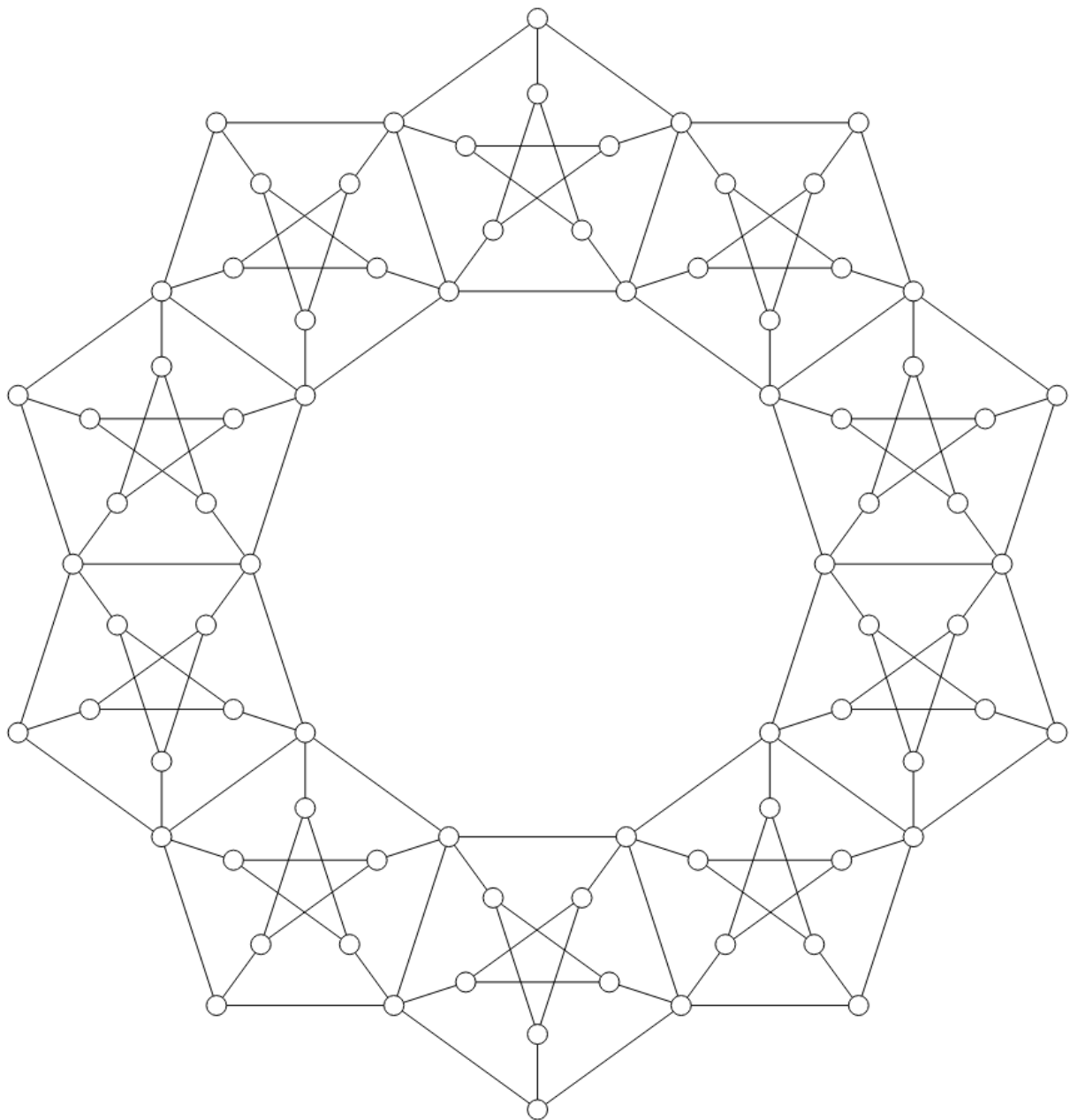


Graphic Guide

May 3, 2026



Preface

Graphic is a program whose purpose is to help people create (hopefully beautiful) graph drawings from ideas or sketches they already have in mind. It is not designed to create a drawing from the node/edge information contained in some abstract description of a graph (such as a list of edges).

Graphic has been designed to be as self-explanatory as possible, and most (if not all) features can be discovered via experimentation with the program. However, this document is available for those who would rather have a bit of guidance.

Don't worry too much about the length of this guide — it has lots of pictures.

Contributors

Graphic's design grew out of the ideas generated and investigated during Nick Wetmore's design and implementation of his *Grapha* graph drawing program, which was part of the work for his honours thesis. Graphic was initially designed and implemented by Rachel Bood for her honours thesis; her design expanded on the ideas and principles discovered by Nick. Ian Cathcart added considerable capability to Rachel's final version during a co-op work term, taking her program from a solid proof of concept to a more polished tool. Jim Diamond supervised these projects, and also added some features and fixes to the software.

This document was written by Jim Diamond.

Cover Image

The graph drawing shown on the title page took about two minutes to create using Graphic.

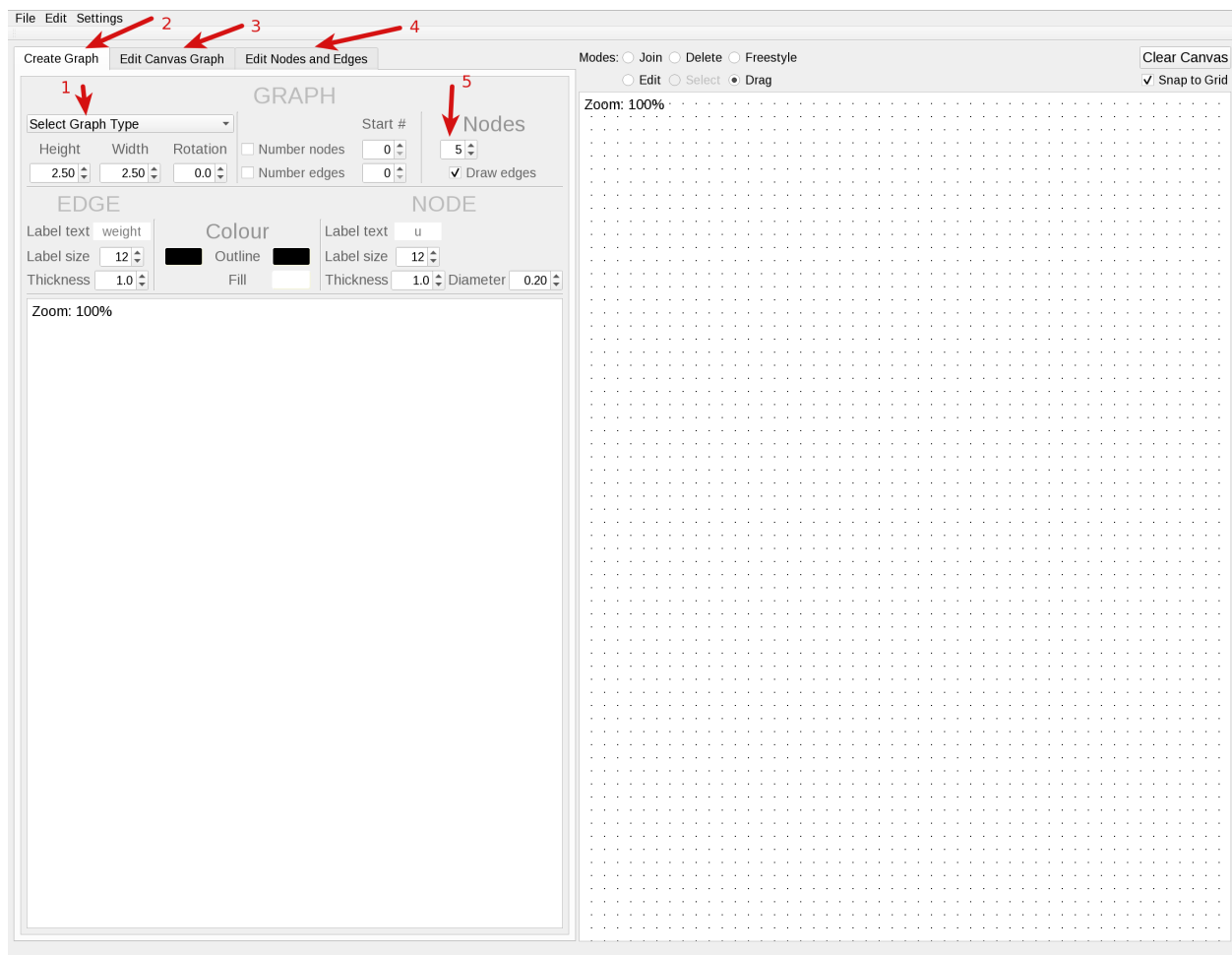
Contents

Section 1: First Steps: the “Create Graph” Tab	1
Section 2: Styling Graphs in the Preview Area	6
Section 3: Other Graph Drawing Operations	9
3.1: Operations Available on the Canvas	9
3.2: The “Edit Canvas Graph” Tab	15
3.3: The “Edit Nodes and Edges” Tab	18
Section 4: All About Node and Edge Labels	21
4.1: \TeX math mode syntax for people who don’t know \TeX	21
4.2: Differences between \TeX math mode syntax and Graphic label syntax	22
4.3: Visual differences between Graphic’s labels and \TeX math output	22
4.4: Details of Graphic’s TikZ output for \TeX , \LaTeX and \ConTeXt Users	23
Section 5: Graphic’s Settings and Miscellaneous Controls	26
5.1: Graphic’s Settings	26
5.2: Miscellaneous Controls: Zooming	29
Section 6: Continuing Development	29

1. First Steps: the “Create Graph” Tab

This user guide assumes you have downloaded and installed Graphic as per the instructions found at <https://graph-drawing.acadiau.ca>, and that you are able to execute Graphic on your computer. Graphic is somewhat hungry for screen real-estate, so if you are using a laptop with a small screen you may find it useful to use a large external monitor, if at all possible.

