

QGeNle Modeler

USER MANUAL

Version 4.1.R2, Built on 4/27/2024
BayesFusion, LLC

This page is intentionally left blank.
Remove this text from the manual
template if you want it completely blank.

1.	Read me first	7
2.	Hello QGeNIe!	9
3.	Introduction	33
3.1	Guide to QGeNIe manual	34
3.2	QGeNIe	34
3.3	SMILE Engine	35
3.4	BayesBox	36
3.5	BayesMobile	37
3.6	Distribution information	37
3.7	QGeNIe on a Mac	39
3.8	QGeNIe on Linux	41
3.9	Non-Latin Alphabets in QGeNIe	41
3.10	Copyright notice	42
3.11	Disclaimer	42
3.12	Acknowledgments	43
4.	Decision-theoretic modeling	45
4.1	Decision analysis	46
4.2	Probability	46
4.3	Bayesian networks	47
4.4	Bayesian updating	50
4.5	Changes in structure	50
5.	Building blocks of QGeNIe	53
5.1	Introduction	54
5.2	QGeNIe workspace	54
5.2.1	Introduction	54
5.2.2	The menu bar	56
5.2.3	Graph view	57
5.2.4	Tree view	72
5.2.5	Status bar	76
5.2.6	Output window	77
5.2.7	Help menu	78
5.3	Components of GeNIe models	79
5.3.1	Node types	79

5.3.2	The DeMorgan gate	80
5.3.3	The CAST gate	95
5.3.4	Submodels	97
5.3.5	Arcs	106
5.3.6	Node status icons	107
5.3.7	Text boxes	109
5.3.8	Annotations	110
5.4	Model and component properties	112
5.4.1	Network properties	112
5.4.2	Submodel properties	117
5.4.3	Tools menu and Standard toolbar	119
5.4.4	Node properties	121
5.4.5	Arc properties	128
5.5	Visual appearance, layout, and navigation	128
5.5.1	Introduction	128
5.5.2	Viewing nodes in the Graph View	129
5.5.3	Zooming and full screen mode	132
5.5.4	Format toolbar and Layout menu	134
5.5.5	Graph layout functions	138
5.5.6	Selection of model elements	140
5.5.7	Model navigation tools	145
5.5.8	In-degree analysis	147
5.6	Saving and loading models in QGeNIe	148
5.6.1	Introduction	148
5.6.2	File menu	155
5.6.3	QDSL file format	160
5.7	Inference algorithms	160
5.7.1	Introduction	160
5.7.2	Node menu	161
5.7.3	Network menu	162
5.7.4	Clustering algorithm	164
5.7.5	Relevance-based decomposition	164
5.7.6	EPIS Sampling	164
5.8	Keyboard shortcuts	164
6.	Using QGeNIe	167
6.1	Introduction	168
6.2	Applications of qualitative probabilistic modeling	168
6.3	Bayesian networks	170
6.3.1	Building a qualitative Bayesian network	170

6.3.2	Simple interaction patterns	170
6.3.3	Entering and retracting evidence	172
6.3.4	Controlling values	175
6.3.5	Viewing results	179
6.3.6	Structural analysis	182
6.3.7	Case Manager	199
6.4	Most Effective Actions calculation	202
6.4.1	Introduction	202
6.4.2	Enabling Most Effective Actions calculation	203
6.4.3	Most Effective Actions calculation	205
6.5	Dynamic Bayesian networks	208
6.5.1	Introduction	208
6.5.2	Creating DBN	209
6.5.3	Inference in DBNs	215
7.	Resources	229
7.1	Books	230
7.2	Research papers	230
7.3	Conferences	231
7.4	Model repository	231
7.5	Social Media	232
7.6	References	232
Index		237

This page is intentionally left blank.
Remove this text from the manual
template if you want it completely blank.

Read me first

1 Read me first

Welcome to QGeNIe manual, Version 4.1.R2, Built on 4/27/2024.

This manual is available in CHM, PDF and HTML formats, all available at <https://support.bayesfusion.com/docs/>.

CHM version of QGeNIe manual is also distributed with QGeNIe.

If you are new to QGeNIe and would like to start with an informal, tutorial-like introduction, please start with the [Hello QGeNIe!](#) section. If you are an advanced user, please browse through the *Table of Contents* or search for the topic of your interest.

Hello QGeNle!

2 Hello QGeNIe!

This section offers an informal introduction to QGeNIe, similar to the light introduction to the C programming language offered by Brian Kernighan and Dennis Ritchie in their milestone book (see Kernighan & Ritchie, 1988). We will show you how to create a simple qualitative Bayesian network model, how to save and load it, and how to use it to answer simple questions. Once you have made yourself familiar with QGeNIe in this informal way, you can proceed with the *Elements of QGeNIe* chapter, which offers a thorough introduction to various elements of QGeNIe.


QGeNIe is an interactive program that allows for rapid creation of causal models in uncertain domains. These models represent propositions by means of nodes, which always take two possible values: *True* and *False*. Even though the underlying methodology is quantitative and grounded in probability theory, models created in QGeNIe use no numbers and rather qualitative sliders, colors, and gradients. Degrees of truth of propositions are represented by colors. Mathematically speaking, they represent the probability of the state *True* (or *False* - it is the users' choice). While QGeNIe users can define the color scale themselves, the default color scale is a range between red and green, representing undesirable and desirable states. QGeNIe allows for an interactive exploration of the models, examining the effects of observations and manipulations of individual variables. The focus of reasoning in QGeNIe are not exact probabilities but rather sign and order of magnitude effects that can be perceived by means of colors.

We will demonstrate the basic functionality of QGeNIe on a simple model that will describe relationship among four variables: *High Productivity*, *Hot Weather*, *Climate Control in the Building* and *Work To Do*. Each of these variables is propositional and can take two states: *True* and *False*. Through this modeling effort, we would like to create a model that will answer the following questions: *What is the expected impact of failure of the climate control in the building on productivity on a hot day and on a normal day respectively?* We would also like to obtain the conditional probability table for the variable *High Productivity* for a further use and refinement in GeNIe. While this example contains only four variables, it illustrates all basic concepts, which once understood can be used in building more complex models. Please keep in mind that the functionality covered in this section merely touches what you can do with QGeNIe and gives you just a taste of qualitative Bayesian modeling.

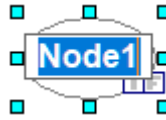
If you have not already started QGeNIe, please start it now.

A. Let us create the nodes that will represent the variables in our model.

The [Tool Menu](#) shows a list of model elements that you can create. These are also displayed as buttons on the [Standard Toolbar](#).

Select Node button () from the [Standard Toolbar](#) or [Tool Menu](#).

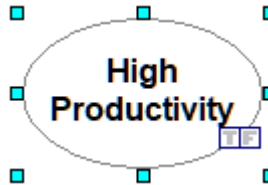
The *Node button* will become recessed and the cursor will change to an arrow with an ellipse in bottom right corner. Move the mouse to a clear portion of the screen inside QGeNIe window (the main model window is called the [Graph View](#)) and click the left mouse button. You will see a new node appearing on the screen as shown below:




The small squares around the node indicate that the node is selected. The most recently created node is automatically selected. You can also select any node by clicking on it. You can change the size of the selected node by dragging the small squares.

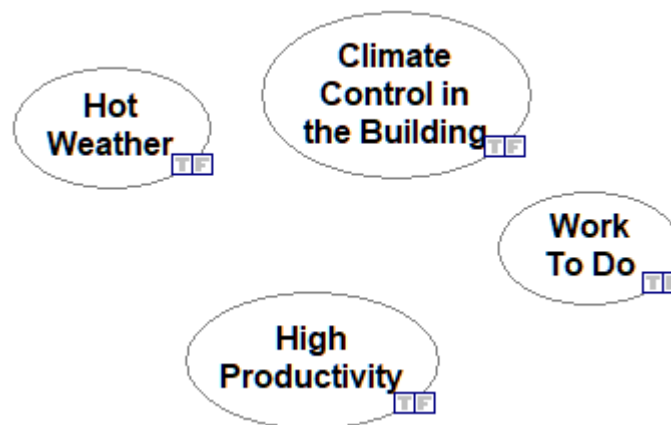
Once you have drawn the node on the [Graph View](#), the *Node* button on the toolbar will become normal again and the *Select Objects* button will become recessed.

QGeNIe allows you to name nodes. Names are simply strings of any characters and any length. QGeNIe assigned the node that you have just created the name *Node1*. QGeNIe also placed the newly created node's name in *Edit* mode immediately, so you can enter a more descriptive name. In this case, we want to name it *High Productivity*.




All QGeNIe variables are binary and should refer to some proposition. The truth of this proposition will be the subject of inference in QGeNIe. It is a good heuristic to think of these propositions as desirable and undesirable. It is best to assign variable names that are meaningful and self-explanatory. Please note that QGeNIe does not put any limitations on the length of the names. When working with the model, posterior probabilities will be displayed by means of colors. Typically, one assigns green color to desirable propositions and red color to undesirable ones. Red has been found to draw user's attention and we advise that it be used to undesirable propositions. QGeNIe allows its user to define a node coloring scheme. When defining this scheme, it is a good idea to follow the meaning of the majority of nodes. Single nodes can be designated as having the reverse meaning and the coloring scheme will be reversed for them.

Similarly, we add nodes for the variables *Hot Weather*, *Climate Control in the Building* and *Work To Do*. It is worth mentioning here that if you want to draw multiple model elements of the same type, then you can avoid having to select the node button again and again by double-clicking on a [Standard Toolbar](#) button instead of single-clicking it the first time. This will place you in "sticky mode," in which the tool button stays recessed and you can draw multiple elements of that type. You can return to normal mode by clicking on the *Select Objects* button () or clicking on the recessed button again. After adding the three nodes, we enter their names, resize them to look pretty and display the entire name, and move them to their destination positions using mouse drag and drop functionality. Here is what the effect of this operation could look like:

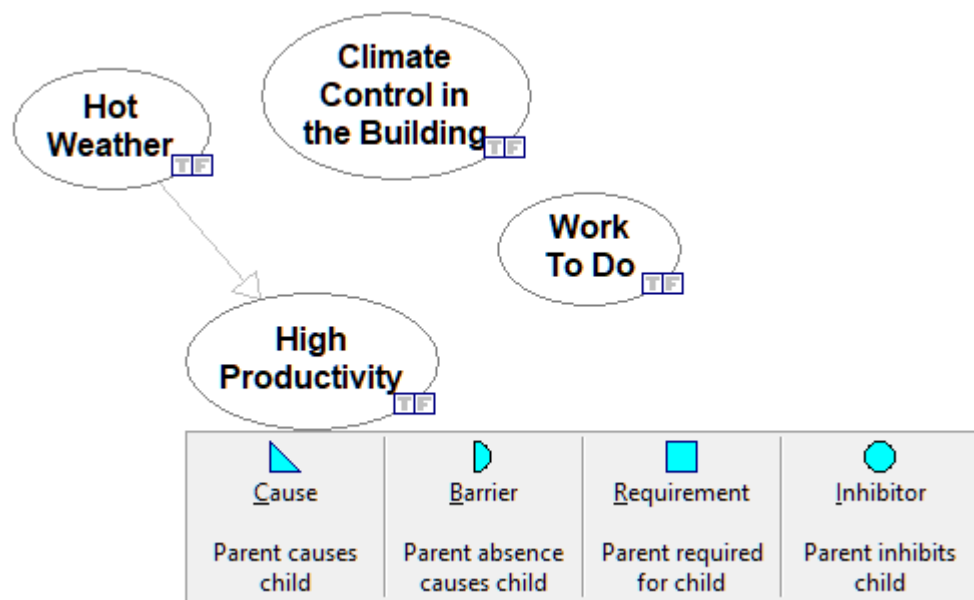


B. Let us specify the causal connections between the nodes.

In order to represent the fact that each of the three nodes (*Hot Weather*, *Climate Control in the Building* and *Work To Do*) influences the node *High Productivity*, we will add influence arcs to the model.

Click on the Arc () tool (note that the cursor changes), then click on the *Hot Weather* node, hold the left mouse button and drag the mouse to the node *High Productivity*, and release the button anywhere within the new node.

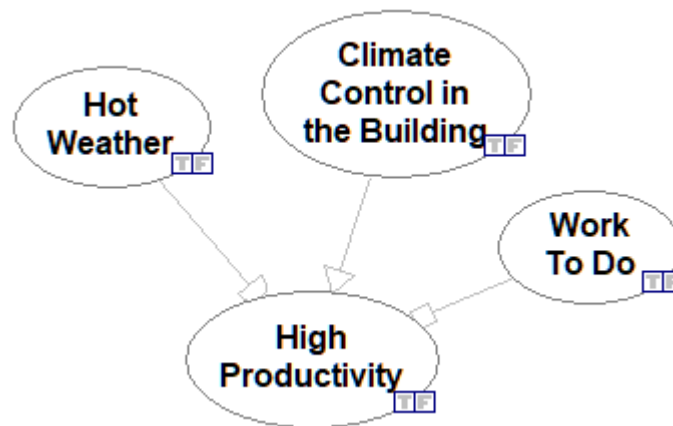
GeNIe will draw an arc from *Hot Weather* node to *High Productivity* and display the following dialog:



which allows you for specifying the type of causal relationship between *Hot Weather* and *High Productivity*. There are four possible types of causal relationships: *Causes*, *Barriers*, *Requirements*,

and *Inhibitors*. Causes make their effects more likely, i.e., have a positive influence on their effects. Barriers have a negative effect. Requirements are necessary for the effects to happen. Inhibitors, when true, make the effect impossible. Each of these four types of relationships is imperfect, so the effect may happen even if the requirement is not present and may happen even if the requirement is absent. Similarly, an inhibitor may be imperfect and the effect may take place even if it is present or not take place even if it is absent.

