodes"]
        },

        "ShowMetricRuntimes": true,
        "MetricRuntimeConfig": {
          "Name": "Metric Runtimes",
          "ShowDefinedValues": false,
          "Values": []
        },

        "ShowGeneralRuntimes": true,
        "GeneralRuntimeConfig": {
          "Name": "General Runtimes",
          "ShowDefinedValues": false,
          "Values": []
        }
```

Flag = **true**: Hides all except the defined *values*

Flag = **false**: Shows all except the defined *values*. None are defined in this example, resulting in all to be shown.

Figure 2.5 – Configuration for the Control and Statistics area shown in figure 2.4.



Metric Visualizer 1

averageCC V=0.083540...

globalCC V=0.085906...

total V=1.828597...

metrics V=6617127.0

---metrics

x: 01:00:00:02
y: 1.8741997E7

x1: ☑

MultiScalar Visualizer 1

degreeDistribution D=1028

localCC

----- UndirectedClusteringC...

x: 0
y: 0.0

x1: ☑
x2: ☑

logDisplay1 on: log/log.log

Info: ☑ Warn: ☑ Error: □ Debug: □

Figure 2.6 – Example of the Data-Display Area.

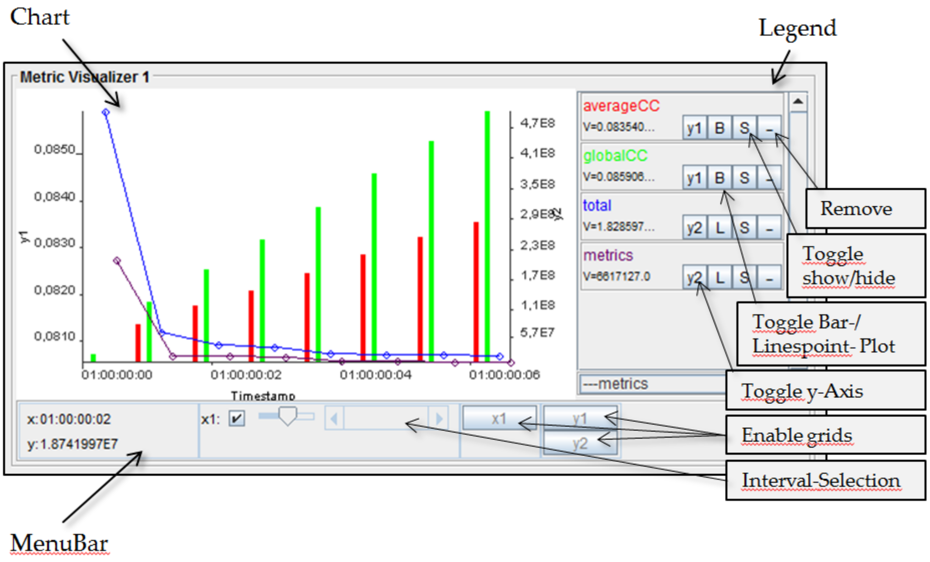


Figure 2.7 – Example of a MetricVisualizer.