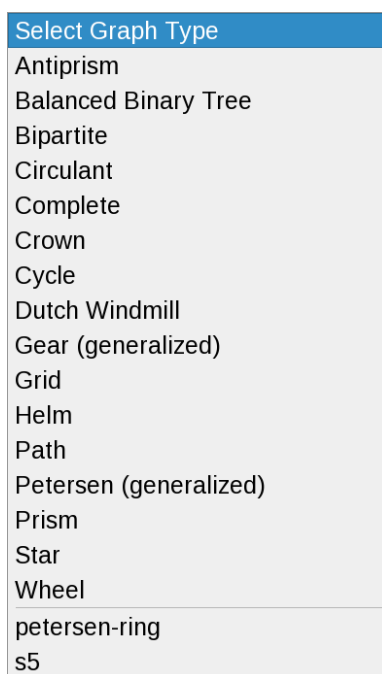
When you start Graphic, you will see the screen shown in Figure 1.1 (except for the red arrows and red numbers, which have been added to the screen shot for the purposes of this guide). Although you may see slight differences in the look of the widgets and menus, depending on which operating system you use, the general layout and features should be otherwise identical.



Initial screen of Graphic
Figure 1.1

The user interface is divided into two parts. The left side can have three different functions, depending on which of the three tabs (red numbers 2, 3 and 4) is selected. When the **Create Graph** tab (red arrow #2) is selected, there is a *preview* area on the bottom of the left side (as shown in the figure). In most cases, the first action a user will take is to select one of the built-in graph types (*i.e.*, a graph which Graphic knows how to draw all by itself), or one of his/her own graphs, by using the drop-down menu (red arrow #1).

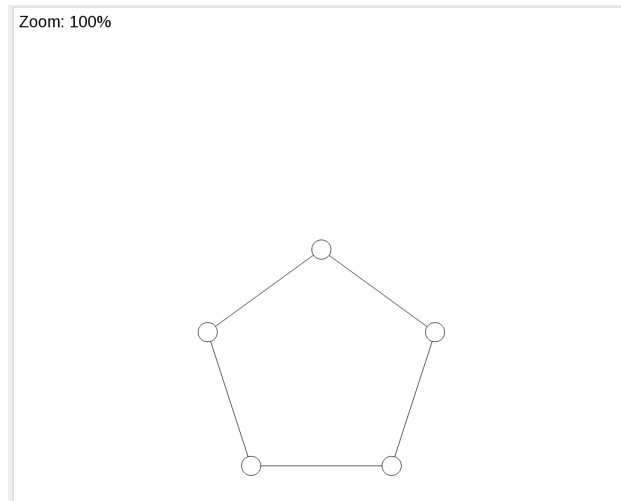
Figure 1.2 shows an example of the drop-down menu accessed via the **Select Graph Type** widget. The graphs above the line (*i.e.*, from **Antiprism** to **Wheel**) are built-in graph types. The graphs below the line (in this example, **petersen-ring** and **s5**) are graphs which have been previously constructed using Graphic, and thus are now part of the user's personal "library".



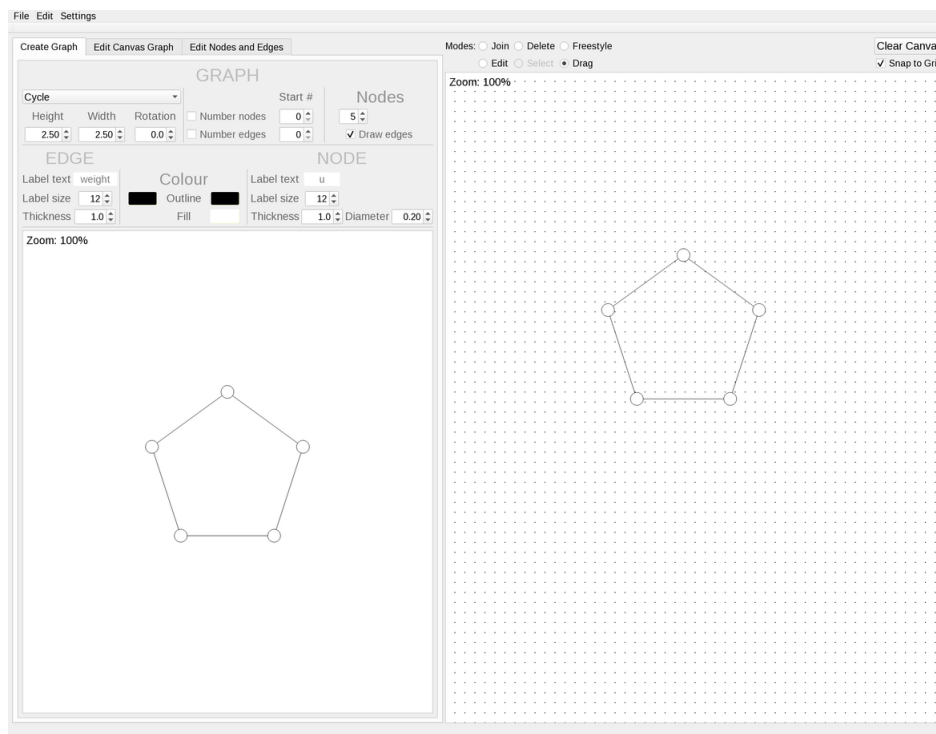
A menu of available graphs
Figure 1.2

As a simple example, if you select **Cycle**, a 5-cycle is drawn in the preview area, as shown in Figure 1.3 (note that the figure shows only the top portion of the preview area).

There are many variations of a cycle which can be generated; how to draw these variations will be discussed in Section 2. But for now, suppose we just want a drawing of a C_5 , and what is shown in the preview area is what we want. In this case, drag the drawing from the preview area to the main canvas on the right side using the left mouse button. Graphic automatically redraws the graph in the preview, and the screen will now look like Figure 1.4.



A 5-cycle in the preview area
Figure 1.3



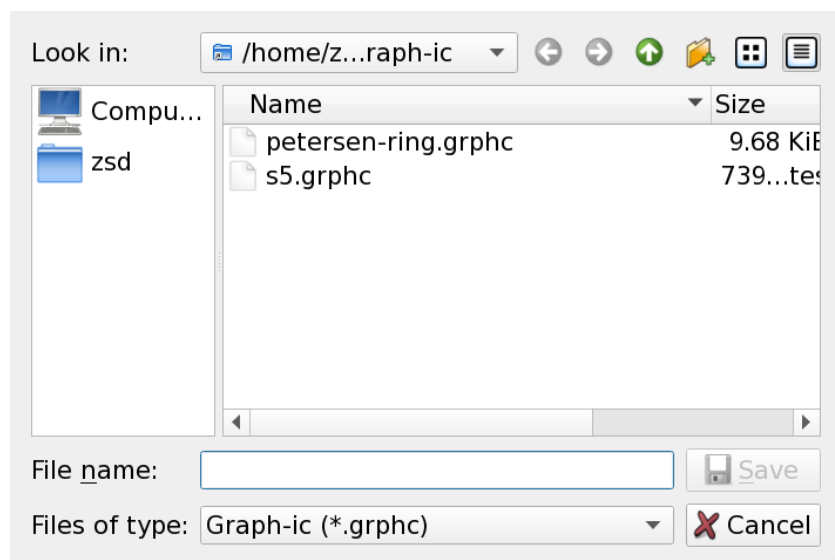
A drawing of a C_5 dragged to the canvas
Figure 1.4

Using the **Save** item in the **File** menu, the graph drawing on the canvas can be saved in a number of formats. By default, the choice will be to save the graph as a **.grphc** file; this is not an image file to be used in a document; rather, this is Graphic’s “library graph” format, which you can use to read the graph back in later, should you wish to

do so. Depending on the particular operating system in use, there are various graphical file formats available, such as PNG, WEBP, TIF, SVG and JPG (although you normally shouldn't use JPG for drawings such as those created by Graphic). The graph drawing can also be saved as TikZ code (input for $\text{T}_{\text{E}}\text{X}$, $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ and $\text{ConT}_{\text{E}}\text{Xt}$). Finally, the graph drawing can be saved as a list of edges, should it be useful to have this information as input to some program which processes graphs[†].

Files saved by Graphic are found in a directory called **graph-ic**, which Graphic creates as a sub-directory of the directory from which Graphic is run. (If you start Graphic via a menu item (or similar) from your computer's graphical interface, you may have to hunt around a bit to find the **graph-ic** directory.)

Figure 1.5 shows[‡] what the **Save Graph** dialog box looks like, when it is summoned either by the **File/Save** menu item or by typing **Ctrl-S**. At the bottom of the window the default file type (**Graph-ic (grphc)** in the **Files of type:** drop-down list). All files currently in the **graph-ic** directory whose names ends in **.grphc** are shown in the dialog.

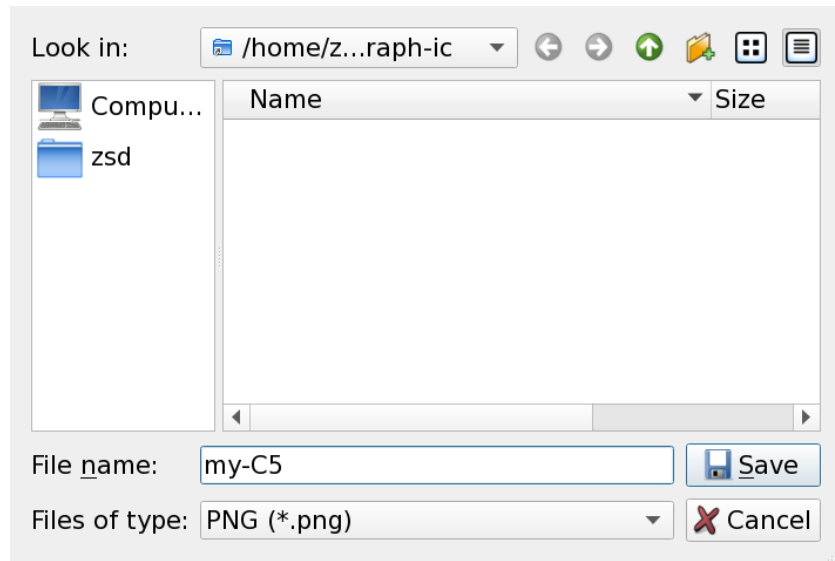


The **Save Graph** dialog box
Figure 1.5

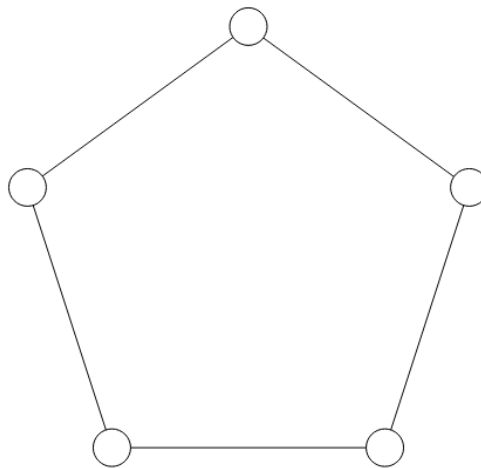
[†] Since different graph-processing programs require input in different formats, it is likely that the edge list produced by Graphic will need some massaging before the information will be acceptable to another program.