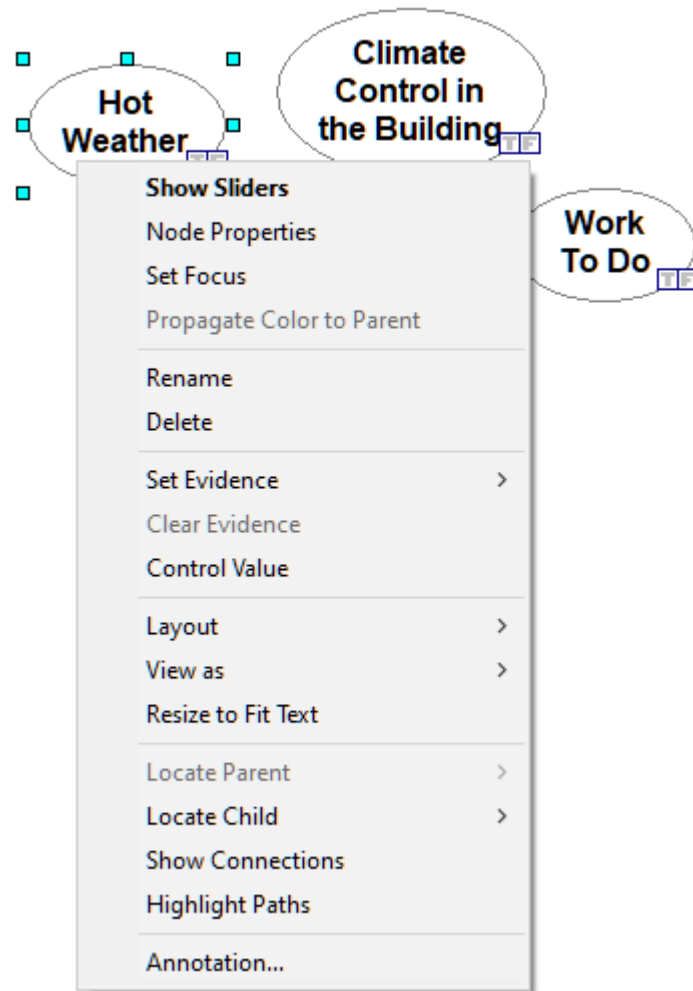
In case of the causal relationship between *Hot Weather* and *High Productivity*, it is fair to assume that it is a *Barrier* - hot weather will tend to decrease productivity. We choose *Barrier* thus. Similarly, we draw arcs from *Climate Control in the Building* and *Work To Do* to *High Productivity*, making them a *Cause* (climate control should increase productivity) and a *Requirement* (existence of work to be done is a necessary condition for productivity) respectively. We obtain the following directed graph:



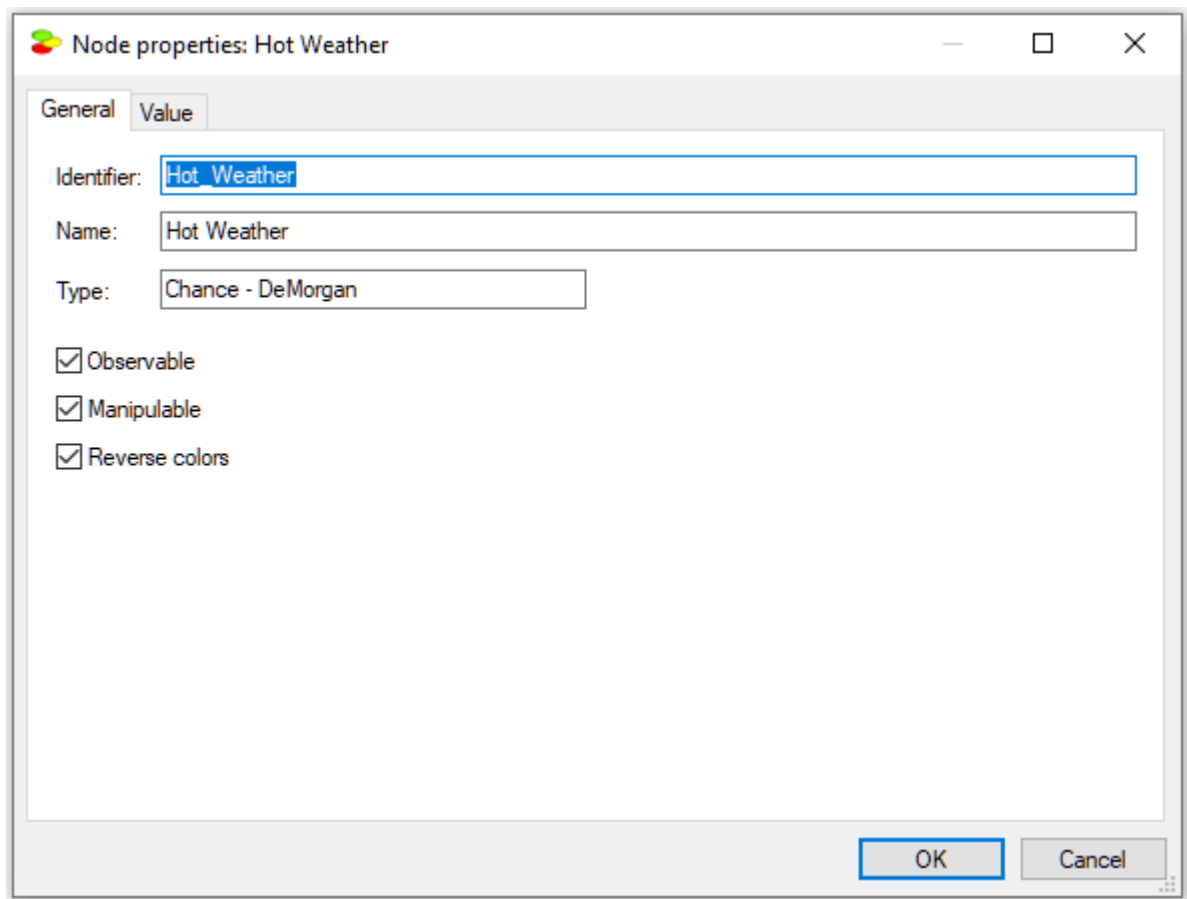
Please note that the heads of the arrows are all different, indicating the types of relationships between a cause and an effect.

C. Setting the desirable state for each node.

QGeNIe uses colors to show desirable and undesirable states. By default, these are the states *True* but this can be changed for each node individually. In our model, the state *True* of each of the variables is desirable, except for the node *Hot Weather*. Let us change the desirable state for node *Hot Weather* to *False*. To do that, we right-click on the node and choose Node properties:



Within node's properties, please check the *Reverse colors* check box:

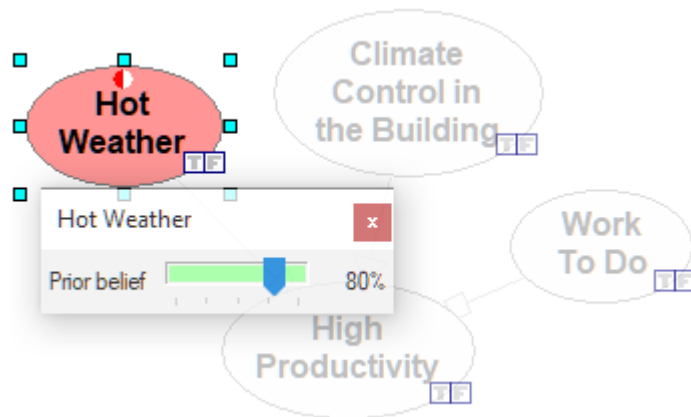


This will cause the node colors to follow the state *False*. Zero probability of *Hot Weather* will be shown as green and probability one will be shown as red.

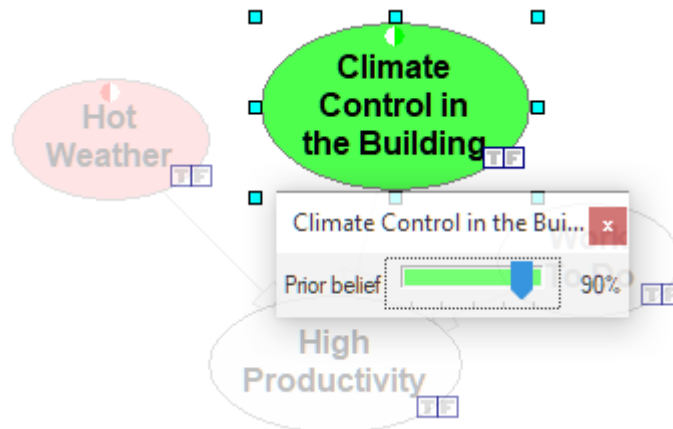
D. Now, let us define the numerical properties of the model, i.e., nodes and the interactions among them.

There are two types of parameters in QGeNIe: (1) Node beliefs (probabilities), and (2) Interaction probabilities.

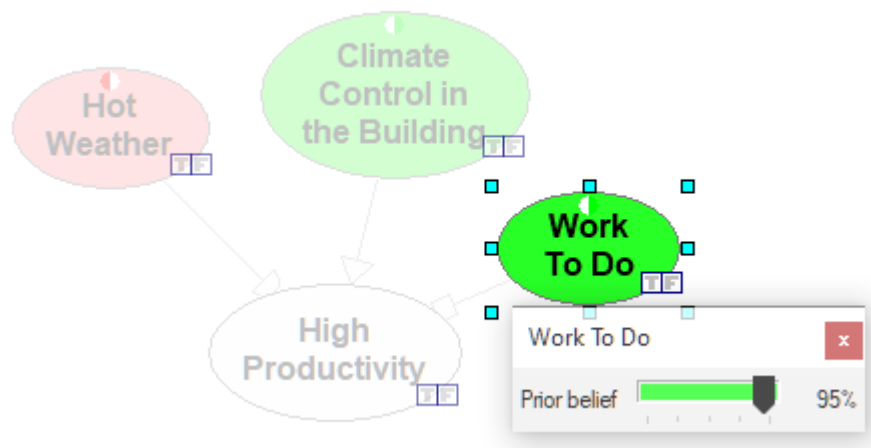
Node probabilities express either the prior probability of the proposition expressed by the node (for those nodes that have no parents in the model graph) or leak, which is the probability of the proposition when each of the node's parents is inactive. To change this probability for any node, just double-click on the node. This will result in dimming every node in the model except for *Hot Weather* and displaying a slider for changing the prior belief (prior probability) of the proposition *Hot Weather*:



The prior probability of (degree of belief in) the proposition can be adjusted by moving the slider. You can also move the slides by finer steps (single percentiles) using the horizontal arrows on your keyboard. Let us move it to the value of 80%, which means that there is a prior probability of 80% of hot weather. As we move the slider, the color of the node changes. We repeat this for the node *Climate Control in the Building*:

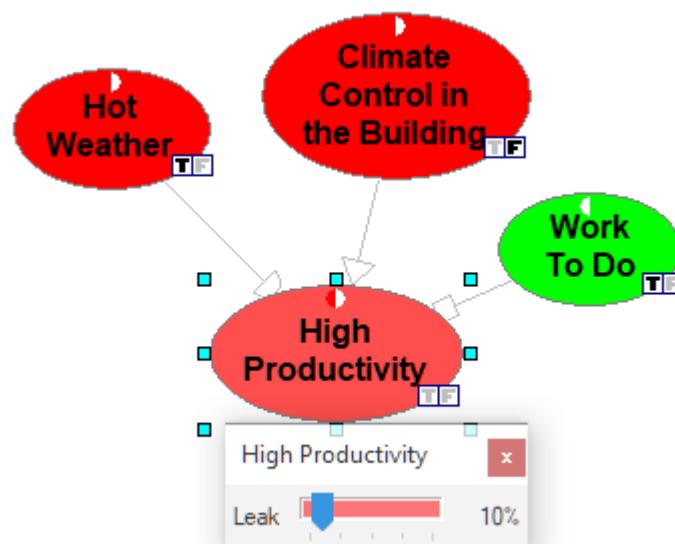


stating that there is 90% chance that climate control is operating in the building and for the node *Work To Do*:



stating that 95% of the time there is work to do.

For any node that has parents in the graph, we specify two parameters: The **Leak** and the **interaction probabilities**. We will explain the meaning of these probabilities in section [The DeMorgan gate](#) in this manual. For now, we ask your patience and trust in the sound probabilistic meaning of these parameters. The only node with parents in this simple model is *High Productivity*. Double-clicking on this node invokes the slider for the *Leak* probability.

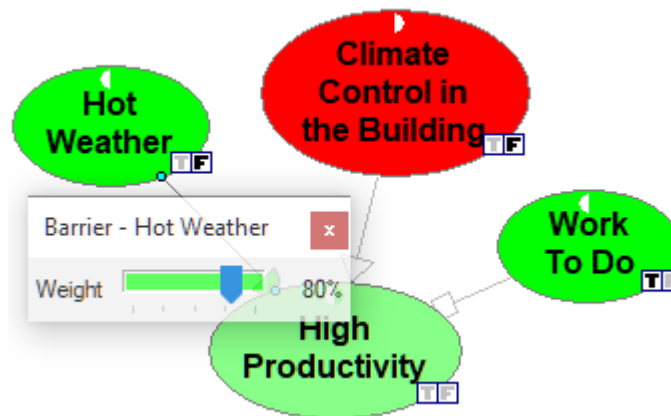


The *Leak* expresses the probability that *High Productivity* is going to be *True* when all of its parents are in their inactive states, i.e., the weather is going to be hot, there is going to be no climate control in the building, and there is going to be work to do, but no other unmodeled causal factors influencing productivity are present. Please note that the user interface to the elicitation shows you this situation by fixing the values of *Hot Weather* to *True* (T), *Climate Control in the Building* to *False* (F), and *Work To Do* to *True* (T). We estimate this probability to be 10%.

E. We will now specify the strength of influences of the different causal factors of *High Productivity*.

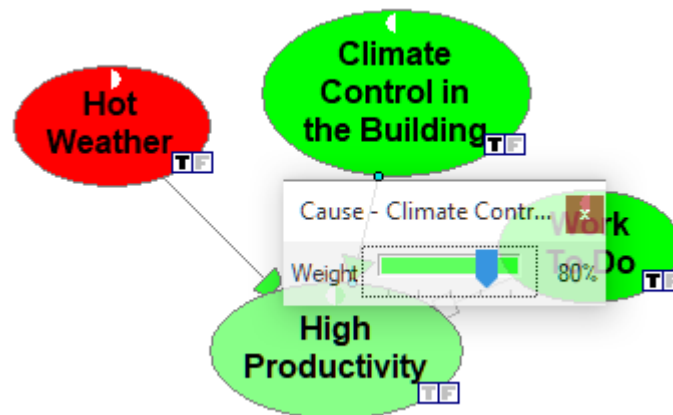
Interaction probabilities, also called *Weights*, generally describe the strengths of causes in their ability to influence the effect. The weight means something slightly different, depending on the type of interaction. Every variable in the model has a so-called "distinguished state," which is the state that exerts no influence on its children. For example, the distinguished state of *Hot Weather* is *False*, as hot weather will decrease productivity but its absence, i.e., mild weather, will have no influence on productivity. The distinguished state of *Climate Control in the Building* is *False*, as presence of climate control will influence productivity. The distinguished state of *Work to Do* is *True*, as having work to do is the normal situation that will not influence productivity. To learn more about the meaning of weights in each case, please look at the [The DeMorgan gate](#) section of this manual.

In order to change any of the interaction probabilities, just double-click on the corresponding arc.

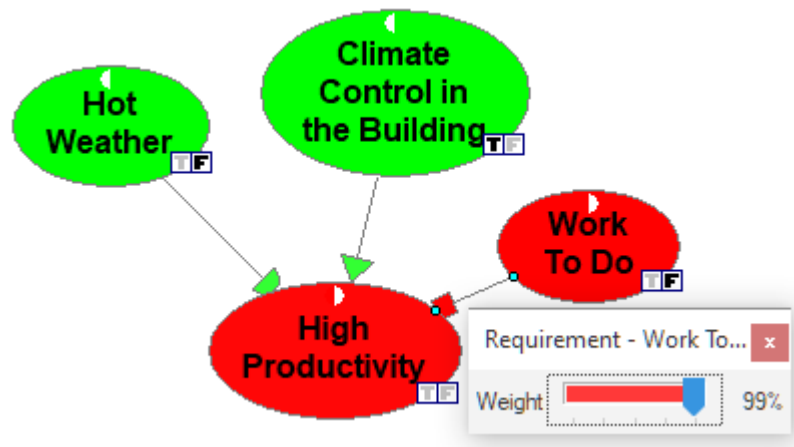


The weight of the influence can be adjusted by moving the slider. Weight of 80%, as pictured above, represents in case of a barrier, which is the relationship between *Hot Weather* and *High Productivity*, the probability that the productivity is going to be high if the weather is not hot, there is no climate control in the building and there is work to do. QGeNIe's user interface to the elicitation helps you with this by fixing the values of *Hot Weather* to *False* (F), *Climate Control in the Building* to *False* (F), and *Work To Do* to *True* (T). Please note that only the variable *Hot Weather* is not in its distinguished state when estimating this probability.

The *Weight* of the relationship between *Climate Control in the Building* and *High Productivity* of 80% (see the image below) means that there is 80% chance that the productivity is going to be high if the weather is hot, the climate control is present in the building, and there is work to do. QGeNIe's user interface to the elicitation helps you with this by fixing the values of *Hot Weather* to *True* (T), *Climate Control in the Building* to *True* (T), and *Work To Do* to *True* (T). Please note that only the variable *Climate Control in the Building* is not in its distinguished state when estimating this probability.



