

SimGraphVisualizer

User guide

working draft

14.12.2023

by Dariusz Pojda

dpojda@iitis.pl

Introduction

SimGraphVisualizer is a library developed in JavaScript designed specifically for the purpose of visualizing and editing Ising graphs. The primary advantage of using JavaScript is that it allows SimGraphVisualizer to operate directly within a web browser. This eliminates the need for a dedicated or powerful server to run the library.

Running in a Web Browser and as a Desktop Application

When using SimGraphVisualizer in a web browser, you can simply open the 'index.html' file that comes with the library in the 'public_html' directory. No additional setup is needed for this mode of operation. Your browser will handle the execution of the JavaScript code, rendering the Ising graphs directly on your web page. This mode is ideal for quick visualizations and edits without any need for server-side computation.

For those who require or prefer a standalone desktop application, SimGraphVisualizer also supports compilation into a desktop executable. This is achieved using the ElectronJS library in conjunction with the Chromium engine. The ElectronJS framework provides the necessary tools to package your JavaScript code into a standalone application that can run independently of a web browser. The Chromium engine serves as the underlying platform for rendering the application's interface and executing the JavaScript code.

To compile SimGraphVisualizer into a desktop application, you will need to have Node.js and npm (Node Package Manager) installed on your system. After installing these prerequisites, you can navigate to the root directory of the SimGraphVisualizer library and run specific npm commands to initiate the compilation process. Detailed instructions on these steps are available in the 'Download and Installation Guide'.

By offering both browser-based and desktop application functionalities, SimGraphVisualizer aims to provide a versatile toolset for anyone working with Ising graphs.

What are Ising Models?

The Ising model is named after Ernst Ising, who studied it as a doctoral student in the 1920s. It's one of the simplest models to describe ferromagnetism, but it has found applications in a variety of fields, from neural networks to optimization problems.

The model consists of a lattice (often a regular grid, but can be more general) where each site on the lattice has a spin that can be either up (+1) or down (-1). The energy of a configuration of spins is typically given by:

$$E = -J \sum_{\langle i, j \rangle} s_i s_j - h \sum_i s_i$$

Where:

- $\langle i, j \rangle$ denotes pairs of neighboring sites.
- s_i and s_j are the spins at sites i and j .
- J is the interaction strength between spins. If $J > 0$, it's a ferromagnetic interaction (spins prefer to align). If $J < 0$, it's an antiferromagnetic interaction (spins prefer to be opposite).
- h is an external magnetic field.

The challenge in the Ising model is often to find the ground state, i.e., the configuration of spins with the lowest energy, especially for certain complex or large lattices.

Quantum Computing and Ising Model

The Ising model and its quantum counterpart, the quantum Ising model, have gained significant attention in the realm of quantum computing. This is because many optimization problems can be mapped onto an Ising model, and quantum annealers, like those developed by D-Wave, are designed to find the minimum energy configuration of such models, thus solving the associated optimization problem.

In the context of "Ising graphs", the graph represents the lattice structure of the Ising model. Nodes represent lattice sites, and edges indicate interactions between spins.

License Information

All rights pertaining to SimGraphVisualizer are reserved by its creator, Dariusz Pojda. He can be reached at dpojda@iitis.pl for any queries or clarifications.

It's important to note that SimGraphVisualizer is licensed under the widely recognized Apache License 2.0, ensuring flexibility and adaptability for its users.

You can find full text of Apache License 2.0 at the end of this document.

Download and Installation Guide

Cloning the Repository:

Start by copying the SimGraphVisualizer repository to your local system using:

```
git clone https://github.com/euro-hpc-pl/simGraphVisualizer.git
```

Once cloned, navigate to the repository's directory:

```
cd simGraphVisualizer
```

Compiling the project

If you have changed something in source code or if you'd like to run simGraphVisualizer as a desktop application, you need to recompile it.

Prerequisites

Node.js library is required to compile simGraphVisualizer from source files. You can simply download installer from its website <https://nodejs.org> and install it on your system.

After you have node.js installed you have available npm command is in your path. Then go to the main directory of the simGraphVisualizer project and enter the command:

```
npm install
```

This command will use the npm tool to download all the additional libraries needed to compile the project.

Compiling

Now you can rebuild the library with the command:

```
npm run build
```

Using SimGraphVisualizer in a Web Browser:

For those who prefer a straightforward web browser experience, there's no additional setup needed. The 'public_html' folder within the repository is your gateway to this. Files within this directory are precompiled and primed for immediate use.

To initiate, just move or copy the entire contents of the 'public_html' directory to your desired location. The included 'index.html' serves as a practical demonstration and starting point for its utilization.

Making simple html project

When you want to embed simGraphVisualizer in your website you need to remember just a few important rules.

First, you need to import the BABYLON.js library in the header of your page by adding the following two lines:

```
<script src="https://preview.babylonjs.com/babylon.max.js"></script>
<script src="https://preview.babylonjs.com/gui/babylon.gui.js"></script>
```

Then, also in the header, you should import the simGraphVisualizer application:

```
<script src="js/sgv.js"></script>
```

We assume that the sgv.js file is in the 'js' subdirectory, as in the default configuration. If is not, you need to enter the correct path to this file.