[‡] As mentioned previously, depending on the particular operating system you use, your dialog box may look somewhat different than the one pictured.

Should you wish to create a PNG image of the graph(s) on the canvas, choose PNG from the Files of type: drop-down list. Then choose a file name, such as `my-C5`, as shown in Figure 1.6. Finally, click the **Save** button, which saves the graph drawing in the file `graph-ic/my-C5.png`. This file is shown in Figure 1.7.



The Save Graph dialog box after entering name `my-C5` and selecting file type “PNG”
Figure 1.6



The PNG image file `my-C5.png` containing a drawing of a C_5
Figure 1.7

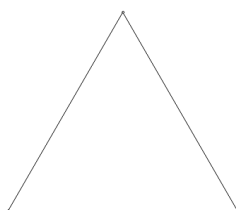
2. Styling Graphs in the Preview Area

In Section 1 an example of creating a simple graph drawing and saving it as a PNG file was given. This chapter describes how such graph drawings can be “styled”. It is probably best to be running Graphic as you read this section, so that you can “play along”.

Figure 2.8 shows the widgets with which graph drawings in the preview area can be styled. In Section 1 the function of the **Select Graph Type** drop-down list was mentioned, but the others will now be explained.

Graphic’s preview styling widgets
Figure 2.8

At the upper left, below the **Select Graph Type** drop-down list, there are “spinboxes” which allow the graph drawing to be scaled or rotated. Specifically, the height and width specify the size (in inches) of a rectangular box in which the drawing must fit (taking into account any rotation specified). Note that some graphs, when drawn as a regular polygon, are not as high as they are wide. For example, the 3-cycle shown in Figure 2.9 is $\sqrt{3}/2$ as high as it is wide.



A C_3 with very small node circles
Figure 2.9

This brings up a general principle of Graphic: given the choice of drawing a regular polygon or making the graph drawing the specified height and width, Graphic chooses to draw the largest regular polygon which is wholly contained in a height×width box. However, since the height and width controls are independent (for most graphs), the user can adjust the height or width to scale the graph more in one direction than the other. Thus, if a drawing of a 3-cycle that is as high as wide is desired, the height control can be adjusted to be (an approximation) of $2/\sqrt{3}$ the value of the width control. Alternatively, the features of the **Edit Canvas Graph** tab can be used, as described in Section 3.2.

Note that the spinboxes have arrows which will adjust the value by default amounts. However, the text in the spinbox can be edited to directly select any (legal) desired value (put the mouse cursor in the numeric text field and click, then edit the value as desired).

Also note that the graph height and width settings include the circles drawn for the nodes. Thus adjusting the node diameter (via the **Diameter** control seen at the lower right of Figure 2.8) causes the center of some or all nodes (depending on the graph) to change position, which may affect the position of edges. This effect can be observed by holding down one of the arrows on the **Diameter** spinbox. (At time of writing, the thickness of the node circle outline is not taken into account when sizing the graph. The thickness of the node outline can be adjusted with the **Thickness** spinbox next to the node diameter spinbox. This value is in pixels, not inches.)

The thickness of the edge lines can be adjusted with the **Thickness** spinbox seen at the lower left of Figure 2.8.

All of the built-in graph types have at least one spinbox for the number of nodes (just under the large **Nodes** word, seen toward the upper right of Figure 2.8). Some built-in graph types have two spinboxes to specify the number of nodes. For example, when creating a **Bipartite** graph, you are able to specify the number of nodes in each partition independently.

Nodes and edges can be sequentially labelled with numbers by clicking, respectively, the box to the left of the **Number nodes** or **Number edges** label (seen at the top middle of Figure 2.8)[†]. By default the number sequence starts at 0, but the spinboxes to the right of **Number nodes** and **Number edges** can be used to choose alternative starting values.

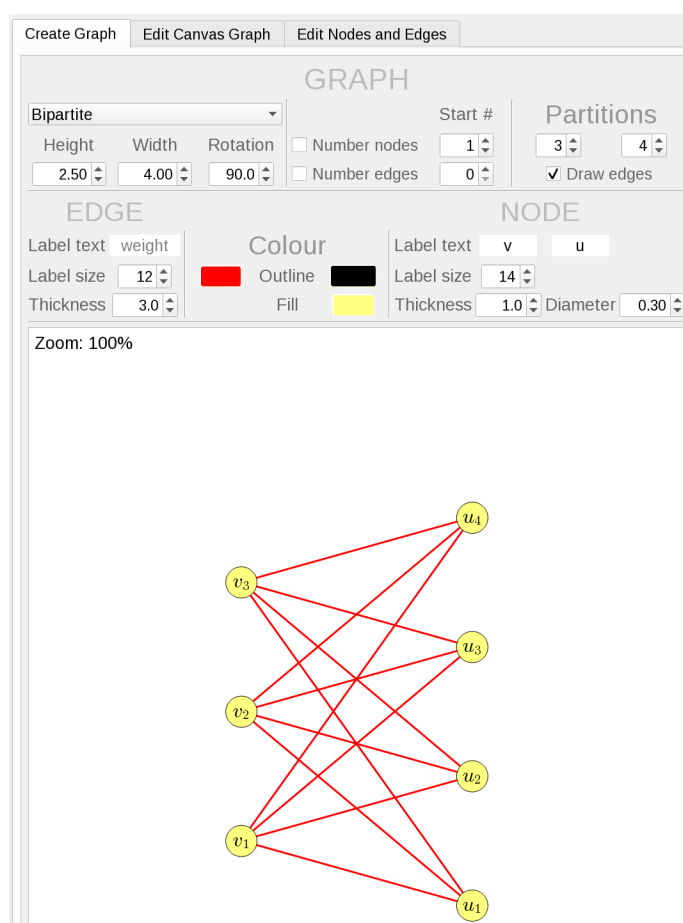
Nodes and edges can also be labelled with labels such as “ v_1 ”. This is done by ensuring that the desired check-box (*i.e.*, the one for edges or the one for nodes) is **not** checked, and then entering text (such as v) in the text entry box to the right of **Label text**. Note that there are two such text entry boxes, one on the left for edges and one on the right for nodes. (In fact, for bipartite graphs there are three such boxes, so that you can have (for example) u_i s for one partition and v_j s for the other partition.)

[†] The edge labels are drawn on top of the middle of the edges, which is probably not the best place from either an aesthetic or readability point of view. Changing this is one possible future enhancement for Graphic.

Finally (for the preview area), it is possible to change the colour for the edge lines, the colour for the node circle outlines, and the colour for the node fills. Like the line thickness and label size widgets, these affect all nodes and edges in the preview; but these features can be individually adjusted using the facilities provided by either the **Edit Canvas Graph** tab (Section 3.2) or the **Edit Nodes and Edges** tab (Section 3.3).

As an example, Figure 2.10 shows the **Create Graph** tab (shrunk to fit on this page), with the controls set up to draw the complete bipartite graph $K_{3,4}$ where

- the edges have been drawn in red, rather than the default black;
- the edges have been drawn with 3.0 pixel thickness, rather than the default 1.0 pixel;
- the nodes have been filled with a light yellow colour, rather than the default white;
- the nodes of one partition have been labelled v_1 through v_3 ;
- the nodes of the other partition have been labelled u_1 through u_4 ;
- the node diameters have been increased from the default 0.25" to 0.30";
- the font size of the node labels has been changed from the default 12 pt to 14 pt;
- the graph's width (before rotation) was changed from the default 2.50" to 4.00"; and
- the graph has been rotated 90° .



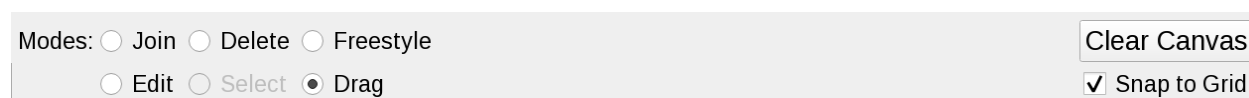
The Create Graph tab displaying a bipartite graph with various stylings
Figure 2.10

3. Other Graph Drawing Operations

This section describes the various other facilities Graphic provides to create graph drawings and/or modify drawings.

3.1. Operations Available on the Canvas

Figure 3.11 shows the controls located above the main canvas (shown here in a larger scale than in Figure 1.1). When a graph is dragged from the preview area to the main canvas (an example of which is shown in Figure 1.4), or when any graph parameters are changed in the **Create Graph** tab, the **Drag** mode is automatically selected (as shown in Figure 3.11). Drag mode allows any of the graph drawings currently shown on the main canvas to be moved about, as desired (move the mouse cursor to be over any graph drawing, hold down the left mouse button, move the mouse, and then release the mouse button).



The controls above the main canvas
Figure 3.11

The **Select** mode is (at time of writing) only of use when the **Edit Canvas Graph** tab is selected, and is otherwise greyed out, as seen in Figure 3.11. This mode will be explained in Section 3.2.

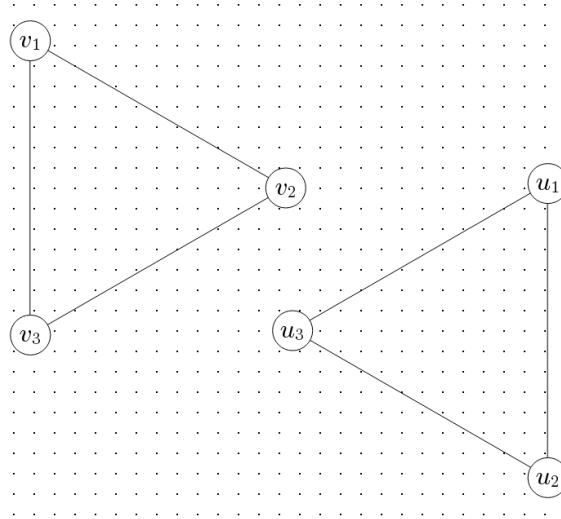
If the **Delete** mode is selected, when the mouse cursor is placed over an edge or a node and the left mouse button is clicked, the edge or node is deleted from the graph drawing. If the mouse cursor is over a graph drawing and the left mouse button is double-clicked, the entire drawing is deleted.