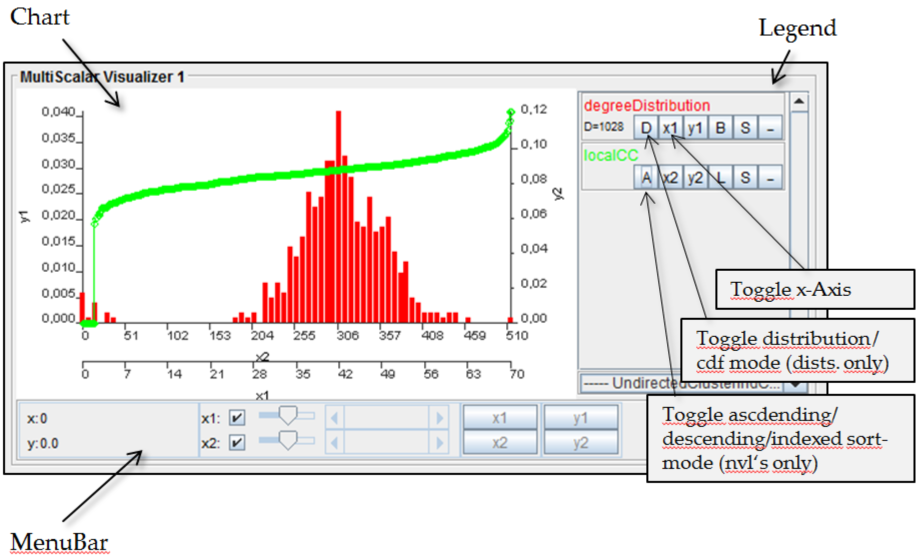
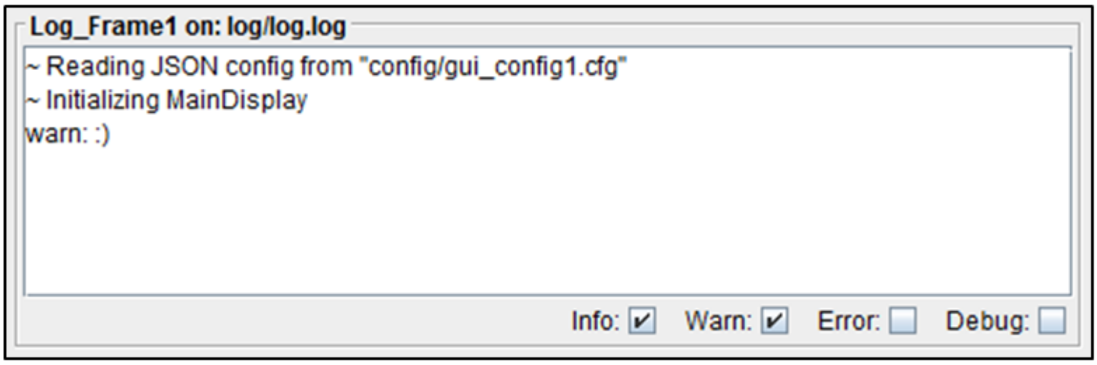


Figure 2.8 – Example of a MultiScalar Visualizer



```
~ Reading JSON config from "config/gui_config1.cfg"
~ Initializing MainDisplay
warn: :)
```

Figure 2.9 – Example of a LogDisplay tailing "log/log.log".

Chapter 3

Configuration

The user interface is designed to be flexible and configurable in the most possible aspects. Configuration Files are written in JSON[json.org] and can be used to adapt the interface to the own specific needs. The general configuration structure is shown in figure 3.1. How to work with the interface and create configuration files will be explained in the following.

3.1 Default-Configuration File

The Default-Configuration file located in ”config/gui_default.cfg” holds default values for each type of object and almost every possible property. It gives a good example on how a configuration should look like, because it is important to keep the hierarchical structure. When creating own configurations, left out values will be automatically replaced with the values set in the default file, thus allowing to craft own configurations easily and with little effort. Note that any changes on the default file can prevent the interface from running, for example removing values that are needed as default references.