Now you still need to include the stylesheets for simGraphVisualizer:

```
<link href="sgv.css" rel="stylesheet">
```

As before, remember to enter the correct path to the sgv.css file.

Finally, complete the <body> tag with a call to the procedure that runs simGraphVisualizer when the browser loads the page:

```
<body onload='sgv.display()'></body>
```

The simplest .html file allowing to use the simGraphVisualizer tool may currently look like this:

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title>Test wizualizacji grafów</title>
    <script src="https://preview.babylonjs.com/babylon.max.js"></script>
    <script src="https://preview.babylonjs.com/gui/babylon.gui.js"></script>
    <script src="js/sgv.js"></script>
    <link href="sgv.css" rel="stylesheet">
  </head>
  <body onload='sgv.display()'></body>
</html>
```

Customization of library view

You can change the look of the application's controls to match the look of your website. To do so, edit the style sheets located in the public_html/css subdirectory.

Optionally, if you use the 'less' tool, you can edit the *.scss source files in the public_html/scss subdirectory and compile them into proper .css sheets

Setting Up for Desktop Application:

To begin with, we assume that you have already installed the node.js library and followed all the steps described in the section 'Compiling the project'.

To begin with, we assume that you have already installed the node.js library and followed all the steps described in the 'Compiling the project' section. If you haven't done so already, then download node.js from its website and install it on your system. Then, in the head directory of your simGraphVisualizer project, execute the following commands:

```
npm install  
npm run build
```

What these commands do we described in the section 'Compiling the project'.

Running the application

Now you can already charm the simGraphVisualizer application in desktop mode. To do this, type the command:

```
npm run start
```

Now wait a short while and you should see the program window.

Creating executables and installer

If you don't want to use the npm tool to run the program every time, you can generate an executable version of application. You can do this by using the following command in the main directory of the simGraphVisualizer project:

```
npm run package
```

You can also build an executable application installer that you pass on to another person or put on a website. The installer will be built using the command:

```
npm run make
```

Coworking with other applications

The desktop variant of SimGraphVisualizer works well with external applications.

This functionality is dedicated to working with applications that perform calculations on the values of nodes and edges of a graph and return the state of these nodes and edges after the calculation. The state of the graph after external calculations is then displayed in SimGraphVisualizer.

Data transfer is implemented via text files, which are saved to a temporary directory defined by the user. Defining the path to this directory, as well as adding external applications to the application interface and defining their parameters, is facilitated by a dedicated settings window.

Settings window

The settings window is currently only available in desktop mode and allows you to define parameters for external applications that work with the simGraphVisualizer tool. User-defined settings are stored in a text file in JSON format and are loaded each time simGraphVisualizer is started.

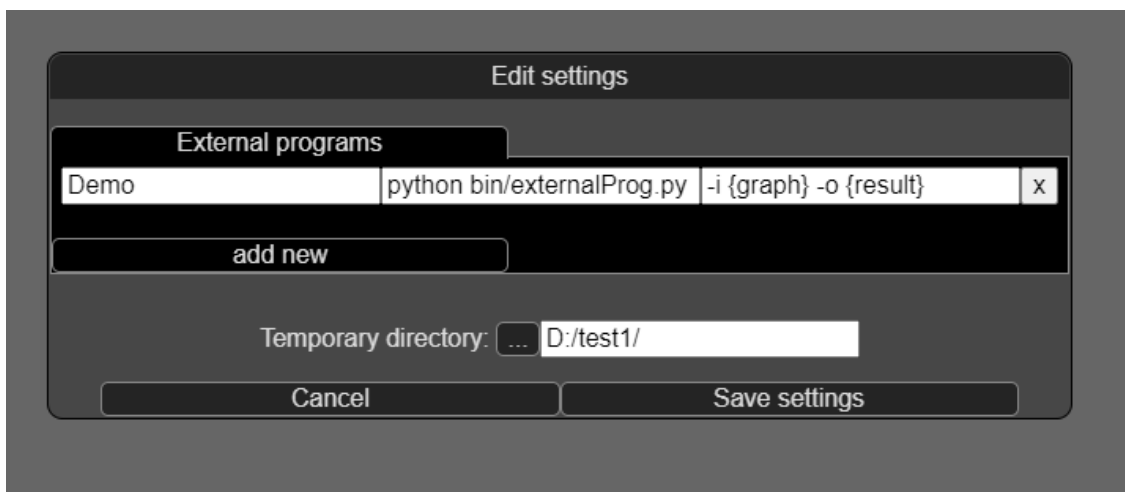


Image: Program configuration window.

It is possible to start an external application by selecting its name from the 'Run' menu. The input parameter is the currently processed graph and the result of the external application will be added to the workspace as 'result' namespace.

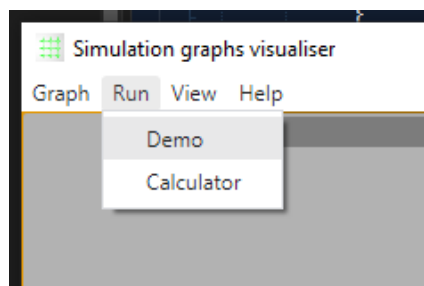


Image: 'Run' menu.

Working with graphs

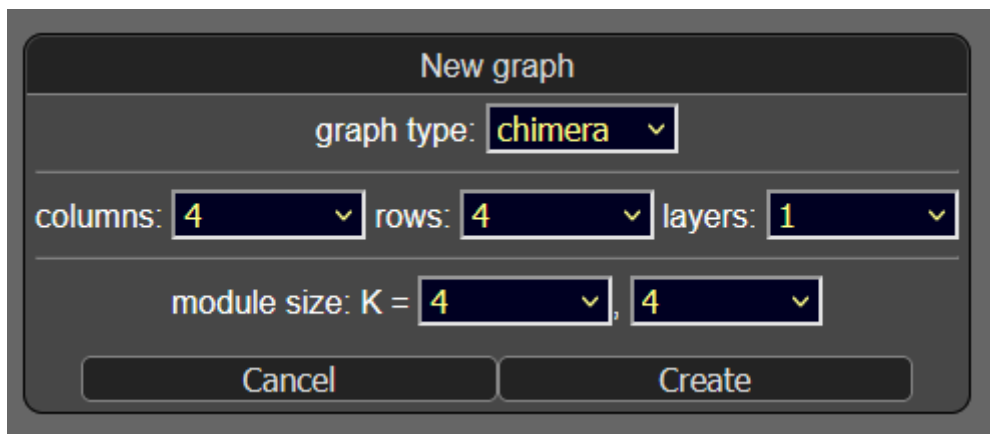
The primary function of simGraphVisualizer is to display Ising graphs and allow the user to view and modify them.

Creating a graph

To create a new Ising graph, you need to:

- in "browser" mode, click the "create graph" button on the control panel.
- in "desktop" mode, select "create graph" in the menu or click the "Create" button on the control panel.
- in both modes, by typing the "create" command in the console

In the last case, if the syntax of the command is correct, a chart will be created with the given parameters. If option 1 or 2 is used, a window will be displayed to select the parameters of the graph to be created.



The image shows a "New graph" dialog box. It has a title bar "New graph". Below the title bar, there is a label "graph type:" followed by a dropdown menu showing "chimera". Below this, there are three labels: "columns:", "rows:", and "layers:", each followed by a dropdown menu. The "columns:" dropdown shows "4", the "rows:" dropdown shows "4", and the "layers:" dropdown shows "1". Below these, there is a label "module size: K =" followed by two dropdown menus, both showing "4". At the bottom of the dialog, there are two buttons: "Cancel" and "Create".

In this window you can choose the type of graph (chimera or pegasus), the number of rows and columns, which define the number and arrangement of cells. For pegasus type graphs you can furthermore specify the number of layers.

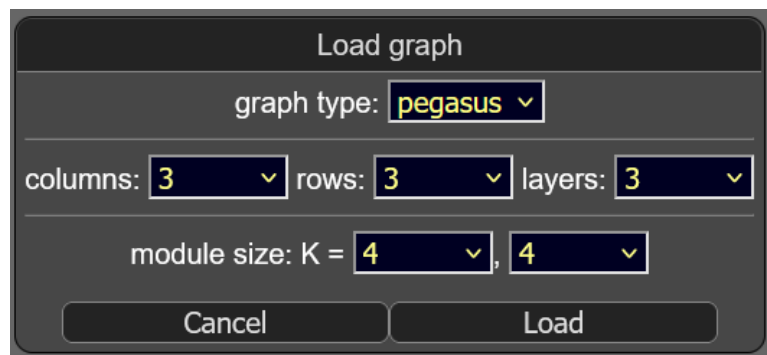
You can also specify the number of nodes in a cell. The default cell type is $K_{4,4}$, but you can create a graph containing cells with fewer nodes.

Loading and saving a graph

You can load the state of a graph from a text file in one of two supported formats: .txt and .gexf.

Since you can only work with one graph, loading a new graph will delete the existing work status. As in the case of graph creation, the corresponding buttons for loading and saving are available on the control panel and in the menu of the desktop application.

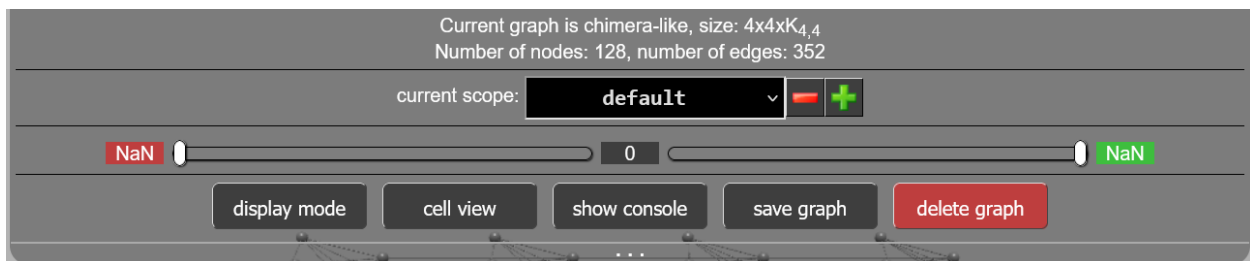
Files in .txt format are not always clear about the structure of the graph stored in them. SimGraphVisualizer will try to interpret this structure correctly, but just to be sure, the user will be shown an analogous window as when creating a new graph. There you can correct the program's interpretation.



A dialog box titled "Load graph" with a dark gray background. It contains several input fields: "graph type:" with a dropdown menu showing "pegasus"; "columns:" with a dropdown menu showing "3"; "rows:" with a dropdown menu showing "3"; "layers:" with a dropdown menu showing "3"; and "module size: K =" with two dropdown menus, both showing "4". At the bottom, there are two buttons: "Cancel" and "Load".

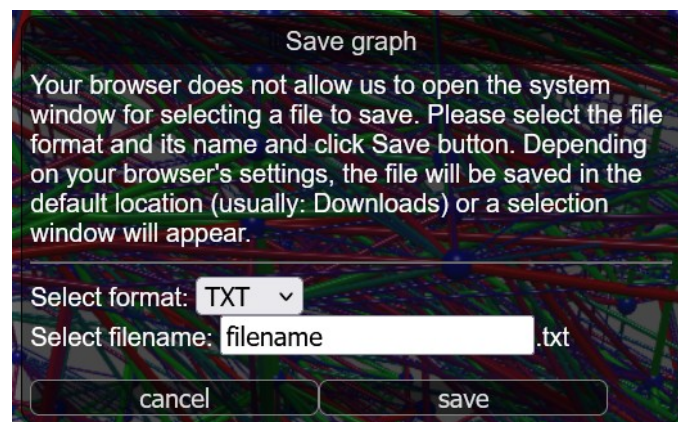
Saving the graph to a file may be difficult in browser mode due to restrictions imposed for security reasons on the javascript engine in web browsers. Nevertheless, every effort has been made to make it possible to save the current operational state in any application mode.

The option to save graph to a file is available in the control panel window.



A control panel window with a gray background. At the top, it displays "Current graph is chimera-like, size: 4x4xK_{4,4}" and "Number of nodes: 128, number of edges: 352". Below this, there is a "current scope:" dropdown menu showing "default" and a red minus button and a green plus button. A horizontal slider is shown with "NaN" on the left and "NaN" on the right, and a "0" in the middle. At the bottom, there are five buttons: "display mode", "cell view", "show console", "save graph", and "delete graph".

Usually a system window will open to select the directory and file to save, but in some browsers you will see a window:



A dialog box titled "Save graph" with a dark background. It contains a text area with the following text: "Your browser does not allow us to open the system window for selecting a file to save. Please select the file format and its name and click Save button. Depending on your browser's settings, the file will be saved in the default location (usually: Downloads) or a selection window will appear." Below the text area, there is a "Select format:" dropdown menu showing "TXT" and a "Select filename:" text input field with "filename" and ".txt" next to it. At the bottom, there are two buttons: "cancel" and "save".

TXT file format:

Text files are the simplest way to exchange information between applications. Each such file consists of consecutive lines corresponding to the nodes or edges of a graph.

The line corresponding to a node, for example, will have the format:

```
1 1 0.234
```

The doubled 1 is the number of the node, while 0.234 is the value assigned to that node.

The line corresponding to an edge will have a very similar format:

```
1 2 3.456
```

The numbers 1 and 2 are the numbers of the nodes that the edge connects, while 3.456 is the weight assigned to that edge.

It is assumed that all nodes are defined first, and then all edges. Of course, all node numbers appearing in the edge list should be declared beforehand.

In addition, the possibility of saving node labels in a .txt file has been introduced. The label is by default the fourth (optional) value in the line defining the node. A label saved in this way should be a string of characters not including spaces. It is not possible to define labels for edges.

GEXF file format:

The Graph Exchange XML Format (GEXF) is an XML standard-based language for describing network structures. The format has been developed since 2007 as part of the Gephi project (www.gephi.org). It allows saving various attribute values for each node and each edge of the graph, enabling memorization of all data series defined by the user or obtained as calculation results in a single file.

Depending on individual browser settings, the user may be shown a save location selection window with the option to change the file name. In other cases, the file will be placed in the default directory for saving files downloaded by the browser. An example .gexf file structure looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<gexf xmlns="http://gexf.net/1.2" version="1.2">
  <meta>
    <creator>IITiS.pl</creator>
    <description>SimGraphVisualizer GEXF export</description>
  </meta>
  <graph defaultedgetype="undirected">
    <attributes class="node">
      <attribute id="0" title="default" type="float"/>
      <attribute id="1" title="losowe" type="float"/>
      <attribute id="2" title="losowe2" type="float"/>
    </attributes>
    <attributes class="edge">
      <attribute id="0" title="default" type="float"/>
    </attributes>
    <nodes>
      <node id="1" label="any label">
        <attvalues>
          <attvalue for="0" value="0"/>
          <attvalue for="1" value="-0.8656414183390766"/>
        </attvalues>
      </node>
    </nodes>
  </graph>
</gexf>
```

```

</node>
<node id="2">
  <attvalues>
    <attvalue for="0" value="0"/>
    <attvalue for="1" value="-0.5303254916023477"/>
    <attvalue for="2" value="0.7"/>
  </attvalues>
</node>
...
</nodes>
<edges>
  <edge id="1" source="1" target="5">
    <attvalues>
      <attvalue for="0" value="0.5"/>
    </attvalues>
  </edge>
  <edge id="2" source="1" target="6">
    <attvalues>
      <attvalue for="0" value="-0.3333"/>
    </attvalues>
  </edge>
  ...
</edges>
</graph>
</gexf>