If the **Freestyle** mode is selected, when the left mouse is double-clicked, a new node is added to the canvas at the mouse's current location. (This node is styled according to the current settings in the **Create Graph** tab.) If the left mouse button is single-clicked when the cursor is over a node, subsequent single clicks over other nodes will join those pairs of nodes with edges, creating a path. The most recently selected node is drawn with a dashed outline to give the user visual feedback of where one end of the next edge (if any) will be drawn. To end the creation of a sequence of edges (perhaps to start from another node), clicking the mouse over empty space on the canvas will de-select any selected node.

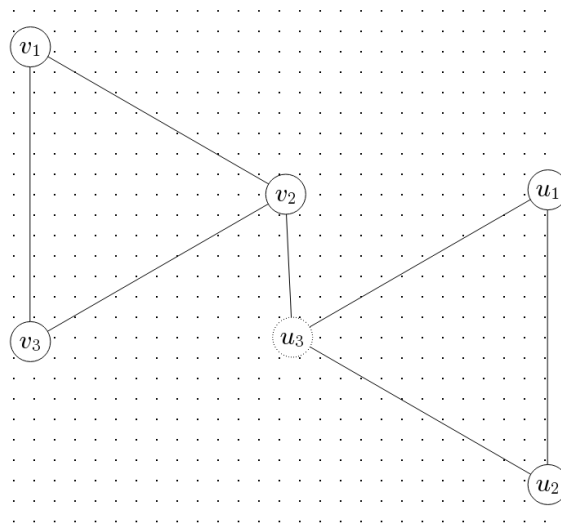
All nodes and edges that are created after **Freestyle** mode is selected, until some other mode is selected, are considered to be in the same graph drawing. Thus if, after using **Freestyle** mode to create (for example) 12 nodes and 3 edges, **Drag** mode is selected, then dragging any one of those nodes or edges causes all 15 items to move together.

Further, if a **Freestyle** edge is created between two separate graph drawings on the canvas, the two graph drawings are joined and become a single drawing. For example,

Figure 3.12 shows two C_3 graphs (on the canvas) which were dragged from the preview area to the canvas. Clicking on node v_2 and then node u_3 adds an edge, as shown in Figure 3.13. Notice that the node with label u_3 is drawn with a dashed outline, indicating that it is the currently selected node.



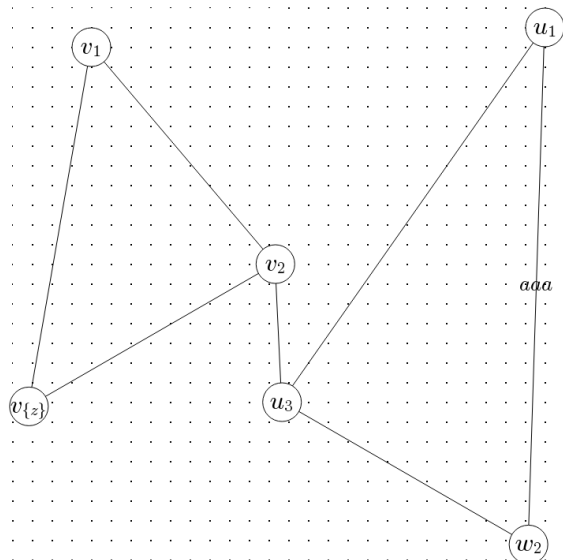
Two C_3 graph drawings
Figure 3.12



After joining the two C_3 graph drawings of Figure 3.12
Figure 3.13

After adding the edge between v_2 and u_3 , using Drag mode to move any of the shown nodes or edges will move all of the shown nodes and edges, as there is now only one graph drawing on the canvas.

When **Edit** mode is selected, two different types of modifications can be done to the graph drawing. First, individual nodes can be dragged around the canvas and dropped at any desired location[†]. Second, if the center of a node or edge is clicked, the label can be modified; the method for specifying the labels is somewhat complex for anyone not familiar with how \TeX typesets mathematics, and is described in detail in Section 4. As an example, the graph in Figure 3.14 was obtained from the graph in Figure 3.13 by dragging nodes v_1 and u_1 , changing the labels v_3 and u_2 to $v_{\{z\}}$ and w_2 (respectively), and putting the label aaa on the edge (u_1, w_2) .



The graph drawing of Figure 3.13 with some edits
Figure 3.14

The **Join** mode provides the facility to join two separate graph drawings. Unlike the **Freestyle** mode edge-addition technique, **Join** mode allows the user to *identify*[‡] one node from each of two graphs, or, alternatively, two nodes from each of two graphs.

Some examples of this graph joining feature are in order. Figure 3.15 shows two C_3 graph drawings on the canvas; suppose we wish to join these two graphs by identifying the nodes with labels “1” and “3”. We begin by selecting these nodes (by left-clicking each node in turn); Figure 3.16 shows the canvas after these nodes have been selected (note that selecting the nodes causes their outlines to be drawn with dashed lines, to give visual feedback to the user). Upon pressing the “j” key, the following happens:

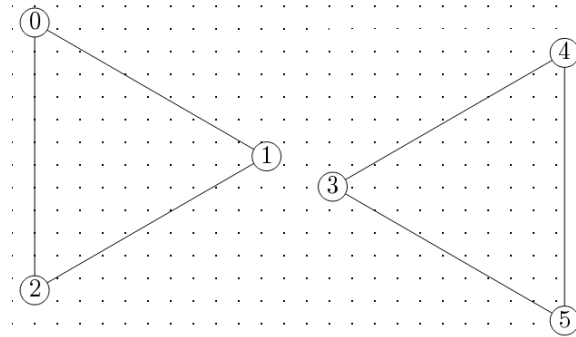
- the graph whose node was selected second is translated on the canvas so that the two selected nodes are at the same location;

[†] But see the discussion of the **Snap to Grid** check-box below.

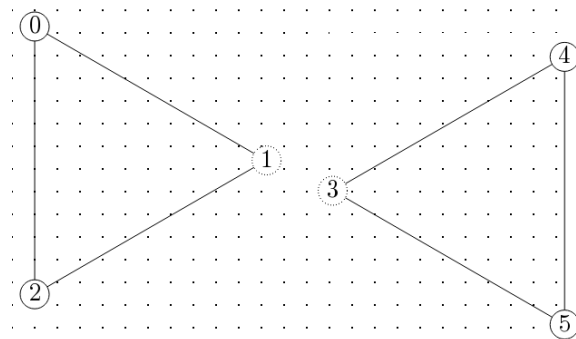
[‡] When two nodes are *identified* they are replaced with one new node, which is adjacent to all other nodes which were adjacent to either of the identified nodes.

- the second selected node is identified with the first selected node; and
- when the first node selected has a label consisting solely of a number, all of the nodes in the (combined) graph are re-numbered in sequence.

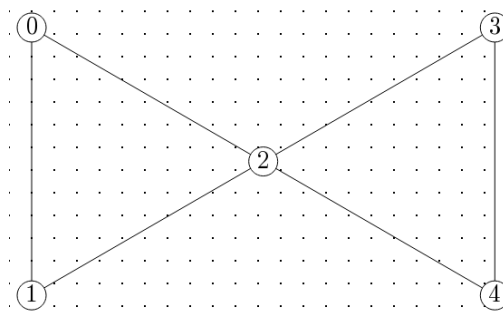
The resulting graph drawing is shown in Figure 3.17.



Two C_3 graph drawings on the canvas
Figure 3.15



Two C_3 graph drawings on the canvas with nodes 1 and 3 selected
Figure 3.16

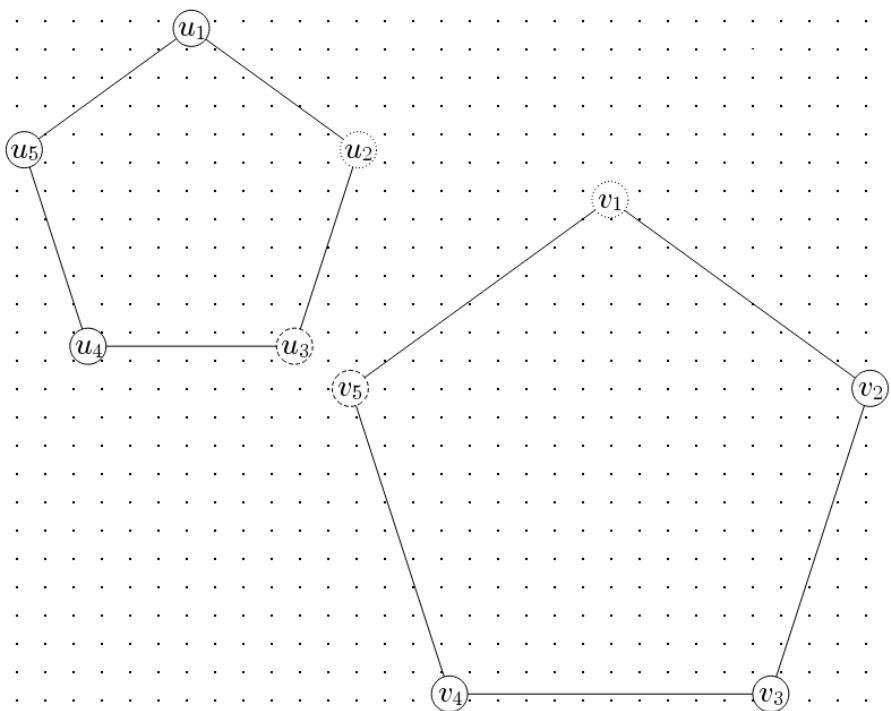


The result of joining the two graph drawings of Figure 3.16
Figure 3.17

Joining graphs with two pairs of nodes is somewhat more complex. Figure 3.18 shows two (unequal-sized) C_5 drawings, with four nodes selected. The order of selection is more important than when a join is performed with only two nodes for the following reasons:

- the graph drawing whose node is selected first does not change shape, whereas the other graph may (as will be seen in this example); and
- the first node selected in the first graph is identified with the first node selected in the second graph.