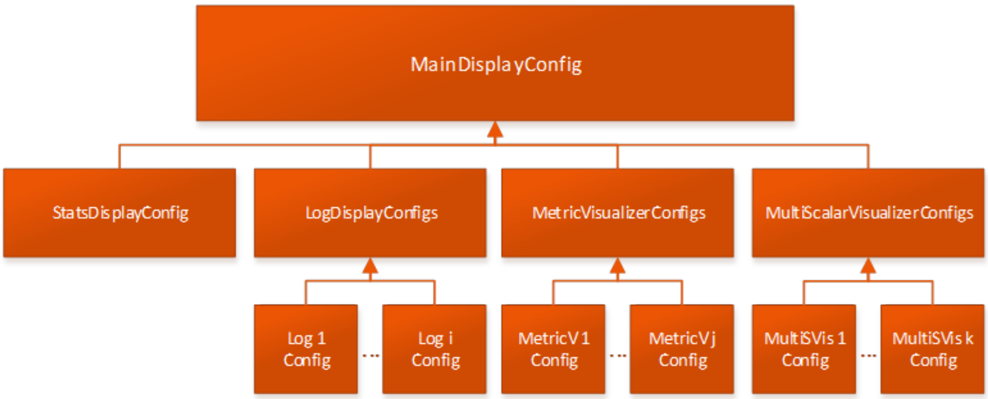
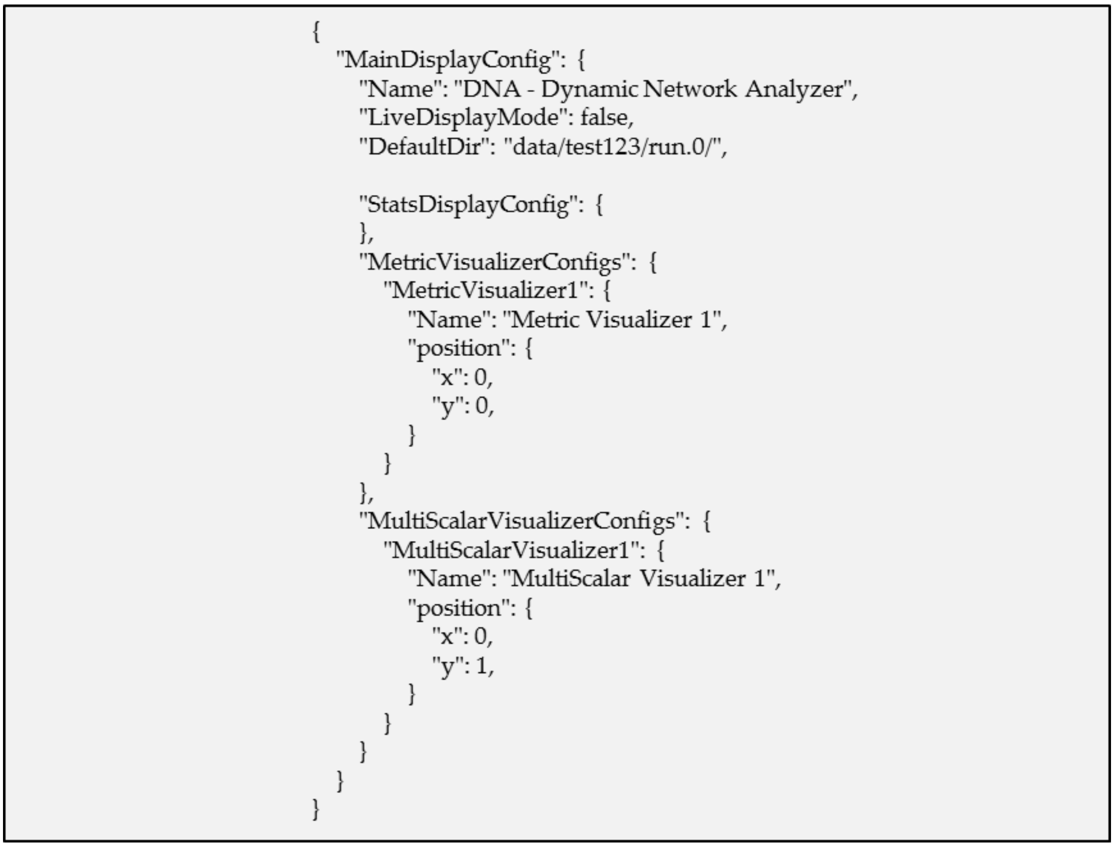


Figure 3.1 – Configuration Structure.

3.2 Building own configuration files

When creating an own configuration file it is recommended to take a look at the minimal display configuration located in ”config/gui_min.cfg” and to expand it for your own needs. It uses only the minimum of values needed to create a useful interface, see figure 3.2. It creates a MainDisplay (1) in Playback-Mode (1.1) checking the relative directory ”data/test123/run.0/” (1.2). It consists of a StatsDisplay (2), a MetricVisualizer (3) and a MultiScalarVisualizer (4). For the latter two, only the names and positions are defined. See figure 3.3 on the next page for the resulting Interface. For more sophisticated settings check the ”config/displayConfig.cfg”.