```

The GEXF format standard does not provide for saving additional attributes for the graph. However, in our case, it is necessary to keep in the file information about the type of graph (chimera, pegasus) and its dimensions. As it seems useful for these files to be correctly interpreted by other programs (although not necessarily with these atypical information), attempts were made to avoid adding non-standard tags and attributes. Therefore, additional information is embedded in the name of the default attribute for the node during saving (part highlighted in red in the listing below), and it is read and interpreted from it during reading. The absence of this information interrupts the reading of the file.

```

<?xml version="1.0" encoding="UTF-8"?>
<gexf xmlns="http://gexf.net/1.2" version="1.2">
  <meta>
    <creator>IITiS.pl</creator>
    <description>SimGraphVisualizer GEXF export</description>
  </meta>
  <graph defaultedgetype="undirected">
    <attributes class="node">
      <attribute id="0" title="default;chimera;4,4,4,4" type="float"/>
      <attribute id="1" title="losowe" type="float"/>
      ...
    </attributes>
  </graph>
</gexf>

```

Working with default 3d view

The application defaults to displaying the graph in 3D. The vertices of the graph are represented as spheres connected by edges. The color of the vertex corresponds to its assigned value, just as the color and thickness of the edge correspond to its weight.

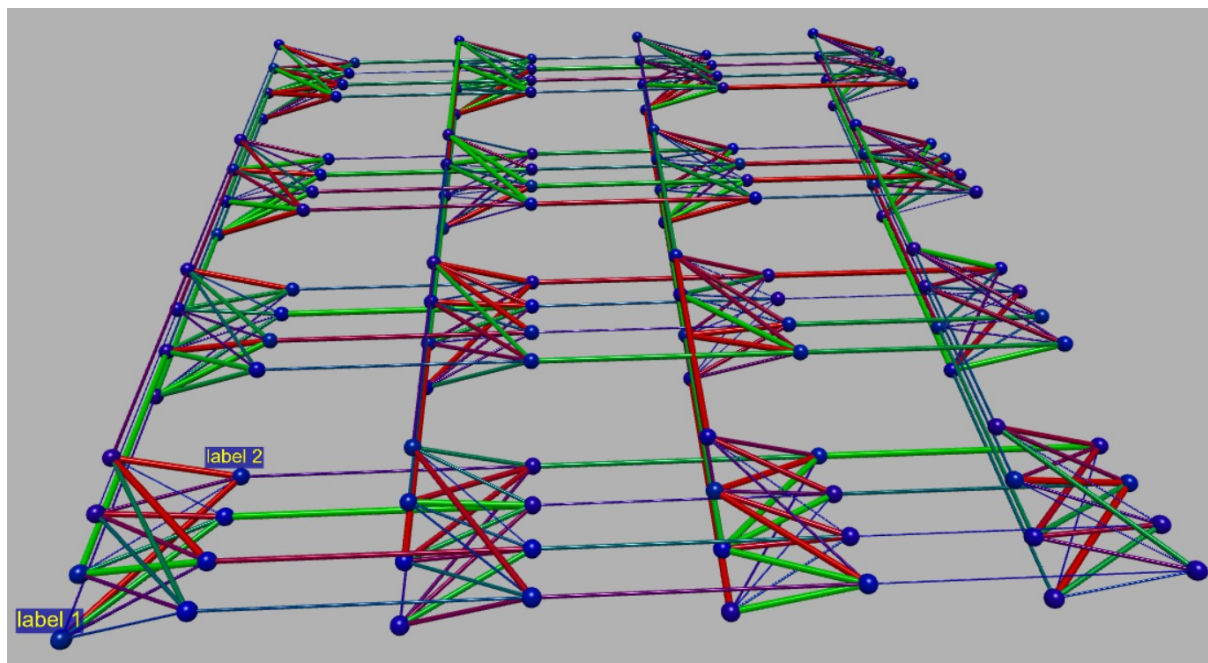


Image: Sample graph

Colors and thickness

Node and edge colours and edge thicknesses can visually represent the values assigned to them. Nodes and edges in grey indicate that they are not active (no value has been assigned to them) and dark blue indicates a zero value. Accordingly, grey and dark blue edges are drawn with a thin line.

Negative values are represented by colours going from dark blue to red and positive values by colours going from dark blue to green. Red corresponds by default to the minimum value occurring in the graph, and green to the maximum value. However, this can be changed using the sliders on the control panel, which allow the range of values between green and red to be selected. Edges which have been assigned values equal to (and exceeding) the set limits are drawn with the thickest line.