Graphic displays the first-selected node of each graph by drawing each such node with a dotted line, and the second-selected node of each graph with a dashed line. (Graphic does not show which node was selected first: the user must remember this in cases where it matters.) The order in which the second node of each graph is selected is irrelevant.



Two C_5 graph drawings, with four nodes selected

Figure 3.18

Denote the graph drawing whose node was selected first as “the first graph”, and the other graph the “second graph”. When two pairs of nodes have been selected from two separate graph drawings, pressing the “j” key causes the following actions:

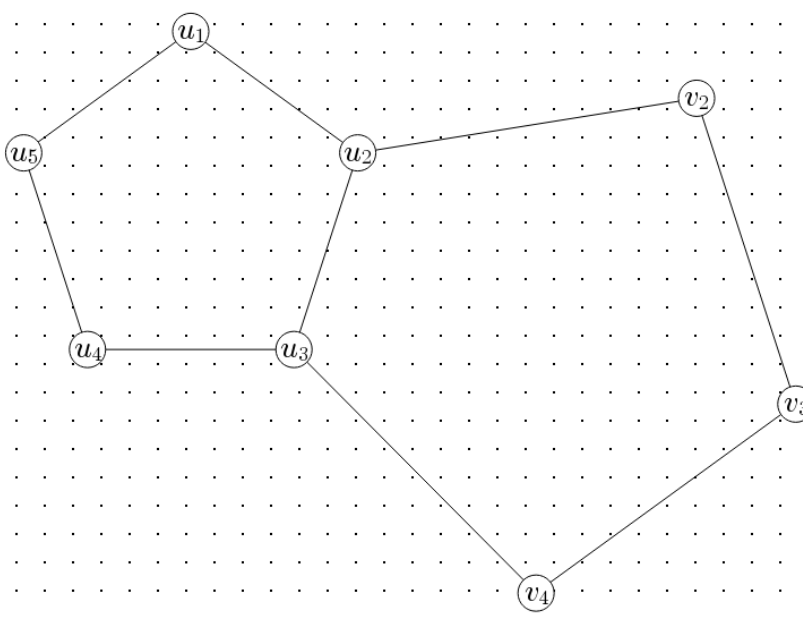
- the second graph is rotated so that the line segment through its two selected nodes is parallel to the line segment through the two selected nodes of the first graph;
- the second graph is translated so that the midpoint of the line segment connecting the second graph’s selected nodes is coincident with the midpoint of the line segment connecting the first graph’s selected nodes;

- the first graph’s first selected node is joined to the nodes adjacent to the second graph’s first selected node;
- the first graph’s second selected node is joined to the nodes adjacent to the second graph’s second selected node; and
- the second graph’s selected nodes are deleted.

(This action is animated to make it easier to see what happens.)

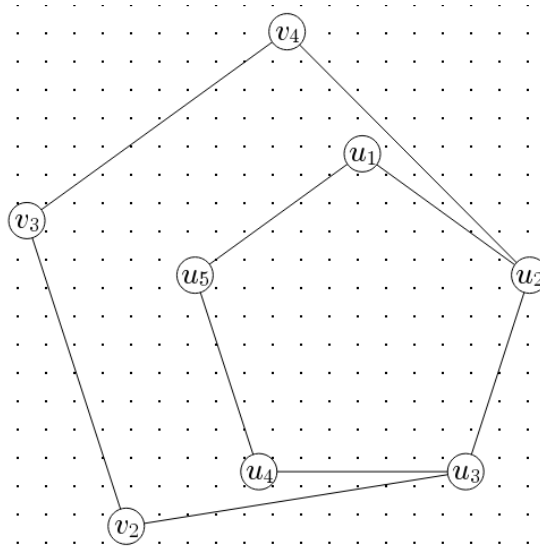
Note that if the distance between the first graph’s selected nodes is different than the distance between the second graph’s selected nodes, a portion of the second graph’s drawing will change shape. However, the actions above help preserve as much symmetry as possible.

Figure 3.19 shows the result of joining the nodes selected in Figure 3.18. Note that the polygon corresponding to the second C_5 is no longer regular, although it retains one degree of symmetry.



A four-node join of two differently-sized C_5 graph drawings
Figure 3.19

A consequence of the algorithm described above is that sometimes an unexpected join may occur, if the nodes are not selected in the correct order. For example, had the nodes been selected in the order u_2, v_5, u_3, v_1 , the resulting join would be the graph drawing shown in Figure 3.20. It is conceivable such a drawing could be desired, but a user wanting the drawing in Figure 3.19 would probably be unhappy with the drawing in Figure 3.20. If nodes are selected in an undesired order, switching to another mode and back to Join mode de-selects all nodes, providing the ability to start over. **Caution:** there is currently no “undo” feature for the join operation; if you want to undo a join, you will need to delete the vertices of the second graph, recreate it and try again.

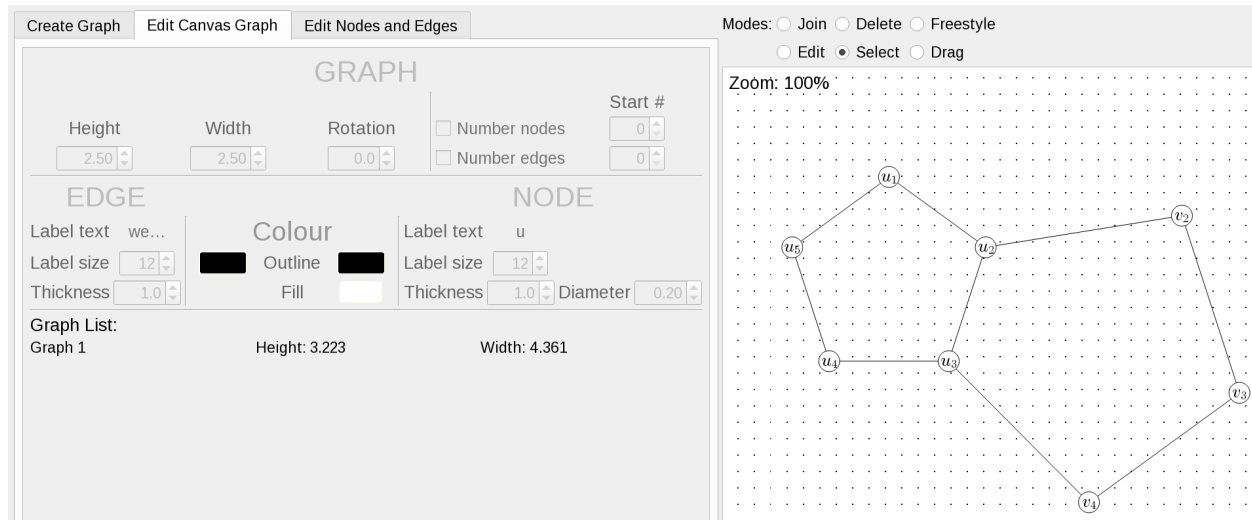


Joining after selecting nodes of two C_5 graphs in another order
Figure 3.20

Figure 3.11 shows two other controls, both at the far right side. The **Clear Canvas** button does exactly what it says: all the graph drawings on the canvas are deleted. Currently, there is no “undo” feature, thus this button should be used carefully! Below that button is found the **Snap to Grid** check-box. If this button is checked a grid of dots is drawn on the canvas, and whenever a graph is dragged to or around the canvas, when the mouse button is released the center of the graph “snaps” to the nearest grid point (assuming the graph wasn’t already exactly on a grid point; ties are broken by moving the graph upward and/or leftward). This feature is designed to aid alignment when multiple graph drawings are being combined “by hand” on the main canvas (i.e., using the “Edit” mode, described above). The size of the grid can be adjusted in Graphic’s settings, as discussed in Section 5.

3.2. The “Edit Canvas Graph” Tab

The **Edit Canvas Graph** tab of Graphic’s main window (red arrow #3 in Figure 1.1) provides some of the styling features of the **Create Graph** tab, but these modifications are applied to one or more graphs (or portions thereof!) on the canvas, as opposed to the graph in the preview area. For example, suppose the canvas contains the drawing shown in Figure 3.19; having clicked the **Edit Canvas Graph** tab, Graphic’s screen (only the part of interest is shown here) will look like Figure 3.21. The controls on the left side look similar to those on the **Create Graph** tab, except that some controls (such as **Select Graph Type**) are missing.



The Edit Canvas Graph tab with a graph to edit
Figure 3.21

Below the controls is a list of graph drawings on the canvas (in this case there is only one), where each graph's (approximate) height and width are displayed.

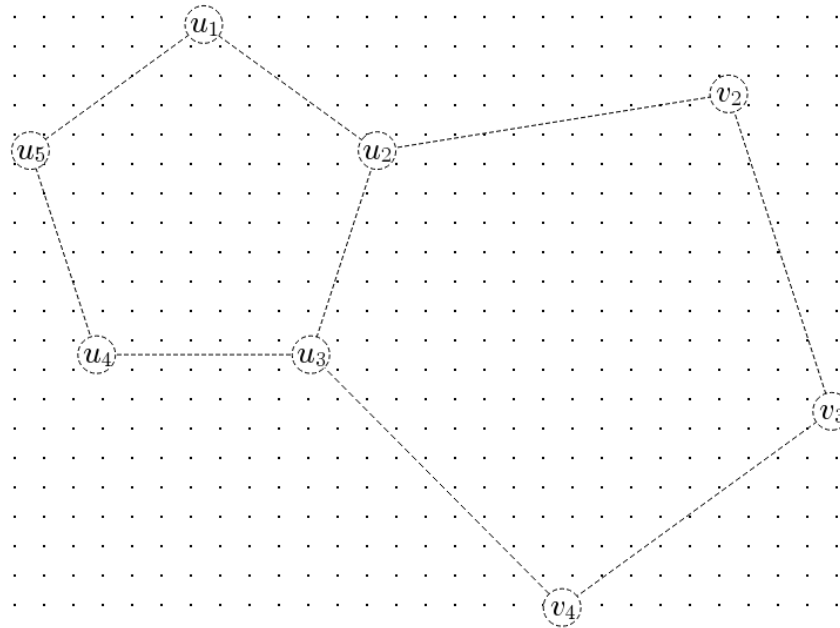
Some of the editing capabilities of this tab (such as rotation) apply only to any graphs which are entirely selected, whereas others (such as changing the colour of one or more nodes) can be applied to entire graph drawings or to portions of graph drawings. In Figure 3.21 most of the controls are greyed out, reminding the user that styling operations can't be performed until a selection has been made.