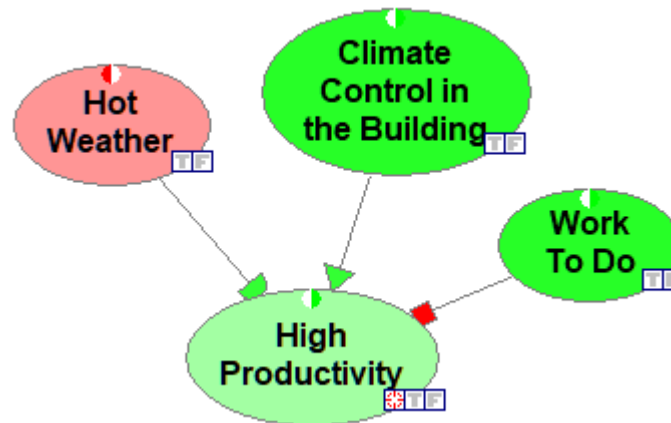
The *Weight* of the relationship between *Work To Do* and *High Productivity* of 99% (see the image below) means that there is 99% chance that the productivity is going to be low if there is no work to do, the weather is not hot and there is climate control in the building. QGeNIe's user interface to the elicitation helps you with this by fixing the values of *Hot Weather* to *True* (T), *Climate Control in the Building* to *True* (T), and *Work To Do* to *False* (F). In this case, none of the variables are in their distinguished states, as the relationship of a requirement with causes and barriers follows the AND function (more about this in the [The DeMorgan gate](#) section of this manual).



There is one more setting that we will perform - we will designate one of the nodes, *High Productivity*, as the focus of our analysis. Usefulness of this setting will become clear later in this section. To designate the *High Productivity* node as the focus of inference, please right-click on it and select *Set Focus*.



Let us pause for a moment and examine the graphical structure of our completed model:




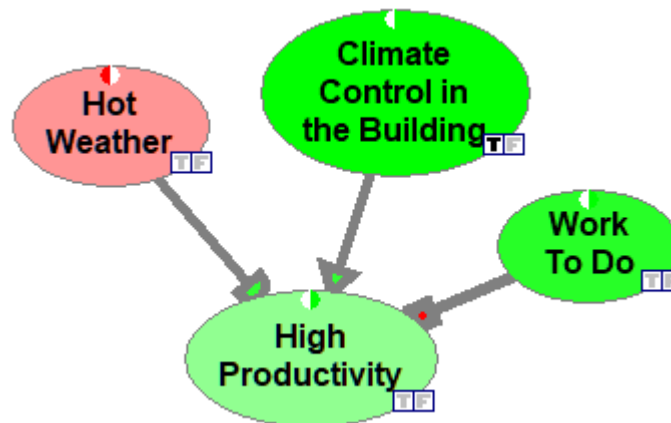
The structure shows causal interactions among the variables. We can see that hot weather, climate control, and availability of work to do all impact productivity. The influence of individual variables differs qualitatively. This is pictured by different arrowheads. Climate control impacts productivity positively, hot weather negatively, and work to do is a necessary condition for productivity to be high.

Colors of nodes reflect probabilities of the propositions that they represent. Generally, green color denotes probabilities that are within an acceptable range, red color denotes probabilities that are worrisome. Small circles on the top of each of the node help in judging the distance of the node color from the extremes (colors representing probabilities zero and one). Please note that one half of each circle is white and the other shows the color of the extreme. And so, *Hot Weather* node's probability is clearly on the side of red, between 0.5 (white color) and dark red (probability 1.0) shown by the other half of the small circle. The probability of *Climate Control in the Building* is quite close to 1.0 - please note that there is little difference between the node's color and the green half of the small circle.

There is one node (*High Productivity*) that is special and designated as the focus of reasoning (we will return to this in Section H). Each node has two small icons in its lower-right corner. These icons allow for setting an observation of the node and also show the observed value.

The model structure can be analyzed additionally by displaying the links between nodes (the arcs) in variable width, corresponding to the magnitude of their influence. To enable variable arc widths, press

the *Enable variable arc widths* () tool. This results in the following view:

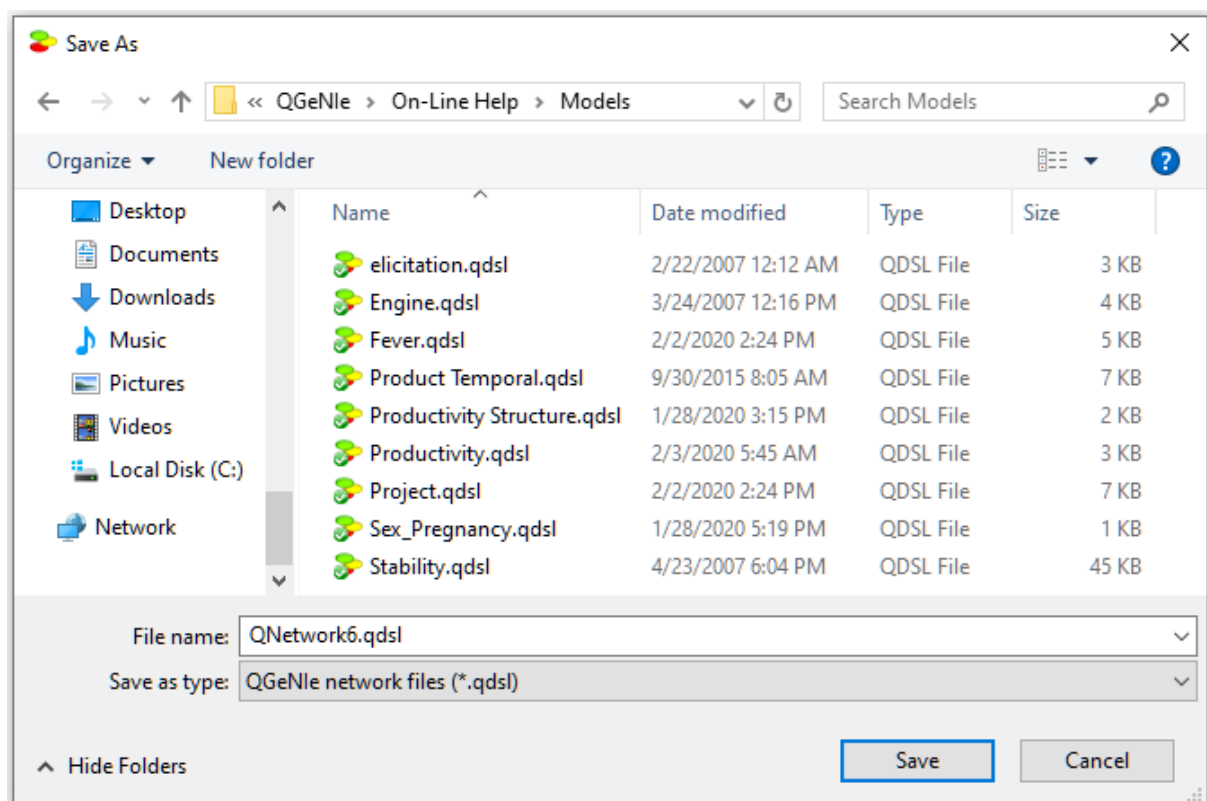


Please note that the thickness of each arc represents the strength with which the parent node influences the child node. It is useful for analyzing the critical paths of information flow.

F. At this point you should save your work.

1. Click on *Save* button (📁) on the [Standard Toolbar](#).

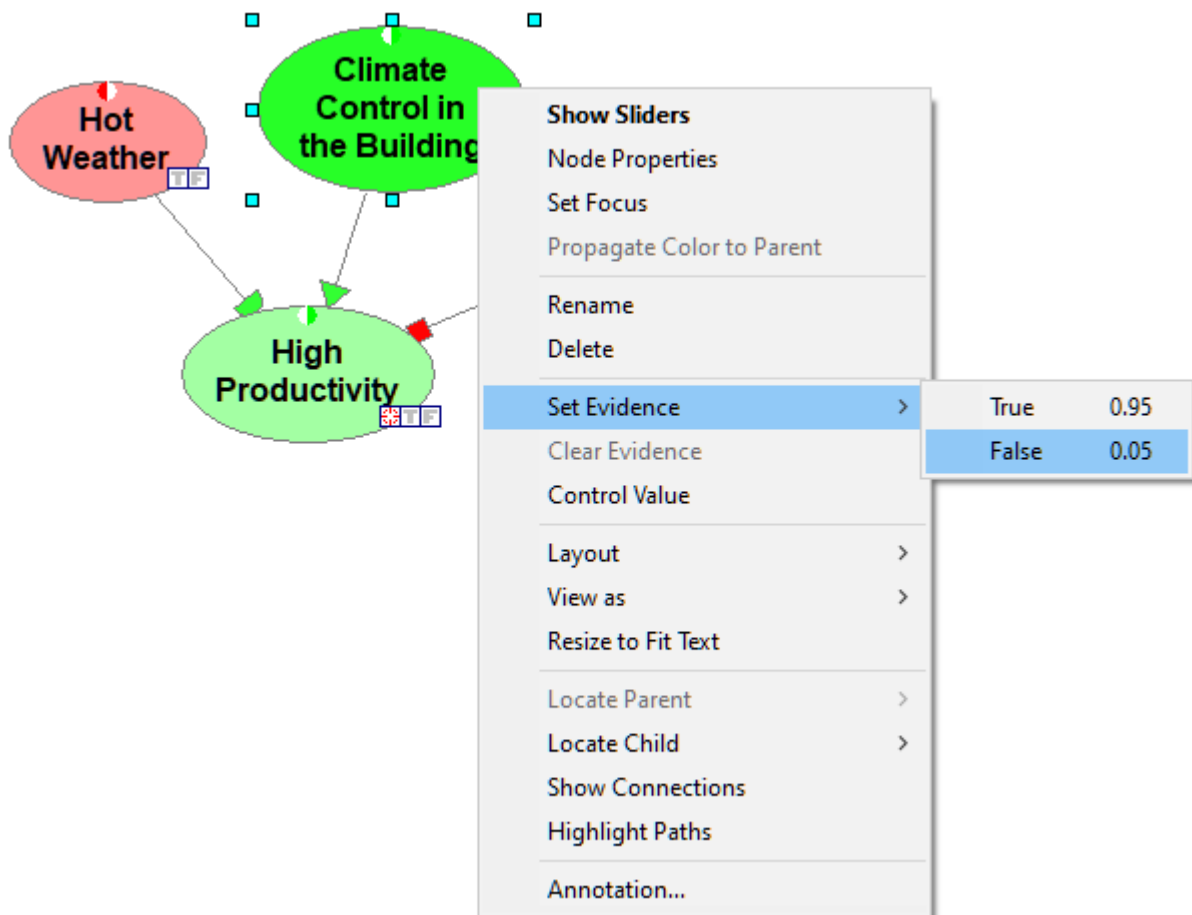
GeNIe will display the *Save As..* dialog shown below:



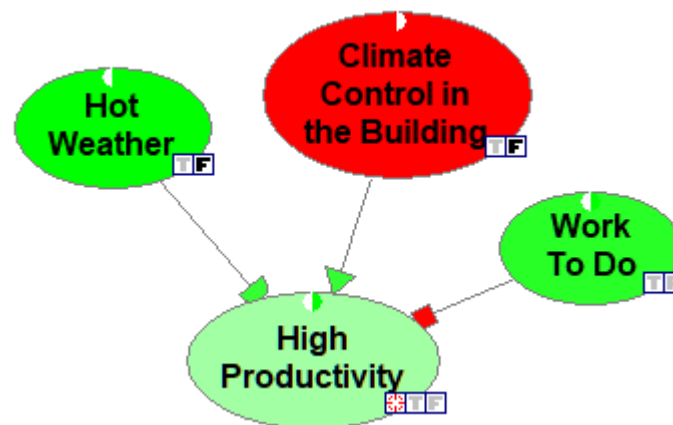
2. Enter *Productivity* as the *File name* and click on *Save*.

G. Now let us put our model to work and answer the question posed in the beginning of this tutorial.

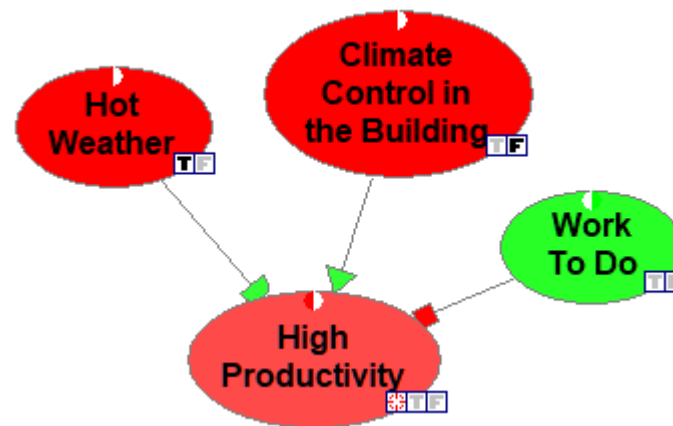
To answer the question "What is the expected impact on productivity of failure of the climate control in the building on a hot day and on a normal day respectively?", you will need to tell QGeNIe that you have observed the *Climate Control in the Building* to be *False*. There are two ways of achieving this. The first, slightly more cumbersome, is to right-click on the node *Climate Control in the Building* and choose *Set Evidence/False* from the context menu that will pop up.



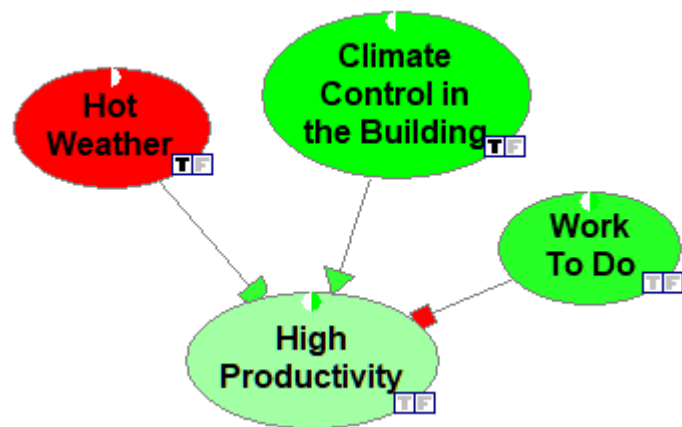
There are two shortcuts for observing a node's value: (1) through double-clicking on one of the small square icons (📏) in the lower-right corner of each node, or (2) through pressing the letter *T* or *F* (for *True* and *False* respectively). In this case, we double-click on the icon *False* (📏) of the *Hot Weather* node, which makes the icon bold or press *F* once the node has been selected. Please note that every time that we make an observation of a node value, the colors of model nodes change. Colors represent probabilities of the selected states and are meant to draw your attention. We observe the impact of the two observation made (please note the bold observation icons in the picture below) on the node *High Productivity*.



The colors indicate that a failing climate control will not have much of an impact when the day is not hot. Please note the general idea of showing colors in QGeNIe. When looking at the model, we can see that there is one variable that is a potential problem -- *Climate Control in the Building*, as its color is deep red. The other nodes and our focus variable, *High Productivity* in particular, seem to be fine and show as green. Let us change the observation for the node *Hot Weather* to *True* by double-clicking on the corresponding observation icon T. We can see in the image below that there are now three nodes that are in red: *Climate Control in the Building*, *Hot Weather* and *High Productivity*. Hot weather in combination with failing climate control in the building cause problems with productivity.