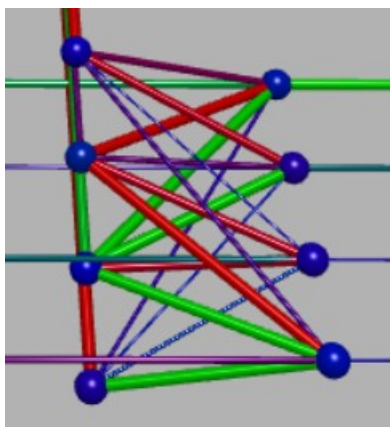


Image: Sliders for adjusting the colour range

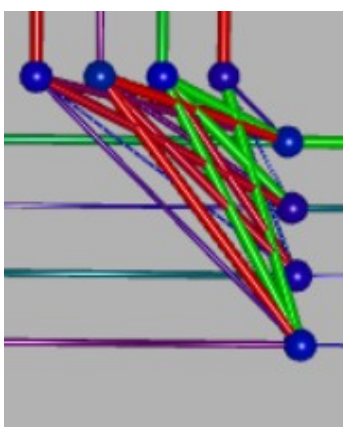
The ability to change the min-max range is intended to allow the user to get the clearest visualisation possible.

Display modes

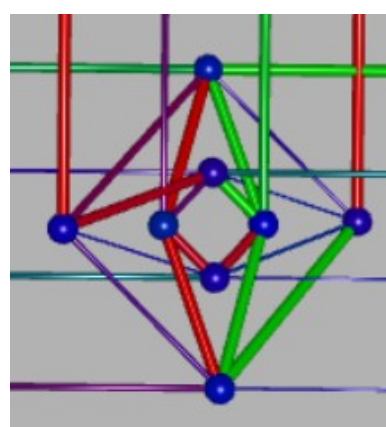
For your convenience, the 3D view has three modes for displaying graph cells: classic, triangle and diamond. They can be switched by a 'display mode' button on the control panel, an option in the desktop application menu and a console command.



classic view



triangle view



diamond view

Graph manipulation

In 3d operation mode, you can change the position of the graph by rotating, panning or zooming it. All these operations are performed with the mouse, with the left mouse button activating the rotation mode, the right mouse button activating the panning mode and the mouse wheel zooming the view of the graph.

Access to graph elements

You can use the left mouse button and click on a node or edge of the graph. You will then see a window in which you can edit the properties of this element or delete it (see sections: "Node editing" and "Edge editing").

Node labels

Nodes can have labels defined which are displayed in the 3D view mode and in the node properties edit window. By default, the labels are displayed as unobtrusively as possible, on a grey background. It is possible to optionally force some labels to be coloured according to a rule:

- when the value of the node is negative: red background
- when the value of a node is positive: green background
- when the value is zero: blue background
- when the value of a node is undefined: grey background

To force the colouring of a selected label, simply place the special character '&' at the beginning of the label text. This can be done either in the definition in the file or in the edit window of the node properties.

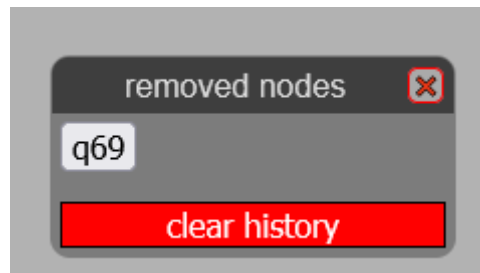
A large number of labels is not recommended due to performance limitations on some devices.

Restore deleted items

Both nodes and edges of a graph can be deleted from their properties window, by clicking the 'delete' button.

Edges are deleted permanently. Restoring a deleted edge is only possible by creating a new edge and assigning appropriate values to it.

Deleting a node is a more complex operation, as all edges originating from this node are deleted at the same time. It would be difficult to restore all these elements manually, so deleted nodes (including edges) are saved and can be restored at any time.

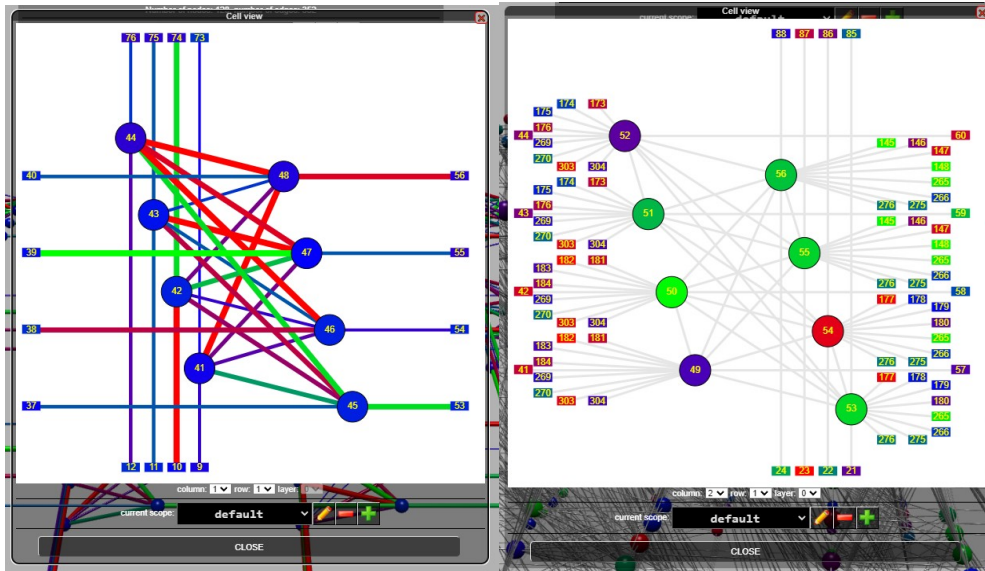


The symbols of the deleted nodes are displayed in the "removed nodes" window. Clicking on the symbol will restore the node and all those edges that were previously removed as long as it is possible to restore them (there is a second node). If a second node has been deleted in the meantime, the missing edge will only be restored after the second node has been restored.