When the **Edit Canvas Graph** tab is selected, the **Select** mode is automatically made active, as seen in Figure 3.21. To select graph drawing(s), or parts thereof, hold the left mouse button down and drag out a rectangle which covers the portion of the graph(s) to be edited. As an example, we will do the following operations:

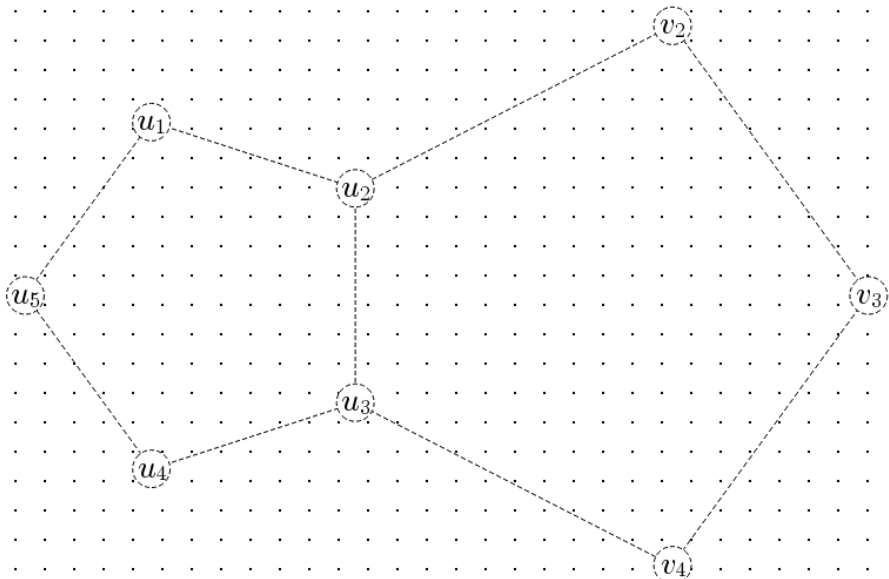
- rotate the graph so that u_2 is directly above u_3 ;
- colour the rightmost three nodes red; and
- relabel the nodes so that all labels are of the form v_i , starting with v_1 .

First, select the whole graph drawing as described above. Selected edges and nodes are drawn with dashed lines, as seen in Figure 3.22, providing the ability to confirm that the entire graph drawing was selected. Having selected the whole graph, set the rotation spinbox to 18° (recognizing that a regular pentagon has 108° corners). This produces the graph shown in Figure 3.23.

As the nodes are all still selected, it is slightly less work to relabel all the nodes next. To do this, just insert a "v" in the **Label text** text-entry box under the word **NODE**, which causes the nodes to receive labels running from v_0 to v_7 and then set the **Start #** spinbox (to the right of **Number nodes**) to 1, which adjusts the labels to run from v_1 to v_8 . (If you do not like the specific choices Graphic makes for which node gets which label these can be changed to your liking using the **Edit Nodes and Edges** tab, as described in Section 3.3.)



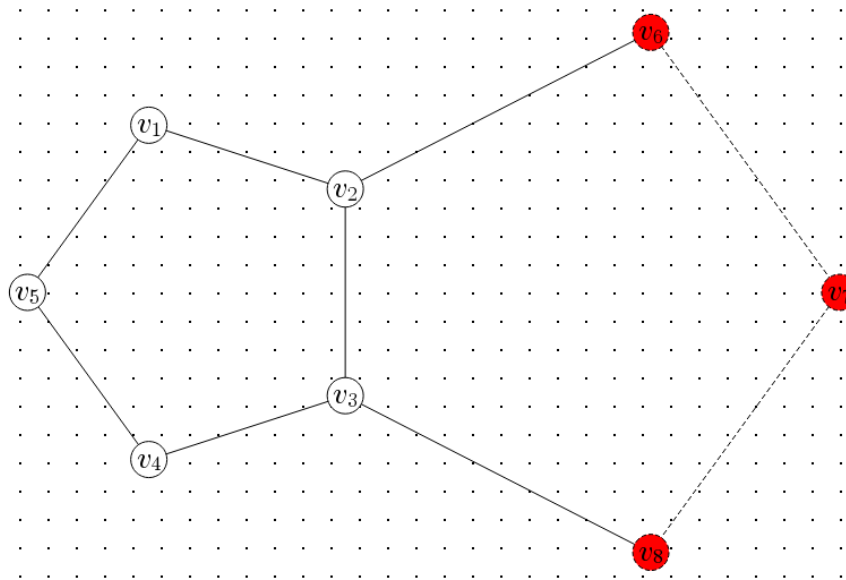
A graph drawing with all nodes and edges selected
Figure 3.22



The graph drawing of Figure 3.22 rotated 18°
Figure 3.23

To colour the three rightmost nodes, drag a selection box which contains these (and no other) nodes and then click on the colour selection box to the immediate right of the word **Fill** in the lower middle of the **Edit Canvas Graph** controls. Pick the desired colour using the colour selection pop-up window, and then dismiss this window by clicking OK. The graph

drawing should now appear as shown in Figure 3.24. (In that figure, three nodes and two edges are drawn with dashed lines because they are still selected; click on an empty part of the canvas to restore the lines to solid.)



The graph drawing of Figure 3.23 relabeled and coloured
Figure 3.24

Editing other features (edge thickness, edge colour, edge labels, label (edge and/or node) size, node diameter, node thickness, and overall width and height) can be done in analogous manners. If a control you wish to use is greyed-out, this means your selection box needs to include a bit more space on the canvas. For example, if you want to adjust the width of the graph, and after selecting the graph the width box is greyed out, this is because your selection box was too close to one or more nodes. In this case, just drag out a new selection box, making sure that all nodes of interest are completely inside the selection box.

At this time selection boxes can't be combined (sorry!).

3.3. The “Edit Nodes and Edges” Tab

With the graph drawing shown in Figure 3.24 as the only drawing on the canvas, the contents of the **Edit Nodes and Edges** tab is shown in Figure 3.25. Every node and edge on the canvas can be individually styled and labelled from this tab. (Each graph drawing on the canvas has a separate grouping of its nodes and edges, preceded by the word **Graph**;

Graph

	Line Width	Node Diam	Label	Text Size	Line Colour	Fill Colour
Node	1.0	0.20	$v_{\{2\}}$	12	Black	White
Node	1.0	0.20	$v_{\{6\}}$	12	Black	Red
Node	1.0	0.20	$v_{\{1\}}$	12	Black	White
Node	1.0	0.20	$v_{\{7\}}$	12	Black	Red
Node	1.0	0.20	$v_{\{5\}}$	12	Black	White
Node	1.0	0.20	$v_{\{3\}}$	12	Black	White
Node	1.0	0.20	$v_{\{4\}}$	12	Black	White
Node	1.0	0.20	$v_{\{8\}}$	12	Black	Red
Edge	1.0			12	Black	
Edge	1.0			12	Black	
Edge	1.0			12	Black	
Edge	1.0			12	Black	
Edge	1.0			12	Black	
Edge	1.0			12	Black	
Edge	1.0			12	Black	
Edge	1.0			12	Black	
Edge	1.0			12	Black	

The contents of the Edit Nodes and Edges tab for Figure 3.24
Figure 3.25

in this case there is only one graph drawing on the canvas, thus all the lines fall under the one and only **Graph** label.)

Each node and edge has a number of items which can be individually edited. Rows corresponding to nodes have the following controls:

- (1) the width of the circle outline, in pixels;
- (2) the diameter of the node circle, in inches;
- (3) the label[†];

[†] The syntax, capabilities and limitations of the node and edge labels is discussed in Section 4.

- (4) the size of the label, in points;
- (5) the colour of the circle outline; and
- (6) the fill colour of the node.

Rows corresponding to edges have the following controls:

- (1) the width (thickness) of the edge line, in pixels;
- (2) the label[†];
- (3) the size of the label, in points; and
- (4) the colour of the node line.

In Figure 3.25 it is relatively easy (at least once you have learned that, for example, “ $v_{\{2\}}$ ” is the way to produce “ v_2 ”) to identify the particular node you want to style. For example, to change the fill colour of the node with label v_4 , just look for $v_{\{4\}}$ in the list of nodes, and then use the Fill Colour colour picker to change the colour.

Alternatively, select Edit mode and click on a node or edge to be styled. In the Edit Nodes and Edges tab, the Node or Edge at the beginning of the corresponding row is emboldened, as seen in the upper row of Figure 3.26.

Node	1.0	0.20	$v_{\{1\}}$	12		
Node	1.0	0.20	$v_{\{7\}}$	12		

Two rows of the Edit Nodes and Edges tab, one of which is emboldened
Figure 3.26

A third possibility exists for editing the label of a node or edge. Clicking the left mouse button while the cursor is over the center of a node or the center of an edge[†] causes the label (if any) to be displayed, with a flashing text cursor surrounded by a dashed rectangle. The label appears in **typewriter** font (as also seen in the Label column in Figure 3.25), which provides visual feedback that you are editing a label. When you are finished editing a label, you can press the keyboard Escape key or Enter key, or you can click in some other location. If the label on the canvas remains in **typewriter** font this means you have a syntactically invalid label (in which case you probably want to read Section 4).

[†] It can be (slightly) difficult to click close enough to the center of an edge to activate the label editing function; a steady hand and a keen eye will help.

4. All About Node and Edge Labels

As already seen, nodes and edges can be given labels. If the labels automatically assigned by Graphic meet your needs, there is no need to read this section. But if your needs are more complex, or you wonder what Graphic can do, read on.

The **Create Graph** and **Edit Canvas Graph** tabs can assign numeric labels or labels comprised of a single letter and a numeric subscript, but Graphic allows the user to assign far more complex labels, should the need exist.

Labels can be individually edited either by using the **Edit Nodes and Edges** tab or by selecting the **Edit** canvas mode and clicking on the center of a node or an edge.

The syntax for specifying labels is, essentially, a subset of the capabilities of \TeX 's math mode. The rest of this section is broken into three parts: one for people who do not know how to typeset mathematics in \TeX , one for people familiar with \TeX math mode, and finally one for everyone to read.