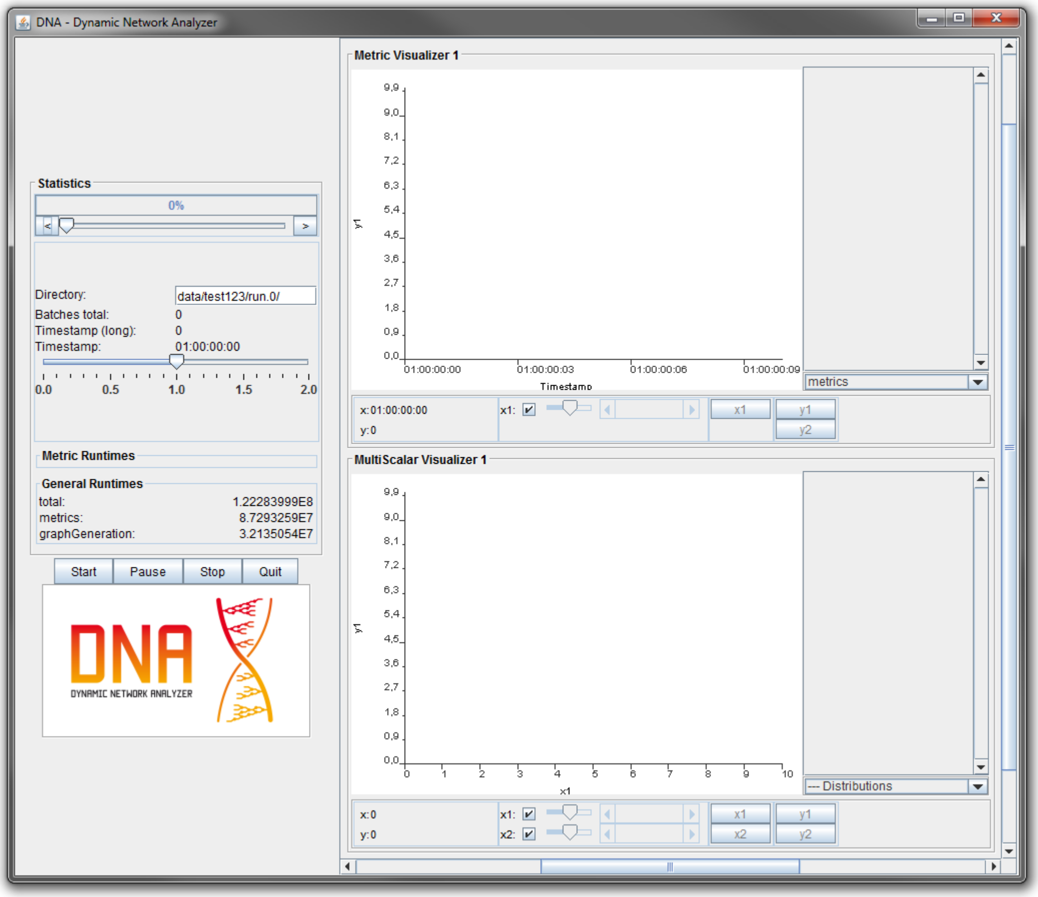
```
{
   "MainDisplayConfig": {
      "Name": "DNA - Dynamic Network Analyzer",
      "LiveDisplayMode": false,
      "DefaultDir": "data/test123/run.0/",

      "StatsDisplayConfig": {
      },
      "MetricVisualizerConfigs": {
         "MetricVisualizer1": {
            "Name": "Metric Visualizer 1",
            "position": {
               "x": 0,
               "y": 0,
            }
         }
      },
      "MultiScalarVisualizerConfigs": {
         "MultiScalarVisualizer1": {
            "Name": "MultiScalar Visualizer 1",
            "position": {
               "x": 0,
               "y": 1,
            }
         }
      }
   }
}
```

Figure 3.2 – Minimal-Configuration-File (”config/gui_min.cfg”).