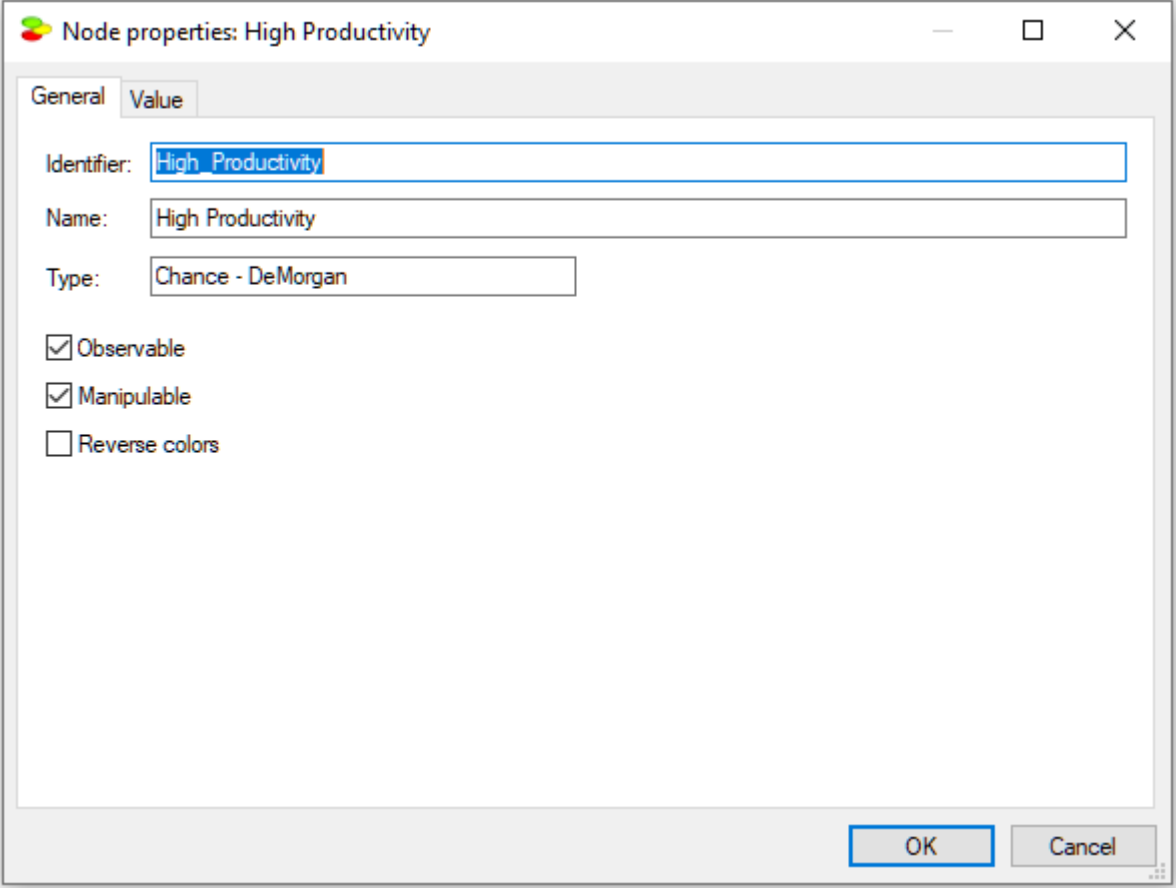
Will fixing the climate control help? We can check this by setting the *Climate Control in the Building* to be *True* by double-clicking on the corresponding observation icon and observe that the node *High Productivity* turns somewhat green again.



It is possible to see the exact numerical posterior probabilities of any of the propositions. For that, please right-click on the node in question and select *Node Properties*:



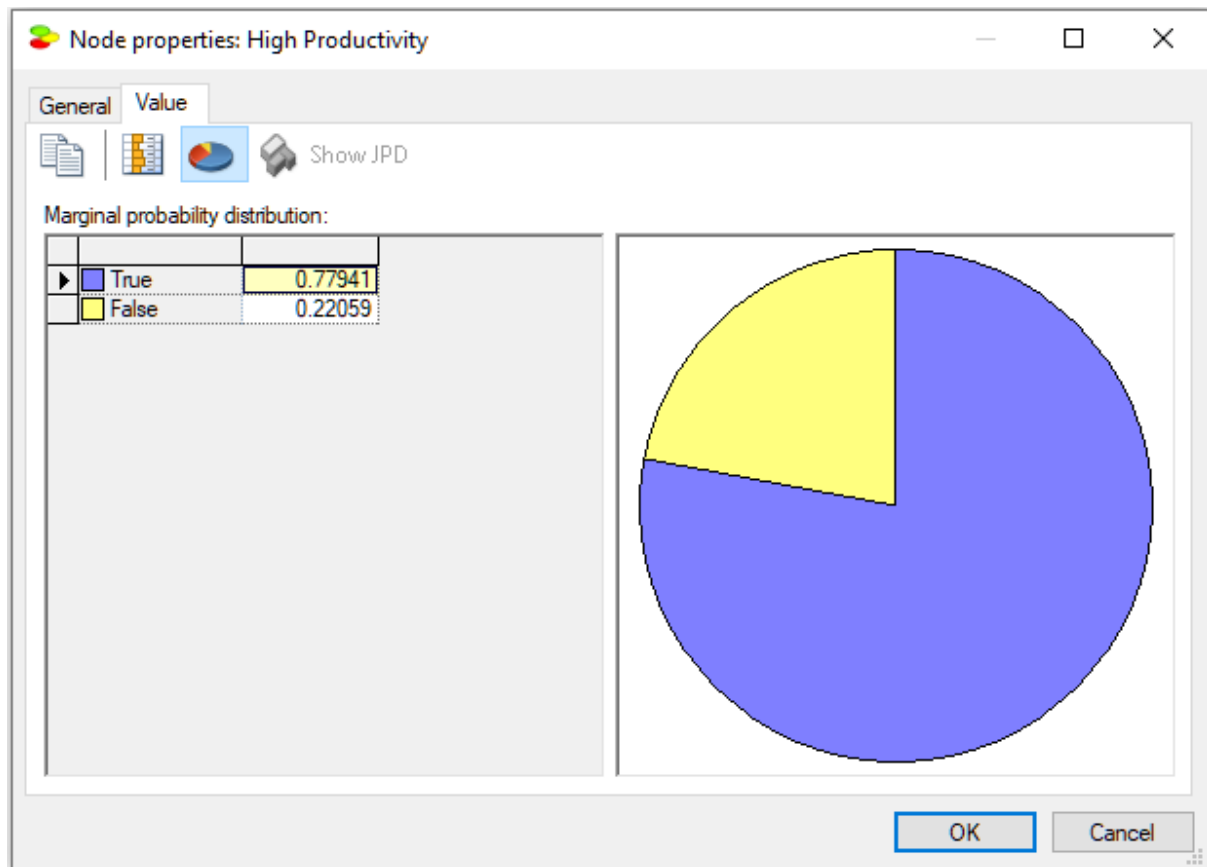
Please choose the *Value* tab in the property sheets,



The image shows a 'Node properties: High Productivity' dialog box. It has two tabs: 'General' and 'Value'. The 'General' tab is active. It contains three text input fields: 'Identifier' with the value 'High_Productivity', 'Name' with the value 'High Productivity', and 'Type' with the value 'Chance - DeMorgan'. Below these fields are three checkboxes: 'Observable' (checked), 'Manipulable' (checked), and 'Reverse colors' (unchecked). At the bottom right are 'OK' and 'Cancel' buttons.

Property	Value
Identifier	High_Productivity
Name	High Productivity
Type	Chance - DeMorgan
Observable	<input checked="" type="checkbox"/>
Manipulable	<input checked="" type="checkbox"/>
Reverse colors	<input type="checkbox"/>

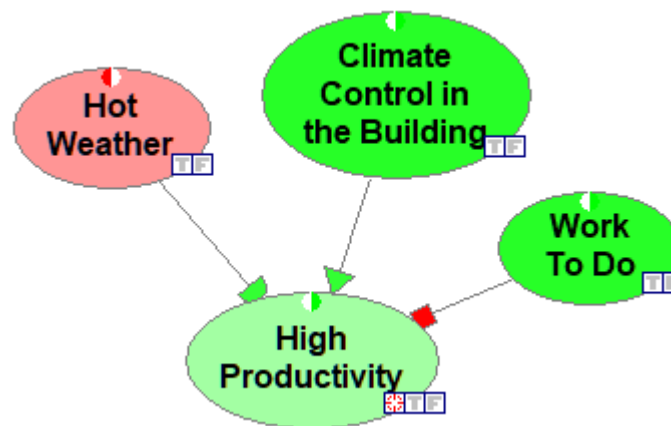
which will look as follows:



We can see that the numerical posterior probability of *High Productivity* being *True* is equal to 0.77941.

H. Value of observation and value of manipulation.

QGeNIe allows for calculating the value of observation and the value of manipulation based on a measure known as cross-entropy. Cross-entropy, or expected change in entropy, is a unit-less measure of the expected change in entropy of a focus variable in the model. In order to calculate the value of information, the model has to contain exactly one focus variable. We have set the focus variable to be *High Productivity* in Section E above. Please note that the node is marked by a target icon (🎯).



Every node in the model can be marked as *Observable* and *Manipulable* by checking a corresponding check box in the node's property sheet.

Node properties: High Productivity

General Value

Identifier:

Name:

Type:

☒ Observable


☒ Manipulable

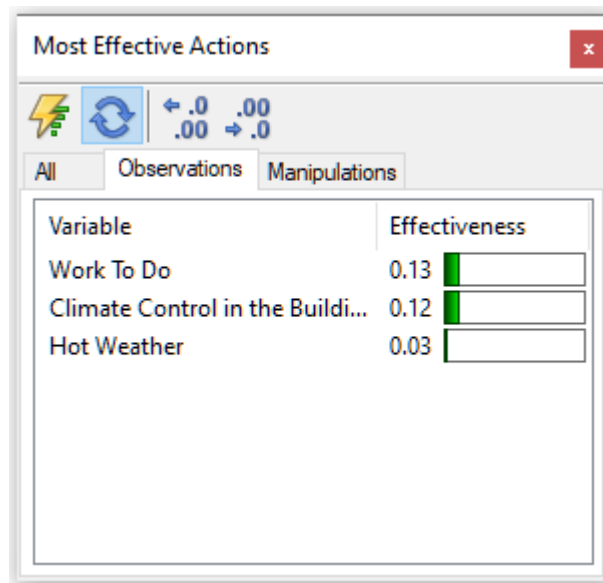
☐ Reverse colors

OK Cancel

Setting a node as *Observable* means that it is possible to observe its value (*True* or *False*), setting a node as *manipulable* means that it is possible to impose a value on it by means of an external intervention (this value is also *True* or *False*). We can, thus, observe the state of the climate control system or we can set its state (by turning it on or off, breaking it, or fixing it when broken). The

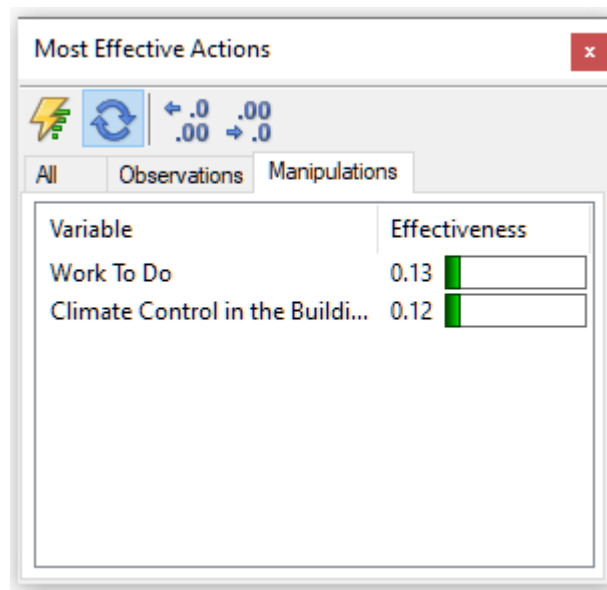
difference between observation and manipulation is subtle but quite clear. It is possible, for example, to observe whether a day is hot but it may be impossible to make it hot. In case of climate control system, we can both observe its value and manipulate it. Once the model nodes have been marked as *Observable* or *Manipulable*, we can let QGeNIe calculate the value of observing any of the observable nodes and the value of manipulating any of the manipulable nodes from the point of view of learning more about the *Focus* variable (in this case, variable *High Productivity*). To invoke this calculation,

please press the *Most Effective Actions* window icon () . This will open the following window pane.



The *Observations* tab shows the list of all observable variables rank-ordered from the most informative to the least informative in learning the probability of the focus variable (*High Productivity* in our case). The numerical values are unit-less and express the expected change in entropy of the focus variable. It looks like observing whether there is enough work to do is the most informative for learning about high productivity.

Clicking on the *Manipulations* tab shows the list of all manipulable variables rank-ordered from the most effective to the least effective in changing the probability of the focus variable:



Please note that the list does not contain the variable *Hot Weather*, as it was not marked as *Manipulable*. It looks like making sure that there is enough work to do is the most effective way of increasing productivity.

What we have created is a simple qualitative causal model. You can create more complex models in a similar way.

You can find the above model named *Productivity.qdsl* in the *Qualitative Models* directory among other example models that come with QGeNIe or download it directly from our model repository.

This page is intentionally left blank.
Remove this text from the manual
template if you want it completely blank.

Introduction

3 Introduction

3.1 Guide to QGeNIe manual

The best user interface to a computer program is one for which there is no need for a manual. This has been from the very start our design motto and, from the compliments of our users, we learn that we have (almost) reached our objective. Most of the tasks in QGeNIe are intuitive and conform to the general standards of well designed user interfaces. If you do not know how to do something, just try what you would find most intuitive and what you would do in a well designed modern user interface. There is a good chance that this will work for you.

This manual may turn out to be superfluous for many users. Still, there are many ways in which people learn and some users will find it useful. Some may find it handy to use the search functionality in case they encounter a problem. Others, familiar with the methodology, may want to read through the section Using QGeNIe, which describes various QGeNIe modules. Yet others, may want to treat this manual as a simple handbook for decision-theoretic modeling.

Decision-theoretic modeling is not as easy as writing a brief document, preparing a slide show, or drawing a picture. While there are many good books available that cover decision-theoretic modeling, they typically use theoretical concepts and draw models symbolically on paper. Building them in software is different. Besides, much of QGeNIe functionality is novel and not described or otherwise covered anywhere. This is the main reason why even QGeNIe needs an easily accessible manual, tutorial-style introductions, and on-line help.

QGeNIe manual, the introduction of which you are reading at the moment, is available for a variety of platforms: HTML, compiled HTML (CHM), PDF, etc., all available from <https://support.bayesfusion.com/docs/>.

3.2 QGeNIe

QGeNIe is an interactive development environment for rapid creation of qualitative causal models in uncertain domains. These models represent propositions by means of nodes in an acyclic directed graph. These nodes are always propositional and take two possible values: *True* and *False*. The colors of these nodes represent the degrees of truth of the propositions. Mathematically speaking, the colors represent the probability of the state *True* (or *False* - it is the users' choice). While QGeNIe users can define the color scale, the default is a range between red and green, representing undesirable and desirable states respectively. QGeNIe allows for an interactive exploration of the models, examining the effects of observations and manipulations of individual variables.

There are two major applications of QGeNIe:

1. A standalone system for rapid creation of causal models, useful in all kinds of strategic planning problems, where problems are complex enough to be a challenge for an unaided human mind and, at the same time, too complex to model by means of fully specified, precise quantitative models. For sufficiently complex problems, it is a challenge for an unaided human mind to predict effects of various actions. Typically, in addition to obvious consequences of a decision, there will be indirect pathways through which actions may propagate through the system and lead to surprising effects. QGeNIe is a tool for capturing the knowledge and intuitions of decision makers and focusing group discussion on calculating the global effects of various decision options. QGeNIe offers what can be called an instant gratification system in the sense of showing interactively the effects of observations and manipulations.

2. A system for generating simple first-cut version of quantitative probabilistic models. Models developed by means of QGeNIe can be exported to GeNIe and refined into fully quantitative models.

An early version of QGeNIe was created and developed at the Decision Systems Laboratory, University of Pittsburgh between 1995 and 2015. In 2015, we created a company, BayesFusion, LLC, and acquired a license for QGeNIe from the University of Pittsburgh. Continuing the tradition of the Decision Systems Laboratory, we are making it available free of charge to the academic community for research and teaching use in order to promote decision-theoretic methods in decision support systems. QGeNIe has been tested extensively in many teaching, research, and commercial environments. We are continuously improving it and are interested in user comments. We encourage the users of QGeNIe to let us know about encountered problems and possible suggestions.