In the next three sub-sections, the use of “ \TeX ” refers to all of plain \TeX , \LaTeX and \ConTeXt . Additional information related to the differences between these systems is found in Section 4.4. Graphic users who don't use any of these systems do not need to read this section.

4.1. \TeX math mode syntax for people who don't know \TeX

\TeX is a typesetting system designed with mathematics in mind. Due to the fact that many users will prefer to export graph drawings as TikZ for inclusion into \TeX documents, a design goal of Graphic was to have the on-canvas graph drawings look as similar as possible to the ones produced with TikZ and \TeX . (You may have noticed that digits in labels used an upright font (“1”) and letters used an italic font (“ v ”); this is the standard way for \TeX to typeset numbers and variables in math mode.)

By now you have probably noticed the use of braces (“{” and “}”) and underscore (“_”), as seen in Figure 3.26 and other figures. The underscore signifies that the following text is a subscript. The braces are used to surround the desired subscript text, in case the subscript is more than one letter or digit. For example, `v_{ab}` will generate the label v_{ab} , and `v_{ab}z` will generate the label $v_{ab}z$. Note that although the braces are entered as part of the label, they do not show up in the output.

Some people might want superscripts; these are generated analogously to subscripts, except that “^” is used instead of “_”. For example, the label text `v^{ab}` will generate the output v^{ab} . A math formula can have both subscripts and superscripts; for example, `v^{a}_{b}` generates the output v_b^a ; unfortunately, as noted in Section 4.3, Graphic currently displays uglier text on the canvas.

In \TeX , text to be typeset as mathematics is surrounded by a pair of “\$” characters[†]. \Graphic supplies these surrounding $\$ \dots \$$ characters “for free”, but users should be aware that if a dollar sign is desired in the output, it must be entered with a preceding backslash character. For example, $\backslash \$^{\text{\code{a}}}$ will generate the output $\a and $\text{\code{b}}_{\backslash \$}$ will generate the output $b_{\$}$. (At time of writing, \Graphic will accept dollar signs without preceding backslash characters, but if you create a $\text{\code{.tikz}}$ file for inclusion in a \TeX document with a non-backslashed dollar sign, Bad Things will happen when you try to “compile” your \TeX document.)

Careful readers might have noticed that Figure 3.14 displays a label with visible brace characters (specifically, the label “ $v_{\{z\}}$ ”). In the rare event that you want to see a brace character, you can precede a brace with a backslash character. The previous label was generated with $\text{\code{v}}_{\backslash \{\text{\code{z}}\}}$. Typing “ $\backslash \{$ ” or “ $\backslash \}$ ” causes the brace to lose its “grouping” effect, so two sets of braces were required in this example: one to group the subscript, and one as part of the subscript itself.

\TeX (and \Graphic , to a lesser degree) handles more complex math typesetting than is described here, but this subsection should cover most normal usage.

4.2. Differences between \TeX math mode syntax and \Graphic label syntax

In the \TeX typesetting system, math formulae are surrounded with \$ signs; for example, the input “ $\text{\code{$v_1$}}$ ” instructs \TeX to typeset “ v_1 ”. \Graphic assumes all labels are to be typeset as math, and thus no \$ signs are needed. Further, \$-signs will not be processed as \TeX would, should you enter them; if you wish to have a dollar sign in a label, precede it with a backslash.

\Graphic does not allow the use of symbols which require \TeX control sequences, such as α ($\backslash \text{\code{alpha}}$) and \aleph ($\backslash \text{\code{aleph}}$). But if for some (strange?) reason a user desires a “ \sim ” or “ $_$ ” in a label, this can be obtained by typing “ $\backslash \sim$ ” or “ $\backslash _$ ”, respectively. However, exported $\text{\code{.tikz}}$ files will have this text inserted literally, and thus will not be valid \TeX code. (Of course, a small bit of hand-editing can fix the $\text{\code{.tikz}}$ file.)

4.3. Visual differences between \Graphic ’s labels and \TeX math output

There are currently three differences of note between the way labels are displayed in \Graphic (which is implemented with the Qt5 user interface toolkit) and how they are typeset by \TeX .

(1) As of (at least) Qt 5.15.3, when a subscript and a superscript are both specified for the same symbol, the subscript and superscript are not aligned vertically, as they are in \TeX [‡].

[†] In fact, things are a bit more complex than this, depending on whether plain \TeX , \LaTeX , or \ConTeXt is used. \Graphic does not support any other methods of typesetting mathematics, but at time of writing, all three accept \Graphic ’s output.

[‡] That is, the authors of \Graphic have not yet discovered how to convince Qt5 to align the subscript and the superscript vertically. Solutions from Qt wizards are welcome.

For example, typesetting the text “`a_1^2`” in \TeX math mode produces a_1^2 , but Graphic produces a_1^2 .

(2) \Qt5 does not (seem to) allow sub-subscripts or super-superscripts. With \TeX , one can produce u_{vw} using “`u_{v_w}`”, but currently Graphic would display u_{vw} . This is relatively inconsequential, but may nonetheless be fixed at some point in the future.

(3) When using TikZ to draw graphs in \TeX , the node labels are vertically centered inside the node circle. This means that the baseline of a label may vary, depending on the vertical height of the label. At time of writing, all Graphic labels are positioned at the same vertical height within the node circles. This produces slight differences: in Graphic, a label consisting of a number appears somewhat higher in the circle than when the exported TikZ is processed by \TeX . This also is relatively inconsequential, but may nonetheless be fixed at some point in the future.

4.4. Details of Graphic’s TikZ output for \TeX , \LaTeX and \ConTeXt Users

If you don’t plan to use Graphic’s TikZ output (either because you don’t use \TeX , \LaTeX , or \ConTeXt , or because for some reason you prefer to import a graph drawing in another way), you do not need to read this subsection: skip forward to Section 5.

Depending on the \TeX “flavour” in use (plain \TeX , \LaTeX , or \ConTeXt) some of the structure of TikZ code varies. Graphic allows users to specify their \TeX flavour using the **Settings/Graphic Settings...** menu item. Depending on this selection, there are some relatively minor differences in the TikZ output, as explained next.

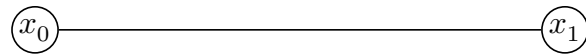
In addition to the TikZ code generated by Graphic to draw the graph itself, a \TeX document must include the TikZ libraries:

- (1) Plain \TeX documents must have the line `\input tikz` somewhere in the document before a TikZ graphic; normally, this would be found at the very beginning of the document. When outputting TikZ output, Graphic outputs some additional \TeX code which automatically executes the `\input tikz` command if necessary.
- (2) \LaTeX documents must have the line `\usepackage{tikz}` in the **preamble**. For TikZ output, a \LaTeX **comment** is output by Graphic reminding the user that this `\usepackage` is necessary.
- (3) \ConTeXt documents must have the line `\usemodule[tikz]` somewhere before a TikZ graphic. Normally `\usemodule` is placed in the **preamble** before `\starttext`, but it also works after `\starttext`. Since it is harmless to repeat the `\usemodule[tikz]` command, Graphic outputs that command just before the TikZ code to draw the graph.

For Graphic’s TikZ output, plain \TeX (with no add-on packages) has one important shortcoming compared to \LaTeX and \ConTeXt : there are no commands which make it easy to change just the font size. Consequently, the plain \TeX output contains the \LaTeX font-size change command(s) plus a “replacement” macro which outputs a diagnostic message to the log file and standard output, but otherwise does nothing. Depending on the font

sizes in the graph and the font sizes of the document, this may suffice; otherwise some effort will be required by plain T_EX users.

To expand on this, complete plain T_EX, L^AT_EX and ConT_EXt examples are shown below. All of them contain TikZ code for the graph shown in Figure 4.27.



A path of length two with node labels
Figure 4.27

Figure 4.28 shows the code output by Graphic for plain T_EX. Note that the code in **red** is added by Graphic specifically for plain T_EX (in addition to the actual TikZ code to draw the graph), and code in **magenta** was code added by hand to produce a completely self-contained plain T_EX input file.

```
% NOTE: you may need to adjust font sizes yourself
% or define your own \fontsize...\selectfont macro
% to do what you want.
\def\fontsize#1#2\selectfont{%
  \message{Changing \string\fontsize{#1}{#2}
  \string\selectfont\space to \string\rm}%
  \rm
}
\ifdefined\tikzpicture\else\input tikz \fi
\tikzpicture[x=1in, y=1in, xscale=1, yscale=1,
  n/.style={fill=white, draw=black, shape=circle,
  minimum size=0.2in, inner sep=0, font=\fontsize{12}{1}\selectfont,
  line width=0.0071in},
  e/.style={draw=black, line width=0.0071in},
  l/.style={font=\fontsize{12}{1}\selectfont}]
\node (v0) at (-1.1500,0.0000) [n] {$x_{0}^{\{ \}}$};
\node (v1) at (1.1500,0.0000) [n] {$x_{1}^{\{ \}}$};
\path (v0) edge[e] node[l] {$$} (v1);
\endtikzpicture
\end{document}
```

Augmented Plain T_EX Code
Figure 4.28

Similar to Figure 4.28, Figure 4.29 shows the TikZ code output by Graphic for the graph shown in Figure 4.27, where L^AT_EX-specific content added by Graphic is shown in red and some code added by hand to make a completely self-contained L^AT_EX document is shown in magenta.

```
\documentclass{minimal}
\usepackage{tikz}
\begin{document}
% NOTE! \usepackage{tikz} is needed in preamble!
\begin{tikzpicture}[x=1in, y=1in, xscale=1, yscale=1,
    n/.style={fill=white, draw=black, shape=circle,
    minimum size=0.2in, inner sep=0, font=\fontsize{12}{1}\selectfont,
    line width=0.0071in},
    e/.style={draw=black, line width=0.0071in},
    l/.style={font=\fontsize{12}{1}\selectfont}]
\node (v0) at (-1.1500,0.0000) [n] {$x_{0}^{\{ \}}$};
\node (v1) at (1.1500,0.0000) [n] {$x_{1}^{\{ \}}$};
\path (v0) edge[e] node[l] {$$} (v1);
\end{tikzpicture}
\end{document}
```