3.2.1 Adjusting the MetricVisualizer

In the following example we will extend the minimal configuration from the previous chapter by adjusting the MetricVisualizer. The ”displayConfig.cfg” configuration file helps us to see what values can be adjusted to fulfill our needs. First we might always want the amount of nodes and edges to be displayed in the chart on the second y-axis. This is done by adding a VisualizerConfig (1) object, which itself holds two SingleConfigs (2) & (3), one for each value. See figure 3.4, the added code is highlighted in red.

Now we might want to add a specific type of value to be added to the chart by default on startup. This is done by adding „GeneralConfigs“ to the previous defined



Directory: data/test123/run.0/
Batches total: 0
Timestamp (long): 0
Timestamp: 01:00:00:00
0.0 0.5 1.0 1.5 2.0

Metric Runtimes

General Runtimes

total:	1.22283999E8
metrics:	8.7293259E7
graphGeneration:	3.2135054E7

x: 01:00:00:00
y: 0
x1: ☑

x: 0
y: 0
x1: ☑
x2: ☑

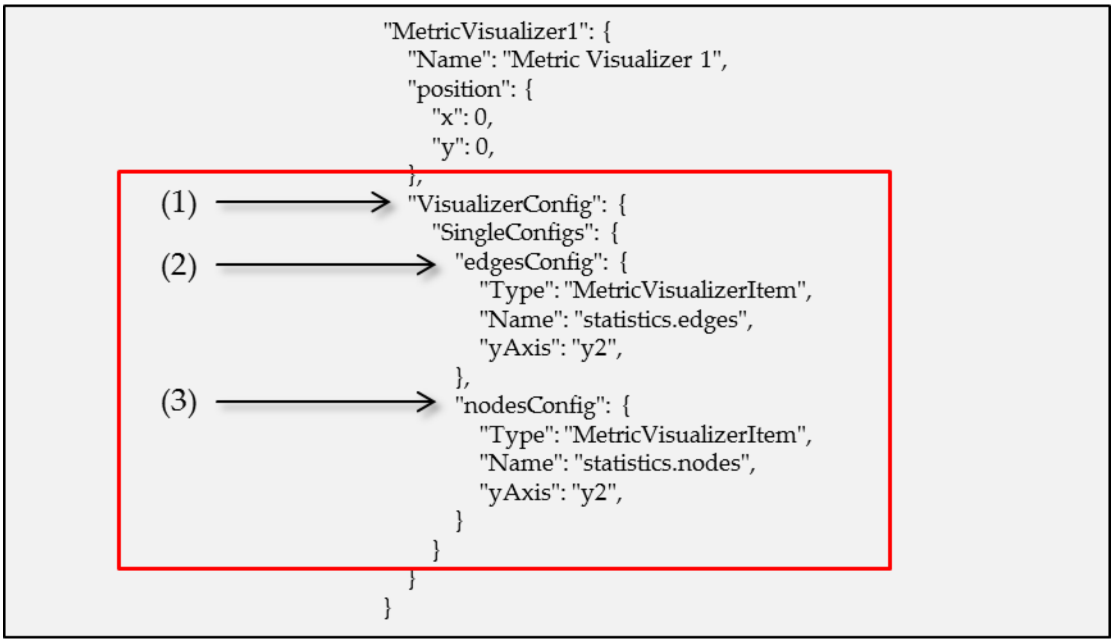
Figure 3.3 – Resulting Interface from the Minimal-Configuration-File ("config/gui_min.cfg").

VisualizerConfig object. They are applied for each value of a specific kind. Possible general configs are "generalMetricConfig", "generalGeneralRuntimesConfig", "generalStatisticsConfig", "generalMetricRuntimesConfig"", "generalDistributionConfig", and "generalNodeValueListConfig". For example to add all available metrics by default, we add a "generalMetricConfig" (1). Figure 3.5 shows which code has been added to plot all metrics as linespoint (2) on the left y-Axis in hidden-mode (3). This allows the user to have all metrics available in the chart and to simply show those he wants with a single mouse-click on the red "H". Figure 3.6 on the next page shows the resulting MetricVisualizer.