All previously deleted nodes (and edges) can be forgotten irretrievably if you use the "clear history" button.

Using cell view window

Handling more complex graphs in 3D view mode is cumbersome due to the large number of elements displayed simultaneously, so we've enabled previewing and editing a single K4,4 cell of the graph in a separate dialog window.



Example dialog window previewing the K4,4 cell for a sample chimera-type graph (on the left) and pegasus-type graph (on the right).

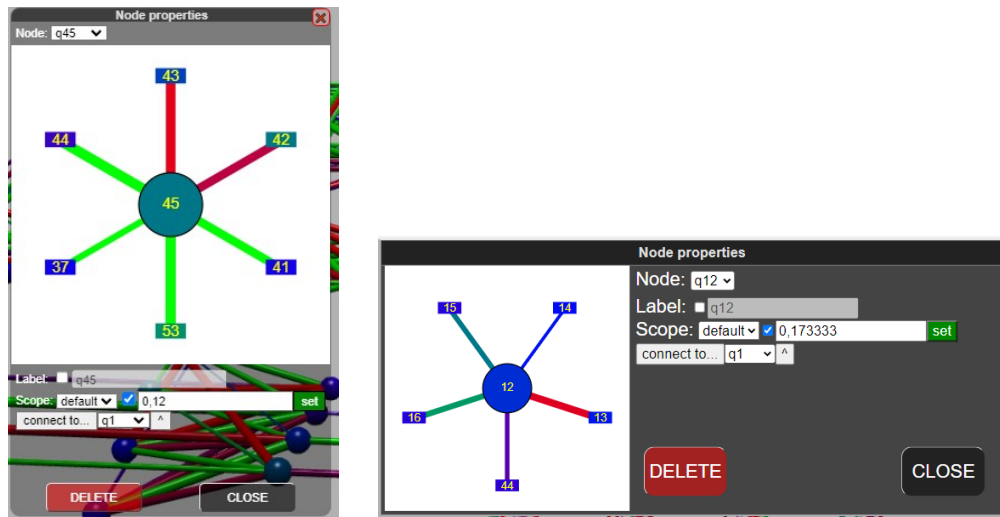
Nodes and edges are presented in this case in two dimensions. The colors (and in the case of edges also their thickness), similar to the 3D view, correspond to the weights assigned to these elements. All graphic elements are displayed using SVG technology, making them interactive. It's possible to display dialog boxes for editing node or edge properties. You can also navigate to another cell.

Operation of the dialog box:

- Select the column, row and layer you want to view using the corresponding drop-down lists. A visualization of the graph cell will be displayed, in which you can browse the contents.
- You can navigate through the cells in several ways:
 - Using the arrow buttons: ↑ (up), ↓ (down), ← (left), → (right).
 - By sliding your finger across the screen in the appropriate direction on touch devices.
- To change the displayed cell, do one of the following:
 - Click the corresponding arrow button.
 - Slide your finger across the screen in the appropriate direction on touch devices.
 - Select new values from the "column", "row" and "layer" drop-down lists.
 - You can close the "Cell view" dialog box by clicking the "Close" button.

Node editing

The node properties window allows editing the values assigned to this node. A graphical visualization of the node is also available, including an overview of edges converging on this node and neighboring nodes. By clicking on graphical elements you can edit edge properties or switch to editing properties of another node.



The dialog box for editing a sample nodes in desktop view (left) and mobile in landscape orientation (right).

You can change the currently edited node by selecting the node of interest from the "node" drop-down list too.

The window displays the properties of the selected node, such as:

- the possibility to enable or disable the display of the node's label using the "Label" checkbox.
- editing the node's label using the "Label" text box.
- activating or deactivating the node via the "Value" checkbox.
- setting the value of a node by entering a number in the "Value" field and clicking the "Set" button.
- being able to create a connection between a node and another node (a new edge) by clicking the "Connect to..." button and selecting the destination node from the drop-down list. Alternatively, it is possible to click the "^" button and then selecting the target node by clicking on it in the graph visualisation.
- you can also select the scope for which the node value is determined using the "scope" drop-down list.
- to delete the selected node, click the "DELETE" button. After deleting the node, the 'Node Properties' dialog box will automatically close.
- To close the 'Node Properties' dialog without making any changes, click the 'CLOSE' button.

Edge editing

The edge editing window allows you to change weights and other edge parameters.