GeNIe's name and its uncommon capitalization originates from the name Graphical Network Interface, given to the original simple graphical user interface to [SMILE](#), our library of classes for graphical probabilistic and decision-theoretic models. Its successor was GeNIe, SMILE's GUI. QGeNIe is a simplified, qualitative graphical user interface to SMILE.

QGeNIe allows for building models of any size and complexity, limited only by the capacity of the operating memory of your computer. QGeNIe is a modeling environment. Models developed using QGeNIe can be exported to GeNIe or embedded into any applications and run on any computing platform, using SMILE, which is fully portable.

QGeNIe, GeNIe and SMILE have been originally developed to be major teaching and research tools in academic environments and have been used at hundreds if not thousands of universities world-wide. Most research conducted at the Decision Systems Laboratory, University of Pittsburgh, found its way into both programs. Because of their versatility and reliability, QGeNIe, GeNIe and SMILE have become incredibly popular and became *de facto* standards in academia, while being embraced by many government, military, and commercial users.

The strongest element of GeNIe (and indirectly of QGeNIe), one that distinguishes it from a large number of other graphical modeling tools, is its user interface. We have paid a lot of attention to it and it shows. While developing decision-theoretic models takes typically an enormous amount of time, GeNIe cuts the effort by orders of magnitude and it will lead to a fast return of the investment in its licensing fees. SMILE is not far behind and belongs to the easiest to learn and use, most reliable, and fastest libraries for graphical models.

3.3 SMILE Engine

SMILE (Structural Modeling, Inference, and Learning Engine) is a fully platform independent library of functions implementing graphical probabilistic and decision-theoretic models, such as [Bayesian networks](#), influence diagrams, and structural equation models. Its individual functions, defined in SMILE Applications Programmer Interface (API), allow to create, edit, save, and load graphical models, and use them for probabilistic reasoning and decision making under uncertainty.

SMILE is implemented in C++ in a platform independent fashion. We also provide Java (jSMILE), .NET (SMILE.NET), .COM (SMILE.COM), Python (PySMILE) and R (rSMILE) wrappers for users who want to use SMILE with languages other than C++. Through the Java wrapper, SMILE can be used in programming environments such as Matlab or Ruby. Through the .NET wrapper, it can be

used, among others, from C# and VB.NET. SMILE is equipped with an outer shell, a developer's environment for building graphical decision models, known as GeNIe, or a qualitative graphical user interface, known as QGeNIe. GeNIe and QGeNIe are platform dependent and run only on Windows computers, although our users have successfully run them under MacOS and Linux operating systems. SMILE can be embedded in programs that use graphical probabilistic models as their reasoning engines. Such programs can be distributed to end users or placed on servers for cloud use. Models developed in SMILE can be equipped with a user interface that suits the user of the resulting application most.

QGeNIe, GeNIe and SMILE have been originally developed to be major teaching and research tools in academic environments and have been used at hundreds of universities world-wide. Most research conducted at the Decision Systems Laboratory, University of Pittsburgh, found its way into QGeNIe, GeNIe and SMILE. Because of their versatility and reliability, QGeNIe, GeNIe and SMILE have become incredibly popular and became *de facto* standards in academia, while being embraced by many government, military and commercial users.

The strongest element of SMILE, one that distinguishes it from a large number of other graphical modeling tools, is its ease of use from a programmer's perspective (it offers a modern object-oriented API), availability for multiple platforms, its reliability (it has been tested heavily in practical research and commercial applications since 1998), and speed (it has done very nicely in speed competitions organized by UAI, the annual Conference on Uncertainty in Artificial Intelligence). Speed especially is crucial, as most calculations in probabilistic graphical models are exponential in nature.

3.4 BayesBox

BayesBox is a member of BayesFusion's family of products developed for modeling, learning, and decision support. It facilitates unlimited online sharing of probabilistic graphical models (such as Bayesian networks, qualitative Bayesian networks, influence diagrams, dynamic Bayesian networks, and hybrid Bayesian networks) created for internal or public use. Customers can upload to the repository any number of models organized into any number of categories. This repository can be accessed by an unlimited number of users through web browsers.

Server part of BayesBox runs on Linux or Windows and is available as a software installable and fully controlled by customers on-premises, or hosted as a service by BayesFusion. Users access BayesBox models through any standards-compliant web browser, can explore the model structure, enter evidence, and examine results.

Bayesian inference in the BayesBox server is performed by SMILE Engine, our cross-platform Bayesian software library.

BayesBox can be fine-tuned to user needs and reflect customer brand, including a name, a logo, and a color scheme. Optionally, administrator interface allows for enabling access control through a login page.

GeNIe allows for opening models from an existing BayesBox instance, which makes BayesBox a useful tool for teams working on decision-theoretic methodologies.

BayesFusion's public model repository is powered by BayesBox (see <https://repo.bayesfusion.com/>). For demonstration purposes, we have also created a BayesBox-based web site of a fictitious company Evidentious, Inc., at <https://demo.bayesfusion.com/>.

3.5 BayesMobile

BayesMobile is a member of BayesFusion's family of products developed for modeling, learning, and decision support. It runs on Apple mobile devices, such as iPhone or iPad and allows for working with diagnostic models. We have created it for the convenience of those users who want to quickly field their diagnostic models. More information about BayesMobile can be found on our web page.

3.6 Distribution information

Hardware and software requirements

Disk space

Full installation of QGeNIe requires less than 20 MB of disk space.

Memory

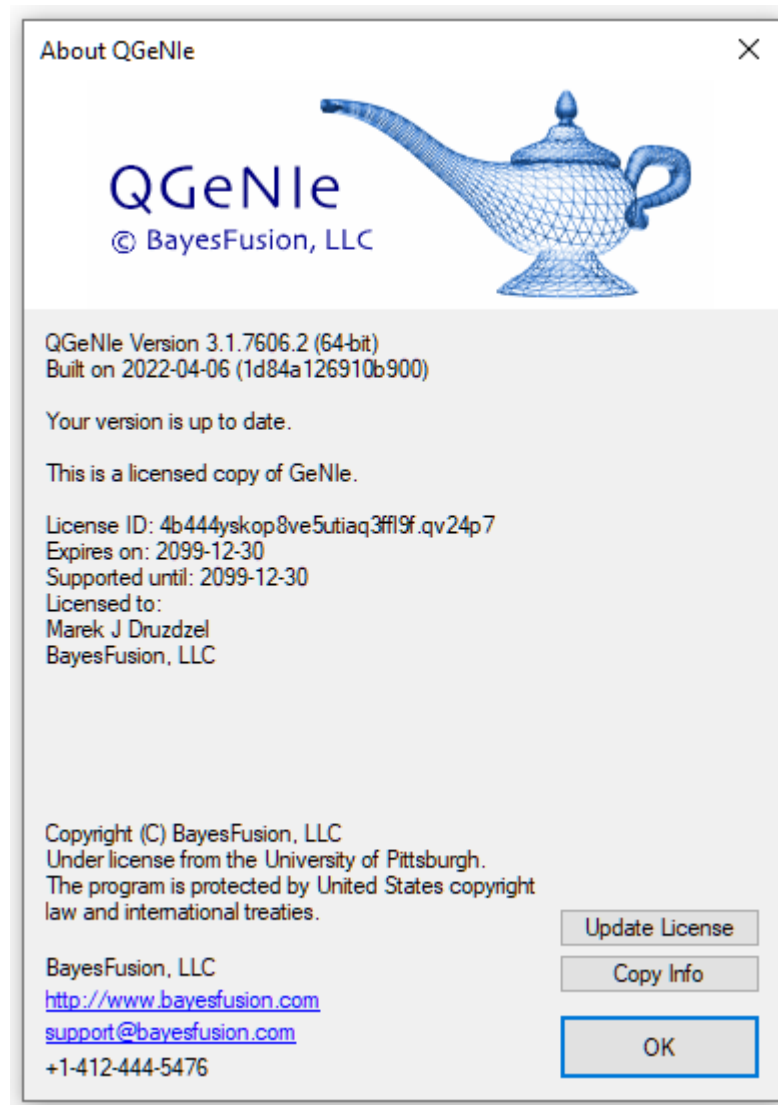
QGeNIe has practically no minimum memory requirements and can run under a minimum Windows configuration. The actual memory requirement will depend on the size and complexity of the models that you create. Too little memory may result in decreased performance. In general, conditional probability tables grow exponentially with the number of parents of a node. The maximum number of parents of a node will, therefore, determine memory requirements. In addition, memory requirements of the clustering algorithm grow with the connectivity of the network.

Operating system

QGeNIe is written for the Windows operating systems. Installation of QGeNIe under Windows operating systems may require administrator privileges. While we cannot guarantee 100% compatibility, we are verifying with each build that it runs on macOS (formerly OS X) and Linux under Wine. Please see [QGeNIe on a Mac](#) section for more information on running QGeNIe on a macOS. Getting QGeNIe to run under Linux is virtually identical.

QGeNIe version

To determine the version of QGeNIe that you have installed, select *About QGeNIe* from the [Help Menu](#). The version number is listed in the small frame of the following window:



Example Networks

QGeNIe installation creates a directory named *Qualitative Models*, containing the following models (all of these models are available for download from [BayesFusion's interactive model repository](#) through [File](#) menu):

Engine.xdsl

The *Engine* models several causes of a possible failure of starting a car engine and illustrates the interaction of *Causes* and *Barriers* in the DeMorgan gate.

Fever.xdsl

The *Fever* model illustrates the interaction of *Causes*, *Barriers*, *Requirements* and *Inhibitors* in the DeMorgan gate.

Fire.xdsl

The *Fire* model has been developed for the purpose of explaining the four types of causal interactions defined in the DeMorgan gate that is at the foundations of QGeNIe models. The model has first appeared in (Druzdzal 2010).

Pregnancy.xdsl

The *Pregnancy* model illustrates working of a *Requirement* in the DeMorgan gate.

Product Temporal.xdsl

The *Product Temporal* model is a qualitative dynamic Bayesian network (DBN), illustrating higher order temporal influences.

Productivity.xdsl

The *Productivity* model is a simple qualitative network used in the [Hello QGeNIe section](#) of this manual and it models various causal factors in productivity.

Project.xdsl

The *Project* model illustrates a causal graph for group brainstorming about various project obstacles and results from the point of view of an airplane manufacturer.

Stability.xdsl

The *Stability* model has been developed by Bradd C. Hayes & Jeffrey I. Sands and published in: Bradd C. Hayes & Jeffrey I. Sands, "Doing Windows: Non Traditional Military Responses to Complex Emergencies." DSD Research Report 97-1, Decision Support Department, Center for Naval Warfare Studies, U.S. Naval War College. Figure 5-1, page 101.

3.7 QGeNIe on a Mac

Running QGeNIe on macOS (formerly Mac OS X) with Wine.

1. Download and install Wine on your Mac. Follow the instructions at Wine website (<https://wiki.winehq.org/MacOSX>).
2. Download QGeNIe on your Mac. Double-click on QGeNIe Installer icon to start the setup wizard.
3. After installing QGeNIe, run Wine from the Launchpad. This opens the *Terminal* window configured to run Windows programs. Do not try to launch QGeNIe by locating its icon and double-clicking on it.
4. In Wine's terminal, use the `cd` command to navigate to QGeNIe's installation directory. Assuming that QGeNIe was installed in its default location, the command is:

```
cd ".wine/drive_c/Program Files/QGeNIe 4.1"
```

Note that quotes are required, because some of the directory names may contain spaces.

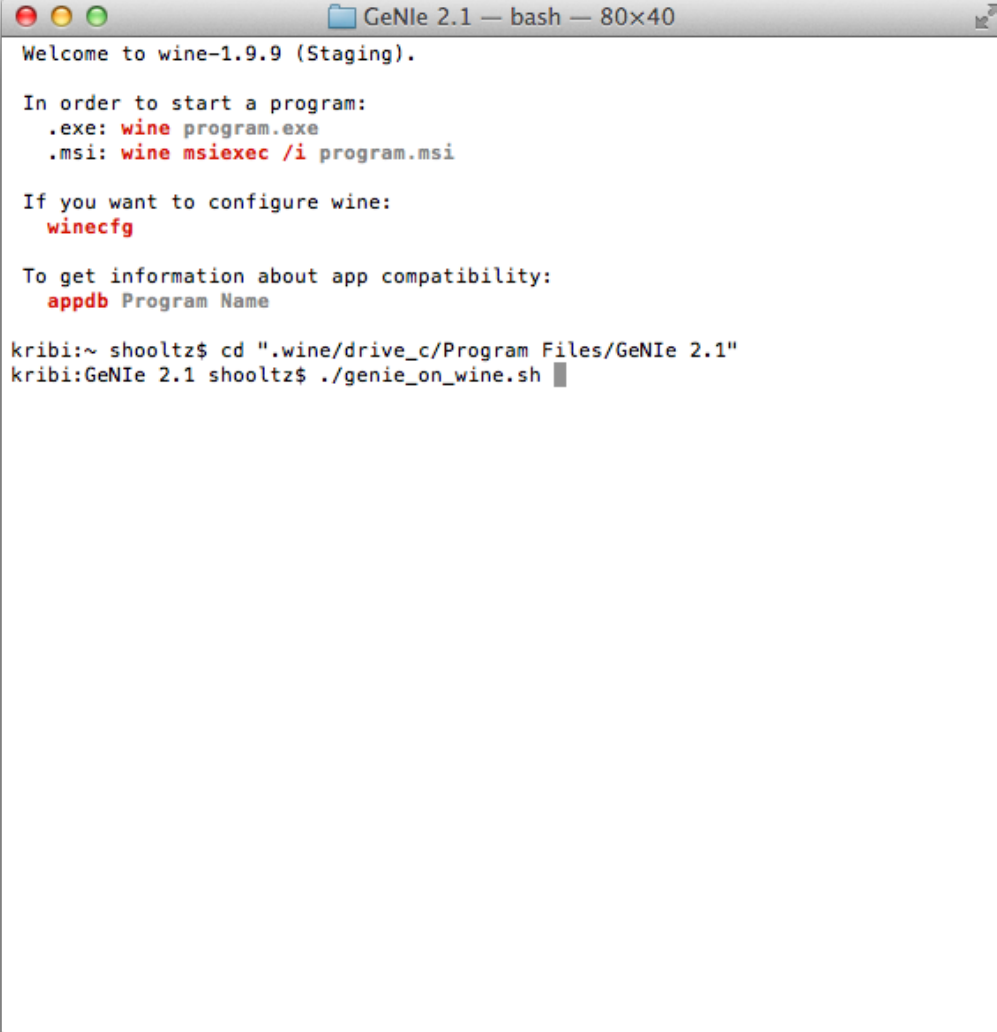
QGeNIe Academic installs to "GeNIe 4.1 Academic" by default.

5. After changing the directory, run the qgenie_on_wine.sh script:

```
./qgenie_on_wine.sh
```

This starts QGeNIe using an appropriate Wine configuration.

See the screen shot below, illustrating steps 3-5.



```
GeNIe 2.1 — bash — 80x40
Welcome to wine-1.9.9 (Staging).

In order to start a program:
.exe: wine program.exe
.msi: wine msiexec /i program.msi

If you want to configure wine:
winecfg

To get information about app compatibility:
appdb Program Name

kribi:~ shooltz$ cd ".wine/drive_c/Program Files/GeNIe 2.1"
kribi:GeNIe 2.1 shooltz$ ./genie_on_wine.sh
```

3.8 QGeNIe on Linux

GeNIe can be used on a Linux system with VM-based solution like VirtualBox, or with Wine (a compatibility layer capable of running Windows applications). See the [QGeNIe on a Mac](#) section for more info about QGeNIe and Wine.