Note: All values that are added to the legend are existent in the chart, even if they are in hidden mode. Therefore the axis scaling and offsets might interfere with the expected outcome of your configuration. The MultiScalar Visualizer can be adjusted analogously to the MetricVisualizer.

3.2.2 Positioning in Data-Display Area

The data display area can be pictured as a grid, see figure 3.7. By default each object will be assigned to one square of the grid, regarding it's (x,y)-coordinates. Additionally it is possible to set row-/colspan values for each object, allowing them to fill more



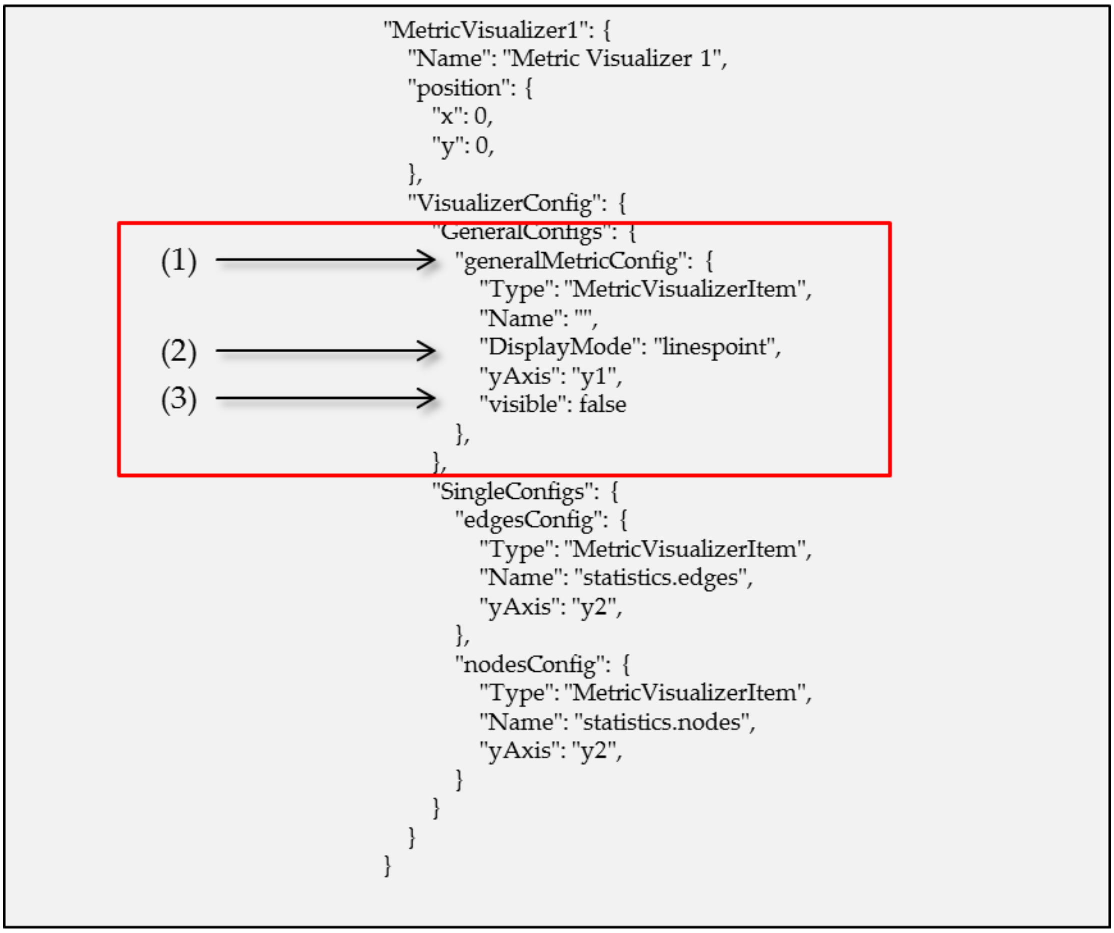
```
"MetricVisualizer1": {
  "Name": "Metric Visualizer 1",
  "position": {
    "x": 0,
    "y": 0,
  },
  "VisualizerConfig": {
    "SingleConfigs": {
      "edgesConfig": {
        "Type": "MetricVisualizerItem",
        "Name": "statistics.edges",
        "yAxis": "y2",
      },
      "nodesConfig": {
        "Type": "MetricVisualizerItem",
        "Name": "statistics.nodes",
        "yAxis": "y2",
      }
    }
  }
}
```

Figure 3.4 – Extended MetricVisualizer example.