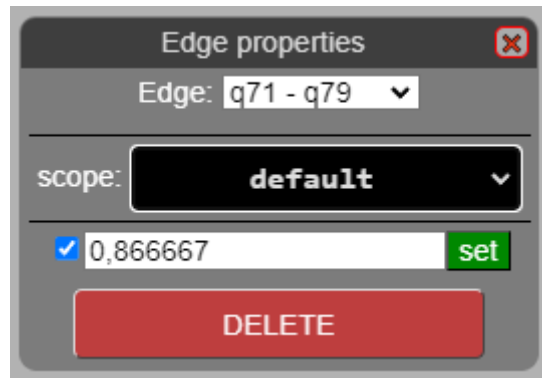


Image: Edge properties window

The 'Edge properties' dialog opens by clicking on the selected edge in the graph. Sometimes it is difficult to select an edge when it is drawn with a thin line or when many edges are visible in close proximity. In this case, you can:

- zoom in and rotate the graph so that the edge of interest is easier to hit,
- click on any edge and then select the edge of interest from the drop-down list in the properties window.

Once the right edge is selected, you have the option:

- activate or deactivate the edge using the "Value" checkbox.
- edit the value assigned to the edge by entering a number in the "value" field and clicking the "set" button.
- delete an edge using the "DELETE" button.

You can also select the scope for which the edge value is determined using the "scope" drop-down list.

To close the "Edge properties" dialog without making any changes, click the "CLOSE" button.

Working with console

Those who prefer to work in text mode can use the console. The console can be started by pressing the "console" button on the control panel and from the menu of the desktop application.

A number of commands are available to facilitate graph creation and editing:

create

Create new default graph:

syntax:

```
create chimera [4|8|12|16],[4|8|12|16],[1..4],[1..4]
create chimera [4|8|12|16],[4|8|12|16],1,[1..4],[1..4]
create pegasus [4|8|12|16],[4|8|12|16],[1..9],[1..4],[1..4]
```

The 'create' command has two mandatory parameters.

The first is the type of graph. This can be one of two types: chimera or pegasus.

The second is a sequence of numbers, separated by commas, specifying the size of the graph to be created.

The size of a chimera graph may be 4 or 5 numbers, while a pegasus graph must be 5 numbers.

If 4 numbers are given (chimera only), then:

The first and second numbers are the number of columns and rows, respectively, on one layer of the graph. The product of the number of columns and the number of rows determines how many cells the graph will contain. These can take one of four values: 4, 8, 12 or 16.

The third and fourth numbers can take the values 1, 2, 3 or 4 and define the number of nodes in a cell on the left (KL) and right (KR) respectively.

If 5 numbers are given (chimera or pegasus):

The first and second numbers are the number of columns and rows, respectively, on one layer of the graph. The product of the number of columns and the number of rows determines how many cells the graph will contain.

These can take one of four values: 4, 8, 12 or 16.

The third number determines the number of layers in the graph. For a chimera graph, it should be 1, other values will be ignored.

The fourth and fifth numbers can take the values 1, 2, 3 or 4 and define the number of nodes in a cell on the left (KL) and right (KR) respectively.

set

Set or delete the value of a node or edge in the current scope.

Assigning a node value

```
set <nodeId>=<value>
```

Assigning a edge value:

```
set <nodeId>+<nodeId>=<value>
```

Deleting a node value:

```
set <nodeId>
```

Deleting a edge value:

```
set <nodeId>+<nodeId>
```

scope

Allows the management of value scopes in the graph. It has the following functions:

Displays a list of existing scopes in the graph:

```
scope list
```

Sets the current scope for the graph to <scope_name> (if it exists):

```
scope set <scope_name>
```

Adds a new scope <scope_name> for the graph (only if a scope with that name does not already exist) and sets it as the current one:

```
scope add <scope_name>
```

Removes scope <scope_name> from the graph (if a scope with that name exists):

```
scope delete <scope_name>
```

clear

Deletes the current graph.

```
clear
```

displaymode

Set style of cells display to classic, triangle or diamond.

```
displaymode classic  
displaymode triangle  
displaymode diamond
```

limits

Allows management of the value ranges used for colouring nodes and edges.

Displays some informations about limits:

```
limits
```

Set a minimum and maximum range. <min> is the value below which the node or edge will always be coloured red, and <max> is the value above which the node or edge will always be coloured green.

```
limits set <min> <max>
```

help

Displays a help screen.

```
help
```

Apache License

Version 2.0, January 2004

<http://www.apache.org/licenses/>

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"**License**" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"**Licensors**" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"**Legal Entity**" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "**control**" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"**You**" (or "**Your**") shall mean an individual or Legal Entity exercising permissions granted by this License.

"**Source**" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"**Object**" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"**Work**" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"**Derivative Works**" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"**Contribution**" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "**submitted**" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "**Not a Contribution**."

"**Contributor**" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to

reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

- a. You must give any other recipients of the Work or Derivative Works a copy of this License; and
- b. You must cause any modified files to carry prominent notices stating that You changed the files; and
- c. You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
- d. If the Work includes a "**NOTICE**" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or

redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [2023] [dpojda@iitis.pl]

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at <http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and limitations under the License.