3.9 Non-Latin Alphabets in QGeNIe

QGeNIe accommodates non-Latin alphabets - any text within a model can be written using letters outside of the Latin alphabet. This includes variables IDs, names, comments, etc.

3.10 Copyright notice

Copyright (C) [BayesFusion, LLC](#), under license from the [University of Pittsburgh](#). All rights reserved. No part of this manual may be reproduced or transmitted in any form or by any means, electronic or mechanical, without an explicit written permission of BayesFusion, LLC.

We would like to acknowledge the following trademarks:

macOS and OS X are registered trademarks of Apple, Inc.

QGeNIe, GeNIe, [SMILE](#), BayesBox and BayesMobile and all accompanying graphics and manuals are copyrighted (1996-2022) by University of Pittsburgh, used under license by BayesFusion, LLC, and cannot be copied or distributed without permission. The only legal way of obtaining the programs is directly from BayesFusion, LLC. We require that interested individuals contact us directly or visit our web site for the most recent copy of the programs. This ensures the quality and completeness of the programs and accompanying manuals. It also allows us to keep track of who is using the programs and to notify the users about updates and new releases.

Academic use of QGeNIe, GeNIe and SMILE

We support teachers using QGeNIe in their classes and maintain shareware resources, such as network repositories, that are useful in teaching. (Please, visit [BayesFusion, LLC](#) web site for more information.) QGeNIe, GeNIe and SMILE are also useful in academic research projects. Our software is free for academic teaching and research use. In return for this, to get credit for our work, we ask that all publications of research or applications in which QGeNIe, GeNIe or SMILE were used contain an explicit acknowledgment to that effect. Examples of simple acknowledgments are listed below:

The models described in this paper were created using QGeNIe, available free of charge for academic research and teaching use from BayesFusion, LLC, <https://www.bayesfusion.com/>.

The core of our implementation is based on the SMILE reasoning engine for graphical probabilistic models, available free of charge for academic research and teaching use from BayesFusion, LLC, <https://www.bayesfusion.com/>.

3.11 Disclaimer

QGeNIe, GeNIe and [SMILE](#) are made available on an "as is" basis. We have performed extensive tests of the software, which has been used in hundreds of research, teaching, and commercial projects, but we are not providing any guarantees as to its correct working and take no responsibility for effects of possible errors. We do appreciate suggestions and bug reports and will do the best within our capabilities to correct errors and accommodate users' needs in our future development plans. If you have suggestions or have discovered a bug in the program, please send us electronic mail at support@bayesfusion.com.

Similarly, while we have taken much care in writing this manual and making it as accurate as possible, we assume no responsibility for possible errors that it may contain. We encourage the readers to send us their corrections and suggestions at support@bayesfusion.com.

3.12 Acknowledgments

Support for the development QGeNIe, GeNIe and [SMILE](#) at the University of Pittsburgh was provided in part by the Air Force Office of Scientific Research under grants F49620-97-1-0225 and F49620-00-1-0122, by the National Science Foundation under Faculty Early Career Development (CAREER) Program, grant IRI-9624629, by Hughes Raytheon Laboratories, Malibu, California, by ARPA's Computer Aided Education and Training Initiative under grant N66001-95-C-8367, and by the University of Pittsburgh Central Development Fund.

While little of the original code has remained and most of the programs have been rewritten with time, the principal developers of QGeNIe, GeNIe and SMILE (listed alphabetically) included:

Saeed Amizadeh, Steve Birnie, Jeroen J.J. Bogers, Girish Chavan, Hanyang Chen, Jian Cheng, Denver H. Dash, Martijn de Jongh, Marek J. Druzdzal, Daniel Garcia Sanchez, Nancy Jackson, Randy Jagt, Joost Koiter, Marcin Kozniowski, Hans van Leijen, Yan Lin, Tsai-Ching Lu, Paul Maaskant, Agnieszka Onisko, Hans Ove Ringstad, Tomek Sowinski, Carl P.R. Thijssen, Miguel Tjon Kon Fat, Daniel Tomalesky, Mark Voortman, Changhe Yuan, Haiqin Wang and Adam Zagorecki.

We would like to acknowledge contributions of the following individuals (listed alphabetically) to coding, documentation, graphics, web site, and testing of SMILE and GeNIe: Kimberly Batch, Avneet S. Chatha, Cristina Conati, Roger Flynn, Abigail Gertner, Charles E. Grindle, Christopher Hall, Christopher A. Geary, William Hogan, Susan E. Holden, Margaret (Peggie) Hopkins, Jun Hu, Kent Ma, Robert (Chas) Murray, Zhendong Niu, Shih-Chueh (Sejo) Pan, Bharti Rai, Michael S. Rissman, Luiz E. Sant'Anna, Jeromy A. Smith, Jiwu Tao, Kurt VanLehn, Martin van Velsen, Anders Weinstein, David Weitz, Zaijiang Yuan, Jie Xu and many others.

Students in the courses *Decision Analysis and Decision Support Systems* at the University of Pittsburgh, *Decision Support Systems for Public Managers* at Carnegie Mellon University and *Decision Support and Expert Systems* at the University of Alaska, Anchorage provided us with useful feedback and suggestions.

QGeNIe embeds a number of good ideas that we have gratefully assimilated over time from other software, whether decision-theoretic or not. The great user interface of [Analytica](#) has been an inspiration and a role model for us. Analytica's user interface has been developed by Max Henrion and Brian Arnold at [Carnegie Mellon University](#) in late 1980s and early 1990s. Our treatment of submodels is very similar to that in Analytica.

This page is intentionally left blank.
Remove this text from the manual
template if you want it completely blank.

Decision-theoretic modeling

4 Decision-theoretic modeling

4.1 Decision analysis

Decision analysis is the art and practice of decision theory, an axiomatic theory prescribing how decisions should be made. Decision analysis is based on the premise that humans are reasonably capable of framing a decision problem, listing possible decision options, determining relevant factors, and quantifying uncertainty and preferences, but are rather weak in combining this information into a rational decision.

Decision analysis comes with a set of empirically tested tools for framing decisions, structuring decision problems, quantifying uncertainty and preferences, discovering those factors in a decision model that are critical for the decision, and computing the value of information that reduces uncertainty. Probability theory and decision theory supply tools for combining observations and optimizing decisions. While QGeNIe is under continuous development, it already implements a large set of these tools.

While decision analysis is based on two quantitative theories, probability theory and decision theory, its foundations are qualitative and based on axioms of rational choice. The purpose of decision analysis is to gain insight into a decision and not to obtain a recommendation. The users of QGeNIe will notice that this important premise is reflected in its functionality and, most importantly, its user interface.

4.2 Probability

Decision theoretic and decision analytic methods quantify uncertainty by probability. It is quite important for a decision modeler to understand the meaning of probability. There are three fundamental interpretations of probability:

- **Frequentist interpretation**

Probability of an event in this view is defined as the limiting frequency of occurrence of this event in an infinite number of trials. For example, the probability of heads in a single coin toss is the proportion of heads in an infinite number of coin tosses.

- **Propensity interpretation**

Probability of an event in this view is determined by physical, objective properties of the object or the process generating the event. For example, the probability of heads in a single coin toss is determined by the physical properties of the coin, such as its flat symmetric shape and its two sides.

- **Subjectivist interpretation**

The frequentist and the propensity views of probability are known as objectivist as they assume that the probability is an objective property of the physical world. In the subjectivist, also known as Bayesian interpretation, probability of an event is subjective to personal measure of the belief in that event occurring.

While the above three interpretations of probability are theoretical and subject to discussions and controversies in the domain of philosophy, they have serious implications on the practice of decision analysis. The first two views, known collectively as objectivist, are impractical for most real world decision problems. In the frequentist view, in order for a probability to be a meaningful measure of

uncertainty, it is necessary that we deal with a process that is or at least can be imagined as repetitive in nature. While coin tosses provide such a process, uncertainty related to nuclear war is a rather hard case - there have been no nuclear wars in the past and even their repetition is rather hard to imagine. Obviously, for a sufficiently complex process, such as circumstances leading to a nuclear war, it is not easy to make an argument based on physical considerations. The subjectivist view gives us a tool for dealing with such problems and is the view embraced by decision analysis.

The subjectivist view interprets probability as a measure of personal belief. It is legitimate in this view to believe that the probability of heads in a single coin toss is 0.3, just as it is legitimate to believe that it is 0.5 as long as one does not violate the axioms of probability, such as one stating that the sum of probabilities of an event and its complement is equal to 1.0. It is also legitimate to put a measure of uncertainty on the event of nuclear war. Furthermore, this measure, a personal belief in the event, can vary among various individuals. While this sounds perhaps like a little too much freedom, this view comes with a rule for updating probability in light of new observations, known as Bayes theorem. There exist *limits theorems* that prove that if Bayes theorem is used for updating the degree of belief, this degree of belief will converge to the limiting frequency regardless of the actual value of the initial degree of belief (as long as it is not extreme in the sense of being exactly, or infinitesimally close to, 0.0 or 1.0). While these theorems give guarantees in the infinity, a reasonable prior belief will lead to a much faster convergence.

The subjectivist view makes it natural to combine frequency data with expert judgment. Numerical probabilities can be extracted from databases, can be based on expert judgment, or a combination of both. Obtaining numbers for probabilistic and decision-theoretic models is not really difficult. The process of measuring the degree of belief is referred to as a probability assessment. Various decision-analytic methods are available for probability assessment.

4.3 Bayesian networks

Bayesian networks (also called *belief networks*, *Bayesian belief networks*, Bayes nets, *causal probabilistic networks*, or *causal networks*) (Pearl 1988) are acyclic directed graphs in which nodes represent random variables and arcs represent direct probabilistic dependences among them. The structure of a Bayesian network is a graphical, qualitative illustration of the interactions among the set of variables that it models. The structure of the directed graph can mimic the causal structure of the modeled domain, although this is not necessary. When the structure is causal, it gives a useful, modular insight into the interactions among the variables and allows for prediction of effects of external intervention.

The name Bayesian originates from the fact that the joint probability distribution represented by a Bayesian network is subjective (please recall that they are sometimes called *belief networks*; *Bayesian approach* is often used as a synonym for [subjective view on probability](#)) and this subjective probability distribution can be updated in the light of new evidence using Bayes theorem.

The purely theoretical view that the structure of a Bayesian network represents independences and that lack of an arc between any two variables X and Y represents a (possibly conditional) independence between them, is not intuitive and convenient in practice. A popular, slightly informal view of Bayesian networks is that they represent causal graphs in which every arc represents a direct causal influence between the variables that it connects. A directed arc from X to Y captures the knowledge that X is a causal factor for Y . While this view is informal and it is easy to construct mathematically correct counter-examples, it is convenient and widely used by almost everybody applying Bayesian networks in practice. There is a well-established assumption, with no convincing counter-examples, that causal

graphs will automatically lead to correct patterns of independences. Lack of arcs between pairs of variables expresses simple facts about absence of causal influences between them. Independences between these pairs of variables follow from the structure of the graph. It is, thus, possible to construct Bayesian networks based purely on our understanding of causal relations between variables in our model.

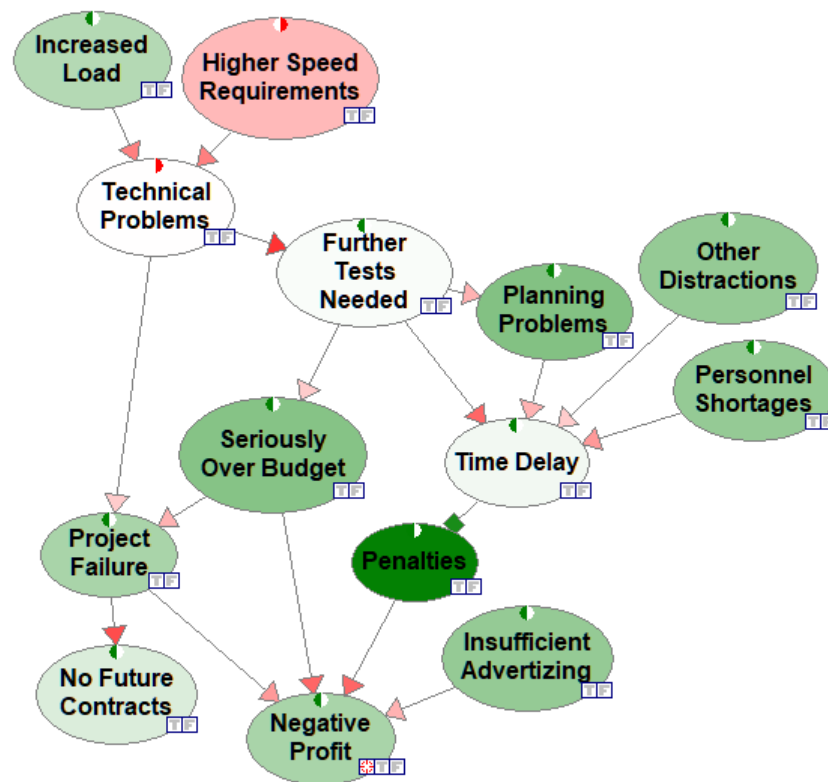
Both, the structure and the numerical parameters of a Bayesian network, can be elicited from an expert. They can also be learned from data, as the structure of a Bayesian network is simply a representation of independences in the data and the numbers are a representation of the joint probability distributions, which can be inferred from the data. Finally, both the structure and the numerical probabilities can be based on a mixture of expert knowledge, measurements, and objective frequency data.

Structural properties of Bayesian networks, along with their numerical parameters allow for probabilistic reasoning within the model. Probabilistic reasoning within a BN is induced by observing evidence. A node that has been observed is called an evidence node. Observed nodes become instantiated, which means, in the simplest case, that their outcome is known with certainty. The impact of the evidence can be propagated through the network, modifying the probability distribution of other nodes that are probabilistically related to the evidence. For example, the network constructed in section [Hello QGeNie!](#) allows for calculating the probability of high productivity (shown by color) given observations of hot weather and broken climate control device.



This calculation amounts at the foundations to a repetitive application of Bayes theorem in order to update the probability distributions of all nodes in the network. Different ways of applying Bayes theorem and different order of updating lead to different algorithms. Essentially, the existing algorithms for reasoning in Bayesian networks can be divided into three groups: message passing, graph reduction, and stochastic simulation. Explicit representation of independences allows for an increased computational tractability of probabilistic reasoning. Probabilistic inference in singly connected BNs is very efficient. Unfortunately, exact algorithms for multiply connected networks are liable to exponential complexity in the number of nodes in the network. Cooper (1990) has shown that the problem is NP-hard in general. Still, efficient software, like SMILE that is embedded into QGeNie, offers reasonable computing times even in networks consisting of thousands of nodes.

The following network (Project) models various considerations faced by an airplane manufacturer



Observed *Increased Load*, *Higher Speed Requirements*, and *Personnel Shortages* lead to light green *Negative Profit*. A Bayesian network model allows for calculating the most informative future observations and actions. The following graph shows the most informative observations for the model and the situation pictured above.

Most Effective Actions

⚡ ↺ ±.0 .00
.00 ±.0

All Observations Manipulations

Variable	Effectiveness
Seriously Over Budget	0.21
Project Failure	0.15
Insufficient Advertizing	0.05
No Future Contracts	0.04
Further Tests Needed	0.03
Technical Problems	0.02
Time Delay	< 0.01
Penalties	< 0.01
Planning Problems	< 0.01
Higher Speed Requirements	< 0.01
Increased Load	< 0.01
Personnel Shortages	< 0.01
Other Distractions	< 0.01

Embedding Bayesian networks technology into user programs

Bayesian networks can be embedded into custom programs and web interfaces, helping with calculating the relevance of observations and making decisions. [SMILE Engine](#), our software library embedding Bayesian networks, has been deployed in a variety of environments, including custom programs, web servers, and on-board computers.