than one row or column. Table 3.1 shows which configuration is needed to build the area from figure 3.7. Note that the grid-sizes are relative and will be dynamically adapted to the largest object in the respective row / column.

Object	X	Y	Rowspan	Colspan
MetricVisualizer 1	0	0	1	1
MultiScalar Visualizer 1	1	0	1	1
Log_Frame1	0	1	1	2

Table 3.1 – Objects and their properties from the example in figure 3.7



```
"MetricVisualizer1": {
  "Name": "Metric Visualizer 1",
  "position": {
    "x": 0,
    "y": 0,
  },
  "VisualizerConfig": {
    "GeneralConfigs": {
      "generalMetricConfig": {
        "Type": "MetricVisualizerItem",
        "Name": "",
        "DisplayMode": "linespoint",
        "yAxis": "y1",
        "visible": false
      },
    },
    "SingleConfigs": {
      "edgesConfig": {
        "Type": "MetricVisualizerItem",
        "Name": "statistics.edges",
        "yAxis": "y2",
      },
      "nodesConfig": {
        "Type": "MetricVisualizerItem",
        "Name": "statistics.nodes",
        "yAxis": "y2",
      }
    }
  }
}
```

Figure 3.5 – Further extended MetricVisualizer example.

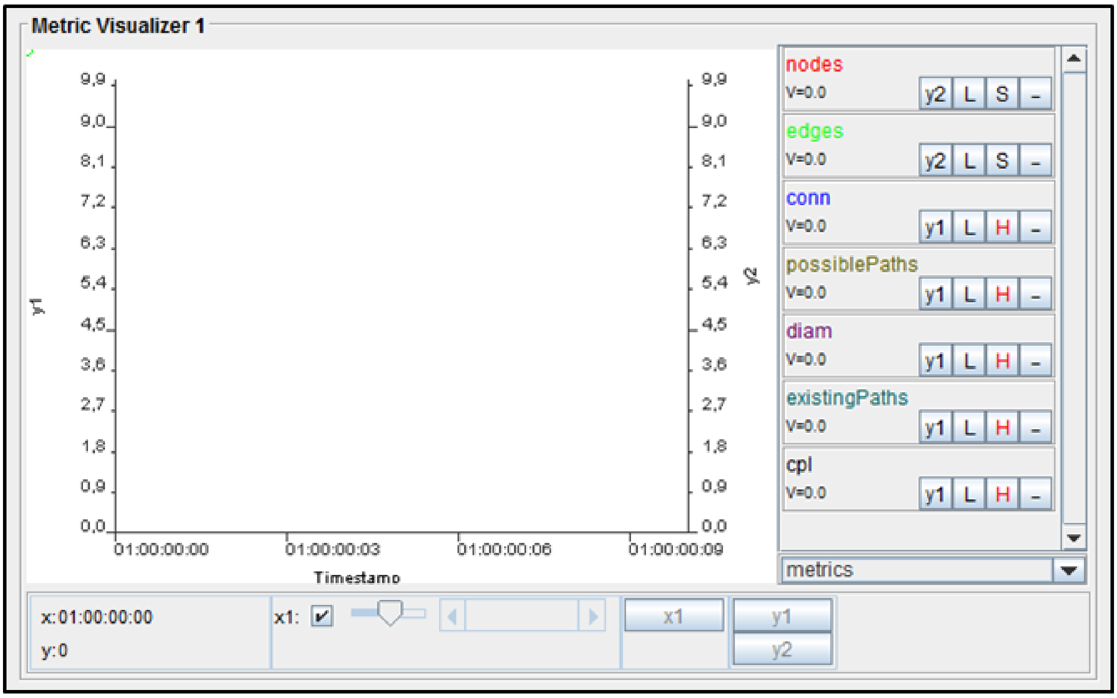
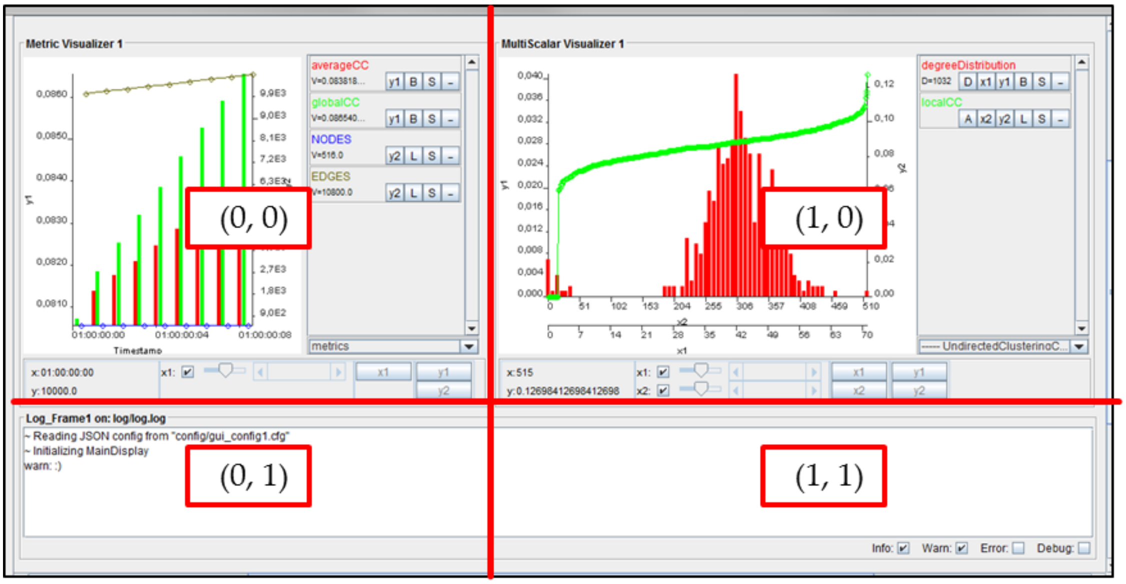