Augmented L^AT_EX Code
Figure 4.29

Last, but not least, Graphic's ConT_EXt output is shown in Figure 4.30; the red and magenta colours have the same meanings as in the plain T_EX and L^AT_EX examples.

```
\starttext
\usemodule[tikz]
\starttikzpicture[x=1in, y=1in, xscale=1, yscale=1,
    n/.style={fill=white, draw=black, shape=circle,
    minimum size=0.2in, inner sep=0, font=\switchtobodyfont[12pt],
    line width=0.0071in},
    e/.style={draw=black, line width=0.0071in},
    l/.style={font={\switchtobodyfont[12pt]}}}
\node (v0) at (-1.1500,0.0000) [n] {$x_{0}^{\{ \}}$};
\node (v1) at (1.1500,0.0000) [n] {$x_{1}^{\{ \}}$};
\path (v0) edge[e] node[l] {$$} (v1);
\stoptikzpicture
\stoptext
```

Augmented ConT_EXt Code
Figure 4.30

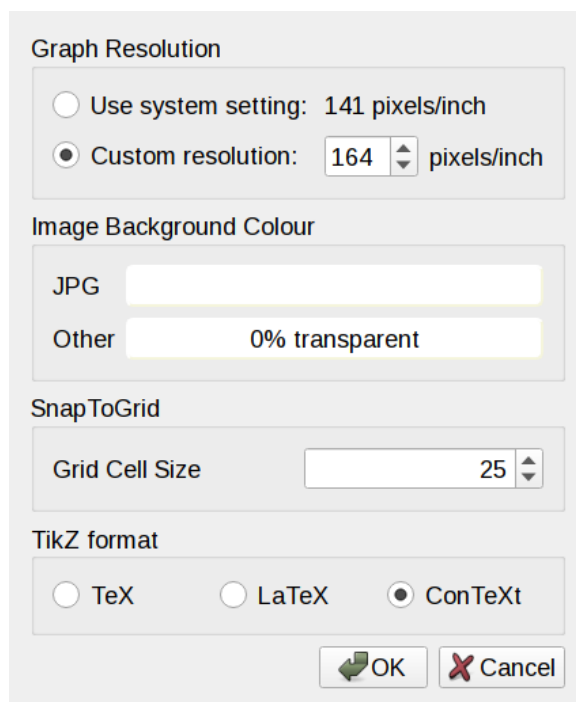
There is one last point which should be explained. Graphic’s output for subscripted node and edge labels has an extra “ $\wedge\{\}$ ”; that is, instead of (for example) “ $v_{\{1\}}$ ”, Graphic outputs “ $v_{\{1\}}^{\wedge\{\}}$ ”. With `pdftex` and `pdflatex` (but not `context`), an empty superscript causes the subscript to be typeset a bit lower, which this author prefers (zoom in and compare v_1 to v_1 to see the difference). \TeX and \LaTeX users who disagree will have to hand edit their TikZ output. ConTeXt users who would like the subscripts to be similarly lowered may have to resort to trickery such as editing the labels to look like `$v_1^{}$` , at the risk of horrifying ConTeXt experts. (No doubt there is a more “proper” ConTeXt way of doing this.)

5. Graphic’s Settings and Miscellaneous Controls

This section explains features of Graphic not already mentioned.

5.1. Graphic’s Settings

Graphic has a few settings available via the Settings pop-up window (available from the Settings menu). This window is shown in Figure 5.31.

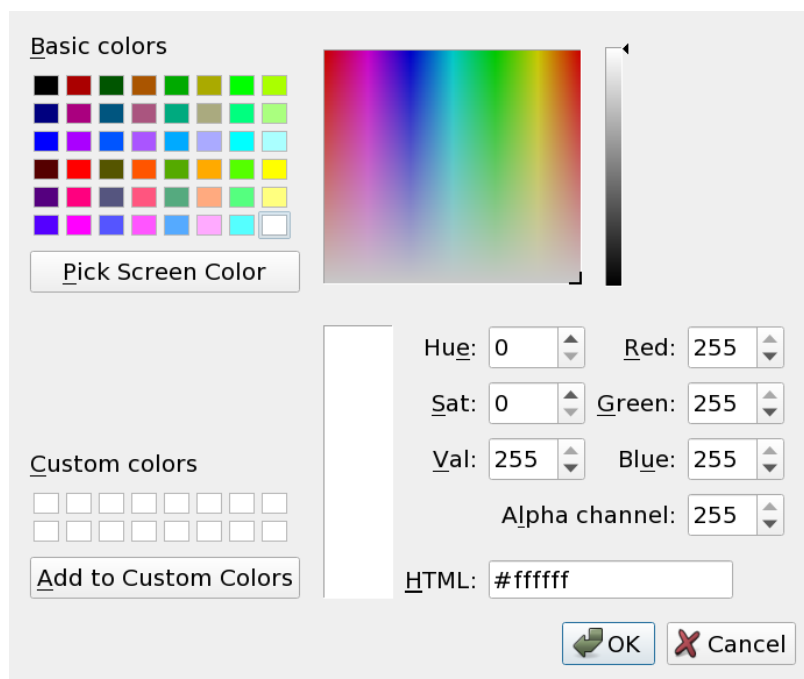


Graphic’s Settings window
Figure 5.31

The top two lines of the settings window show the pixels/inch ratio used by Graphic. In order to show graph drawings at a 1:1 scale, Graphic needs to know how many pixels there are per inch of screen. Normally Graphic can figure this out by itself, but when external monitors are used sometimes it does not compute the correct value. Figure 5.31 shows that it believes the number of pixels per inch is 141 (which is the correct number for the laptop screen in use at the time), but since that is incorrect for the external monitor, the **Custom resolution** radio button was selected and the correct value (164 in this example) was entered in the spinbox.

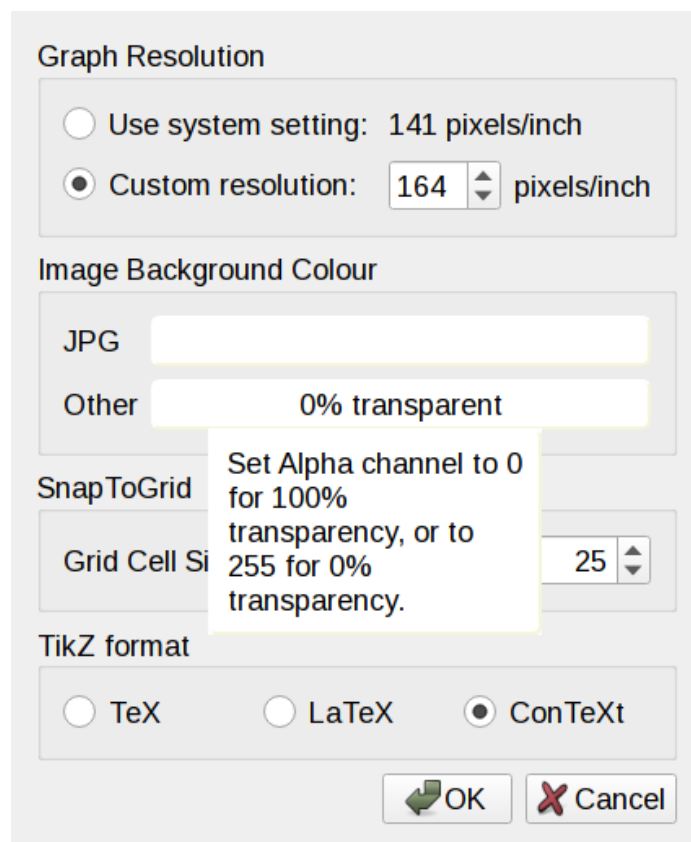
At time of writing, if the pixels per inch setting is changed, the graph in the preview area changes size accordingly, but any graphs on the canvas do not. Consequently it is best to make any changes to that setting when Graphic is started.

When a graph drawing is exported as an image file, it may be desirable to be able to specify the background colour. While (for example) PNG images may have a transparent background, JPEG images can not, so there are two different colour selections available. Clicking on the **Other** colour swatch brings up a colour selection window, as seen in Figure 5.32[†]. The figure shows a spinbox labelled **Alpha channel**, which allows a value ranging from 0 (completely transparent background) to 255 (completely opaque background). Should you forget these numbers, a “tool-tip” reminder is displayed when you hover the mouse over the **Other** colour swatch, as seen in Figure 5.33.



The colour picker widget
Figure 5.32

[†] Depending on your operating system, the colour picker window may look somewhat different.



The tool-tip reminding users how to set background transparency
Figure 5.33

The colour picker for JPEG is similar, except it lacks the ability to pick the alpha channel.

The second last setting (as seen toward the bottom of Figure 5.31) defines the size of the grid cell, as used by the main canvas’ **Snap to Grid** feature. Users who find the snapping behaviour useful when constructing compound graphs on the canvas may wish to adjust this size to suit their work. Recall that when a graph is dragged on the canvas, the center point of the graph (as defined by the operations done to produce the graph, which may not place the center in the visual perception of the center) is moved to the closest grid point to the upper left of where the center is when the mouse button is released.

The last setting is fairly self-explanatory: click on the radio button corresponding to the type of TikZ output you desire. If you don’t use TikZ, then the setting is irrelevant to your use of Graphic.

Finally, instead of saving TikZ output to a file, Graphic will write the graph’s TikZ representation to standard output (the terminal, if you started Graphic from a terminal) when you type **Ctrl-T** in the canvas area of the window.

5.2. Miscellaneous Controls: Zooming

Both the preview area and the main canvas can be zoomed in or out, in case the user is working on a very small graph, or a very large graph (or, for the main canvas, a large collection of graphs).

To zoom, first click the left mouse button in the preview area or main canvas, as desired. Then, to zoom in, type **Ctrl=**, and to zoom out, type **Ctrl-**.

6. Continuing Development

While Graphic does not have any paid software developers, updates will be made when deemed useful or necessary, as time permits. If you identify problems or have feature requests (either for the program or for this document), please email [Jim Diamond](mailto:Jim.Diamond@ndolam.com). Better yet, if you are able to contribute features or improvements to Graphic please look at the source code (found at <https://github.com/Ndolam/Graphic> and then get in touch with Jim Diamond.

As of May 2026, Qt5 is no longer the current version of the Qt toolkit, which is used by Graphic. Porting Graphic to Qt6 will take some time due to some incompatible changes between Qt5 and Qt6; this will happen if and when time and interest permit.