4.4 Bayesian updating

[Bayesian networks](#) allow for performing Bayesian inference, i.e., computing the impact of observing values of a subset of the model variables on the probability distribution over the remaining variables. For example, observing a set of symptoms, captured as variables in a medical diagnostic model, allows for computing the probabilities of diseases captured by this model.

Bayesian updating, also referred to as belief updating, or somewhat less precisely as probabilistic inference, is based on the numerical parameters captured in the model. The structure of the model, i.e., an explicit statement of independences in the domain, helps in making the algorithms for Bayesian updating more efficient. All algorithms for Bayesian updating are based on a theorem proposed by Rev. Thomas Bayes (1702-1761) and known as *Bayes theorem*.

Belief updating in Bayesian networks is computationally complex. In the worst case, belief updating algorithms are NP-hard (Cooper 1990). There exist several efficient algorithms, however, that make belief updating in graphs consisting of tens or hundreds of variables tractable. Pearl (1986) developed a message-passing scheme that updates the probability distributions for each node in a Bayesian networks in response to observations of one or more variables. Lauritzen and Spiegelhalter (1988), Jensen et al.(1990), and Dawid (1992) proposed an efficient algorithm that first transforms a Bayesian network into a tree where each node in the tree corresponds to a subset of variables in the original graph. The algorithm then exploits several mathematical properties of this tree to perform probabilistic inference.

Several approximate algorithms based on stochastic sampling have been developed. Of these, best known are *probabilistic logic sampling* (Henrion 1998), *likelihood sampling* (Shachter & Peot 1990, Fung & Chang 1990), *backward sampling* (Fung & del Favero 1994), *Adaptive Importance Sampling* (AIS) (Cheng & Druzdzel 2000), and quite likely the best stochastic sampling algorithm available at the moment, *Evidence Pre-propagation Importance Sampling* (EPIS) (Yuan & Druzdzel 2003). We believe that the EPIS algorithm is currently the state of the art algorithm for Bayesian networks and have included it in both QGeNIe and GeNIe. Approximate belief updating in Bayesian networks has been also shown to be worst-case NP-hard (Dagum & Luby 1993).

In most practical networks of the size of tens or hundreds of nodes, Bayesian updating is rapid and takes between a fraction of a second and a few seconds.

4.5 Changes in structure

Changes in structure are external interventions that modify a system in question. An example of a change in structure is imposition of a new tax within the economic system of a country. It is of critical interest to decision makers to be able to make predictions of the effects of changes in structure. Since the model reflects the reality in its unmanipulated form and changes in structure of the kind

contemplated by the decision maker have perhaps never been performed, predicting their effect is in general daunting.

In order to be able to predict the effect of arbitrary changes in structure, it is necessary that the model contain causal information. Directed graphs allow for representation of causality. One may adopt the convention that each arc in the graph denotes a direct causal relation between the parent and the child node. We recommend that all models built are causal in that sense. The operation of controlling a value is an example of a causal manipulation and may result in changes in structure.

QGeNIe, GeNIe and SMILE are unique in supporting changes in structure in decision models. Please see [Controlling values of variables](#) section of this document for additional details.

This page is intentionally left blank.
Remove this text from the manual
template if you want it completely blank.

Building blocks of QGeNle

5 Building blocks of QGeNIe

5.1 Introduction

This section of QGeNIe documentation reviews elements of QGeNIe that form building blocks to its functional modules. We will often refer to the following terms, so we will define them briefly:

Property sheets

Property sheets are used to define the properties of the networks, nodes and submodels used. Each element has its own property sheet.

Menus

Menus are collections of possible actions and option switches, typically present on the top of the main program window.

Pop-up menus

Pop-up menus are menus that appear on the screen when the user right clicks on any element in QGeNIe. Sometimes pop-up menus appear when clicked upon in a dialog window.

Toolbars

Toolbars are used for quick access to frequently used commands. All the commands found on the toolbar can be typically found in one of the menus.

Dialogs

Dialogs are usually small windows containing descriptions of features that can be interactively modified.

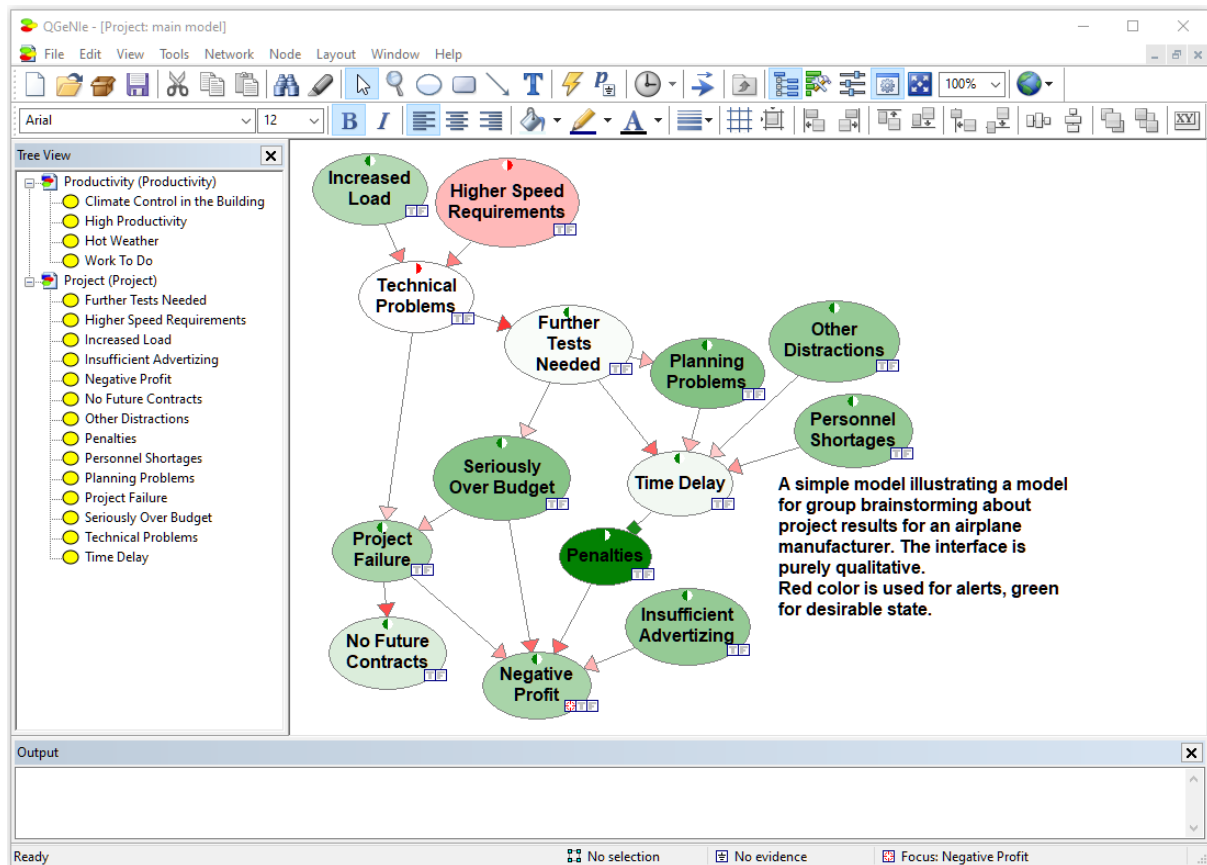
Keyboard shortcuts

Many commands in QGeNIe can be executed by pressing a sequence of keys. We will report shortcuts for many of the commonly used operations but will also provide a section summarizing all QGeNIe shortcuts at the end of this chapter.

5.2 QGeNIe workspace

5.2.1 Introduction

QGeNIe workspace is what you see and use when you work with QGeNIe. Its main goal is to allow you to view the network under development in many alternative ways, which we call *views*. QGeNIe window looks as follows:



The fundamental views that most users work with are the [Graph View](#) and the [Tree View](#).

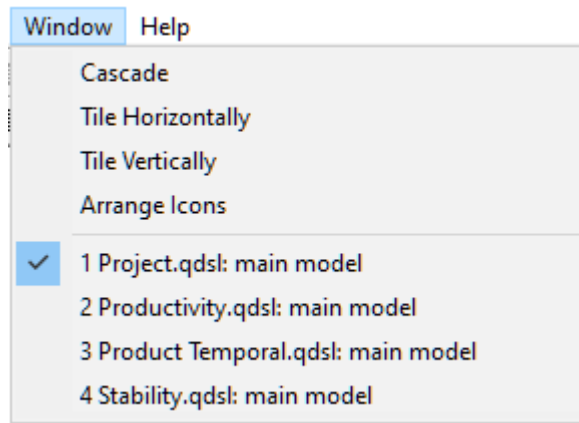
The [Status Bar](#) displays the focus node and the number of evidence nodes that are set for the active model.

The [Output Window](#) is where QGeNIe will display important messages for you.

The tool bars and menus, along with the property sheets for nodes, submodels and networks are described in the [Building Blocks of QGeNIe](#) section.

QGeNIe allows for working with multiple models at the same time. Each model can have an unlimited number of arbitrarily nested submodels. Each model and submodel can be viewed simultaneously in the [Graph View](#) and the [Tree View](#).

At any given moment, only one window in QGeNIe workspace can be active. The active window can be easily recognized by both being completely in the foreground and by its distinct characteristic (such as a darker blue top bar) determined by your Windows settings. To make a window active just click on any of its elements. An alternative way of making a window active, useful when the workspace contains many windows, is by selecting its name on the *Window* menu.



The *Window* menu displays a list of all currently open windows. A check mark appears in front of the name of the active window. The user can select any of the windows in the bottom part of the *Window* menu to be active. To select a window, select its name and release the mouse button.

The *Window* menu offers commands that help in arranging multiple views of multiple documents in the application window:

Cascade command arranges all windows in an overlapping fashion with their *Title* bars clearly visible.

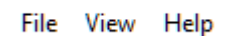
The *Tile Horizontally* and *Tile Vertically* commands arrange windows into non-overlapping tiles either horizontally or vertically (depending on the option selected), allocating equal space to each window.

Arrange icons arranges icons of all minimized windows at the bottom of the main QGeNIe window. Please note that if there is an open window that covers the bottom of the main window, then it may cover some or all of the icons and they may not be visible.

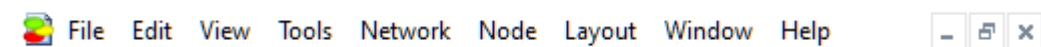
5.2.2 The menu bar

The *Menu Bar* is displayed at the top of the QGeNIe window and displays menu headings. Clicking on a menu heading will open the menu and display a list of commands under that menu. You can click on a command name to choose that command. The most frequently used commands are also displayed as tool bars (collections of buttons under a common theme). The menus that are available depend on what is open in the workspace. QGeNIe has two different menu bars:

(1) when no model is open,



(2) when a model is open in the workspace,



The difference between the two has to do with the fact that most of the menus apply to an open model. Here is a brief introduction to each of the menus. Details for each of the menus are covered in various sections of this manual.

File Menu has commands for creating, opening, saving, closing, and printing a model. It also allows for exporting QGeNIe models to GeNIe and exporting all textual comments and annotations from a model to a text file.

Edit Menu has commands for cutting, copying and pasting elements of the model, searching for an element, selecting multiple elements, as well as highlighting model elements.

View Menu has commands for viewing and hiding various toolbars and *Status Bar*, selecting format of labeling for nodes, zooming the graphical representation of models.

Tools Menu has commands for selecting various drawing tools for model construction.

Network Menu has commands for displaying network properties, clearing all evidence, locating the focus node if there is one defined, selecting the updating algorithm, and operating on qualitative dynamic Bayesian network models.

Node Menu has commands for displaying *Node properties*, setting and clearing evidence, controlling node value, setting the *Focus*, clearing evidence, selecting view type of the node, locating relations of the node, and showing its connections to the rest of the model.

Layout Menu has commands to adjust grid properties and layout options for the nodes.

Window Menu has commands to arrange the open windows in GeNIe and to switch between windows.

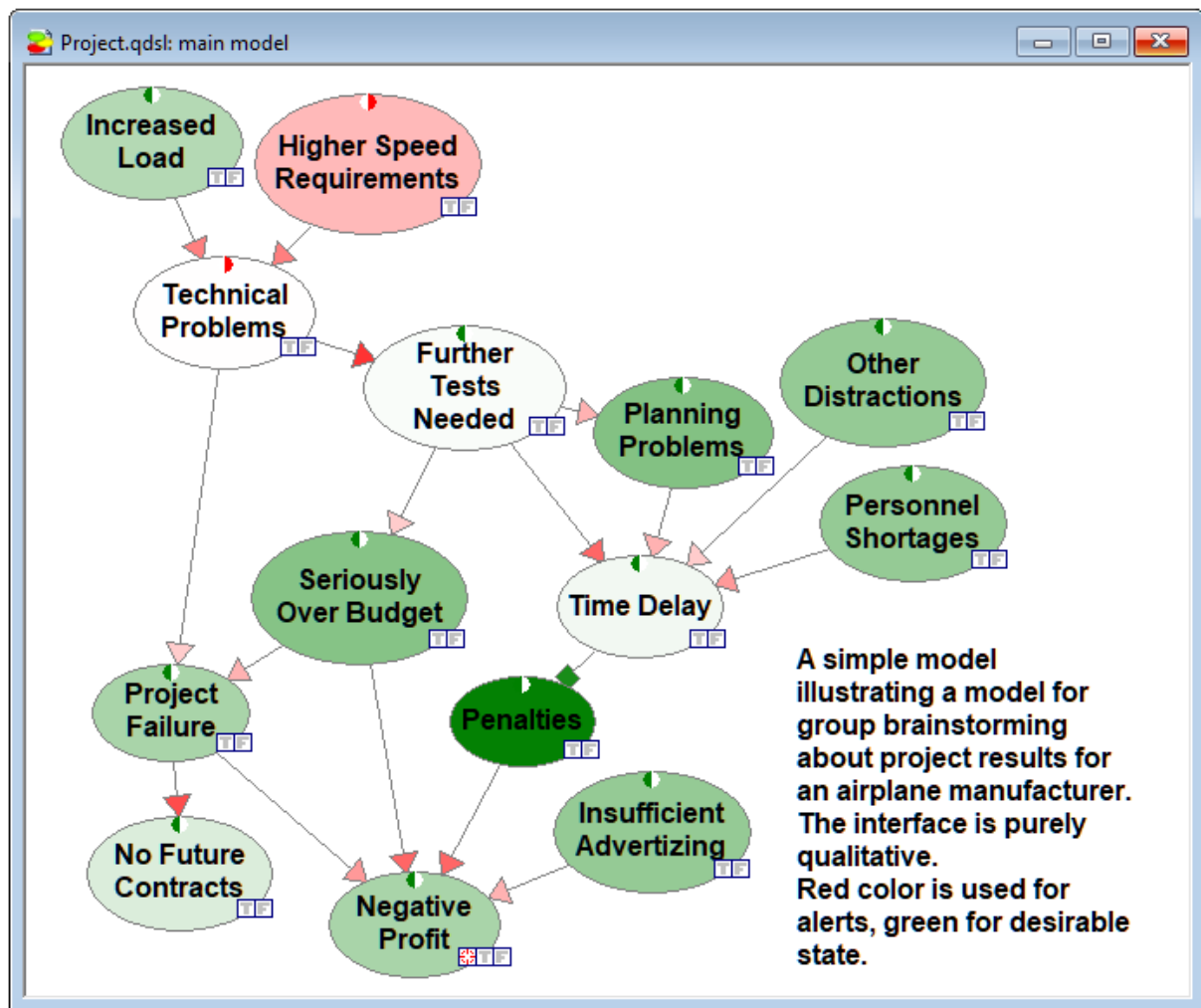
Help Menu has commands to display online help and modify help settings.