Figure 3.6 – Resulting Interface from the further extended MetricVisualizer example.



```
~ Reading JSON config from "config/gui_config1.cfg"
~ Initializing MainDisplay
warn: :)
```

Figure 3.7 – The data display grid.

Chapter 4

Commandline-Parameters

The program can be started with different commandline-parameters. The syntax is as follows:

```
java –jar dna–vis.jar    [–c <config–path>] [–d <data–dir>] [–h] [–l] [–p] [–z] [–zr]
```

Parameter	Description
–c <config–path>	Used to give the program a path to own configuration file. If left out, the default-config will be used.
–d <data–dir>	Overwrites the default data-directory given in the configuration file.
–h	Displays the help message.
–l	Launches the GUI in Livedisplay-Mode.
–p	Launches the GUI in Playback-Mode.
–z	Enables zipped batches.
–zr	Enables zipped runs. Only available in PlayBack mode!

Table 4.1 – Explanation of the commandline-parameters.

4.1 Examples

```
java –jar dna–vis.jar –h
java –jar dna–vis.jar –c ”config/min_cfg.cfg”
java –jar dna–vis.jar –l –d ”data/test1337/run.5/” –z
java –jar dna–vis.jar –d ”data/dd/run.0”
java –jar dna–vis.jar –c ”config/gui_min.cfg” –d ”data/cc/run.42/” –p –z
```