Note: Menu items that are grayed out either do not apply to the current selection or are unavailable.

5.2.3 Graph view

The *Graph View* is the primary model view in QGeNIe. It shows a directed graph in which each node represents a variable and each arc represents an influence between two nodes. It is an intuitive environment for creating and editing networks, useful in gaining insight into models by making the structure of their graphs explicit. A slightly modified version of the *Graph View* is the *Cost Graph View*, described in a separate section.

An example of the graph view is shown below:



Graph View can be enhanced dramatically by structuring the model hierarchically into submodels. Please see the section on QGeNIe [submodels](#) to learn more about it.

The *Layout Menu* and buttons on the [Format Toolbar](#) can be used to change the aesthetic properties of the *Graph View*.

Commands for displaying or hiding the grid and aligning the elements in the graph can be found in the *Layout Menu*.

The [Format Toolbar](#) has buttons for changing the font, color and size of the labels of the nodes, and buttons for performing the aligning operations on text and on the elements of the graph.


Please see *Layout Menu* and [Format Toolbar](#) for more information.

Opening a Graph View window

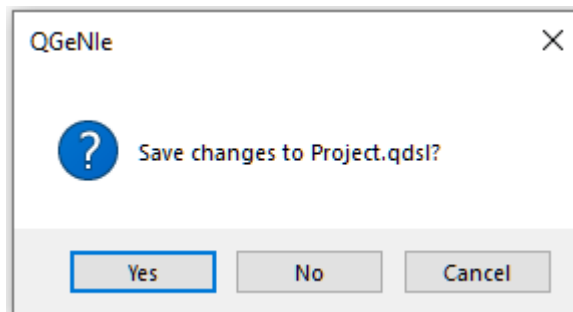
The graph view window is always open by default whenever a new model is opened or created. It is a large sheet with variables placed at user-designated locations. You can select the *Open* option from the [File Menu](#) to open a saved network file. You can create a new network by selecting the *New* option from the [File Menu](#).

Closing a Graph View window

There are three ways in which you can close a *Graph View* window:

- By clicking on the *Close* () button at the top right of the *Graph View* window.
- By selecting the *Close* option in the [File Menu](#).
- By selecting the *Close Network* option from the *Network Pop-up* menu in the *Tree View*.

If you close all the windows (the main window and all sub-model windows) of an open network then it will result in closing the file, and if any changes have been made on the network, QGeNIe will give you a warning with the dialog box shown below.



You can save the changes by clicking on the *Yes* button. Click on *Cancel* to continue working on the network.

Working with models in the Graph View:

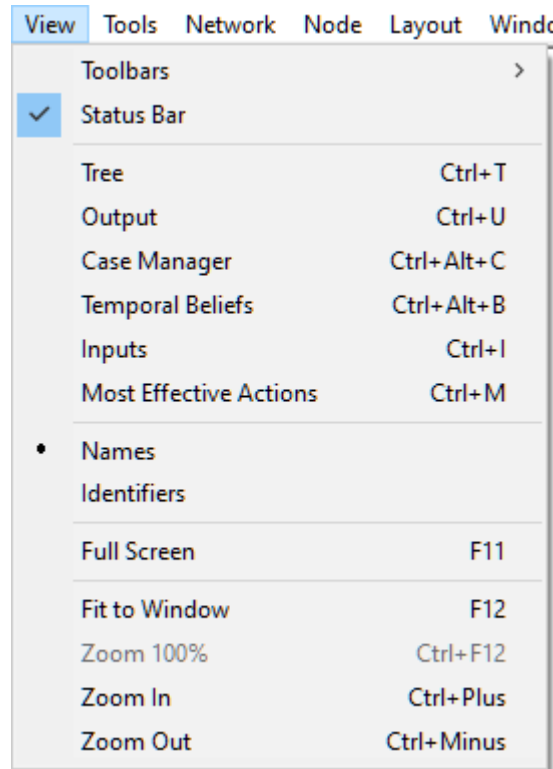
Each network is opened in a separate graph view sheet in the workspace. Double clicking on a clear area of the graph view sheet will open the [Network Property Sheet](#). Right clicking on any clear area of the graph view sheet will display the *Network Pop-up* menu, which can be used to modify various properties of the network.

Working with nodes in the Graph View:

You can draw new nodes in the *Graph View* by selecting the appropriate tool from the [Tool Menu](#) or clicking on the appropriate button on the [Standard Toolbar](#).

Double clicking on any node will open its [Node Properties Sheet](#). Right clicking on the node will display the *Node Pop-up* menu. It can be used to modify the properties of the node.

By default, QGeNIe displays the node names within the node icons in the *Graph View*. If you want QGeNIe to display node identifiers instead, select *Identifiers* in the *View Menu* to switch the display to identifiers.



Working with submodels in the Graph View:

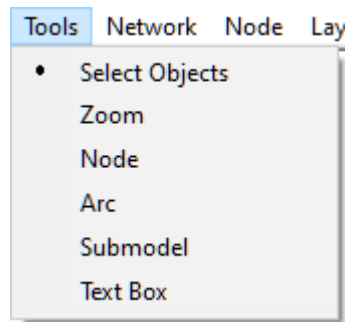
Double clicking on any submodel in the *Graph View* will open a *Graph View* for that submodel. You can go back to the main network by either minimizing or closing the submodel using the buttons on the top right of the submodel window. Right clicking on the submodel will display the *Submodel Pop-up Menu*. It can be used to modify the properties of the submodel.

Adding model elements

You can draw the following model elements in the *Graph View*:

- Nodes
- Submodels
- Arcs
- Text Boxes

To add any model element to the *Graph View*, you have to first select a tool, either from the *Tools* menu below



or a button from the [Standard Toolbar](#) below

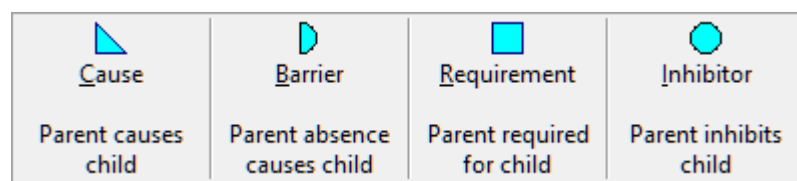


The next step is to click on any clear area of the *Graph View*. For all elements except the arc, QGeNIe will draw the icon of the element in the *Graph View*.

To add an arc between two nodes,

1. Select the arc tool and click on the parent node.
2. Drag the mouse cursor to the child node and release the mouse button.

QGeNIe will show the following dialog:



Clicking on any of the four tiles, representing four types of causal connections, will add an arc of the corresponding type from the parent to the child node.

To learn more about creating nodes and arcs, See [Building a Bayesian network](#).

Selecting, re-sizing, and moving model elements

You need to select an element to perform an operation that is specific to it.

You can select a single element by clicking on it. The element will show tracker points (small squares around the perimeter of the selected element) that can be used to re-size it. You can re-size the element

in any direction by dragging one of these points. Arcs cannot be re-sized, as QGeNIe automatically draws them for you between pairs of nodes connected by arcs.

Sometimes, it may be convenient to select several nodes at a time. There are four ways of selecting nodes in groups in the *Graph View*:

- **Rectangular selection**

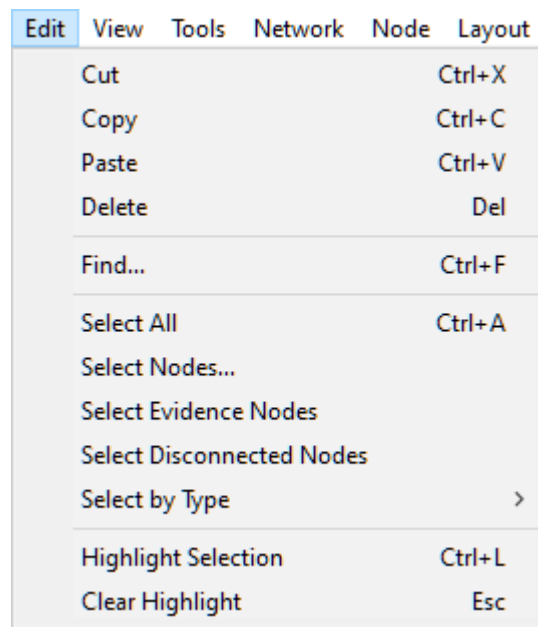
You can select a group of nodes by clicking on an empty area of the *Graph View* and dragging a selection rectangle in any direction that you wish. Any node completely within the rectangle will be selected. When you draw a rectangular selection with *SHIFT* key pressed, the selection will be added to the existing selection. When you draw a rectangular selection with both *CTRL* and *SHIFT* pressed, the current selection status will be inverted, i.e., model elements selected will become unselected and model elements unselected will become selected.

- **Extended selection**

Once you have an element, such as a node or a group of nodes or text boxes, selected, you can add or remove individual elements from the selection by holding the *SHIFT* key while clicking on them. This selection process acts as a toggle, i.e., nodes that are currently selected will be de-selected.

- **Group selection**

You can select a specific group of nodes or all nodes in the current window by choosing *Select All* from the *Edit Menu*. The shortcut for this selection is *CTRL+A*.



- **Select Nodes... dialog**

Select Nodes... dialog allows for selecting nodes based on their names or IDs. We will discuss this powerful selection tool in section [Selection of model elements](#).

- **Select Evidence Nodes**

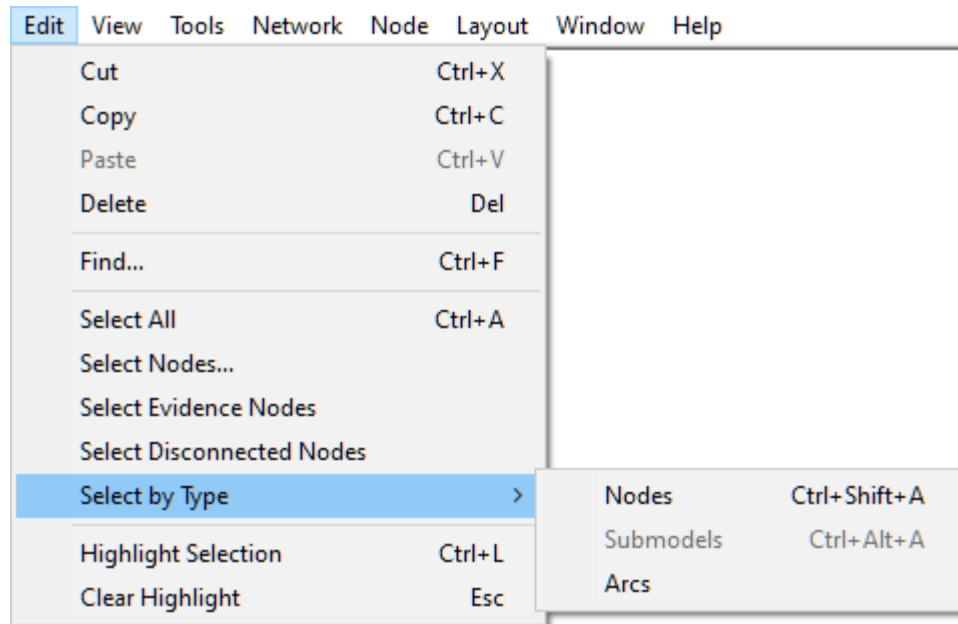
This command selects all nodes that have observed evidence in them. If there are no evidence nodes in the network, this choice is dimmed.

- **Select Disconnected Nodes**

This command selects all nodes that are disconnected from the graph, i.e., have neither parents no children. Typically, such nodes are left disconnected by mistake, so selecting and subsequently highlighting them is a good way of finding them in the model, which may be otherwise challenging in sizable models. If there are no disconnected nodes in the network, this choice is dimmed.

- **Select by Type sub-menu**

You can select model element by type, i.e., nodes, submodels, arcs, and text boxes in the entire model by choosing the appropriate option from the *Select by Type* sub-menu shown below.

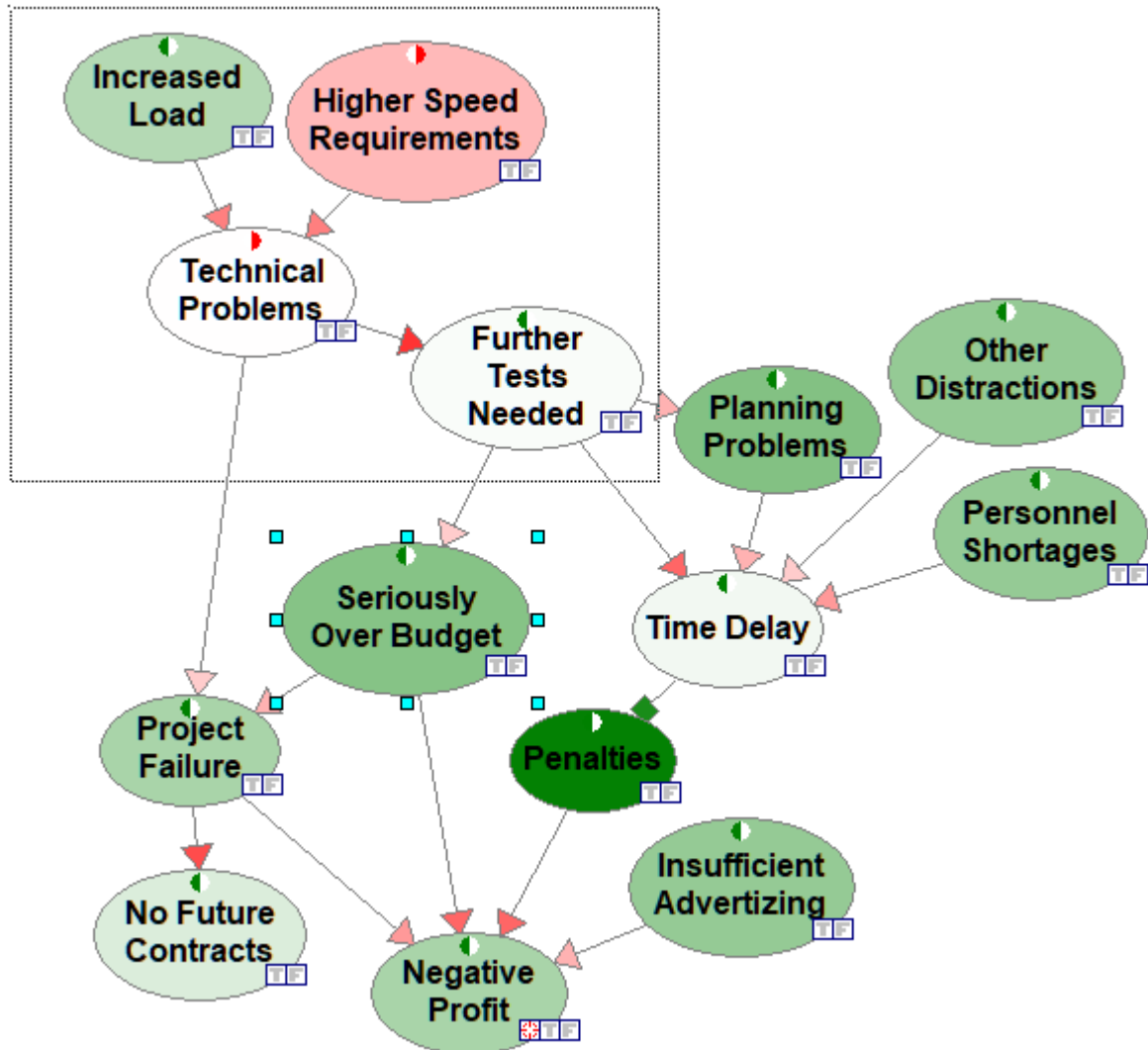


You can move a node or a group of nodes to another location by dragging it. When you press the Shift key (you need to do it after clicking on the node or the selection, as *SHIFT*-Click has the meaning of adding an object to the selection), the moving is limited to horizontal and vertical directions.

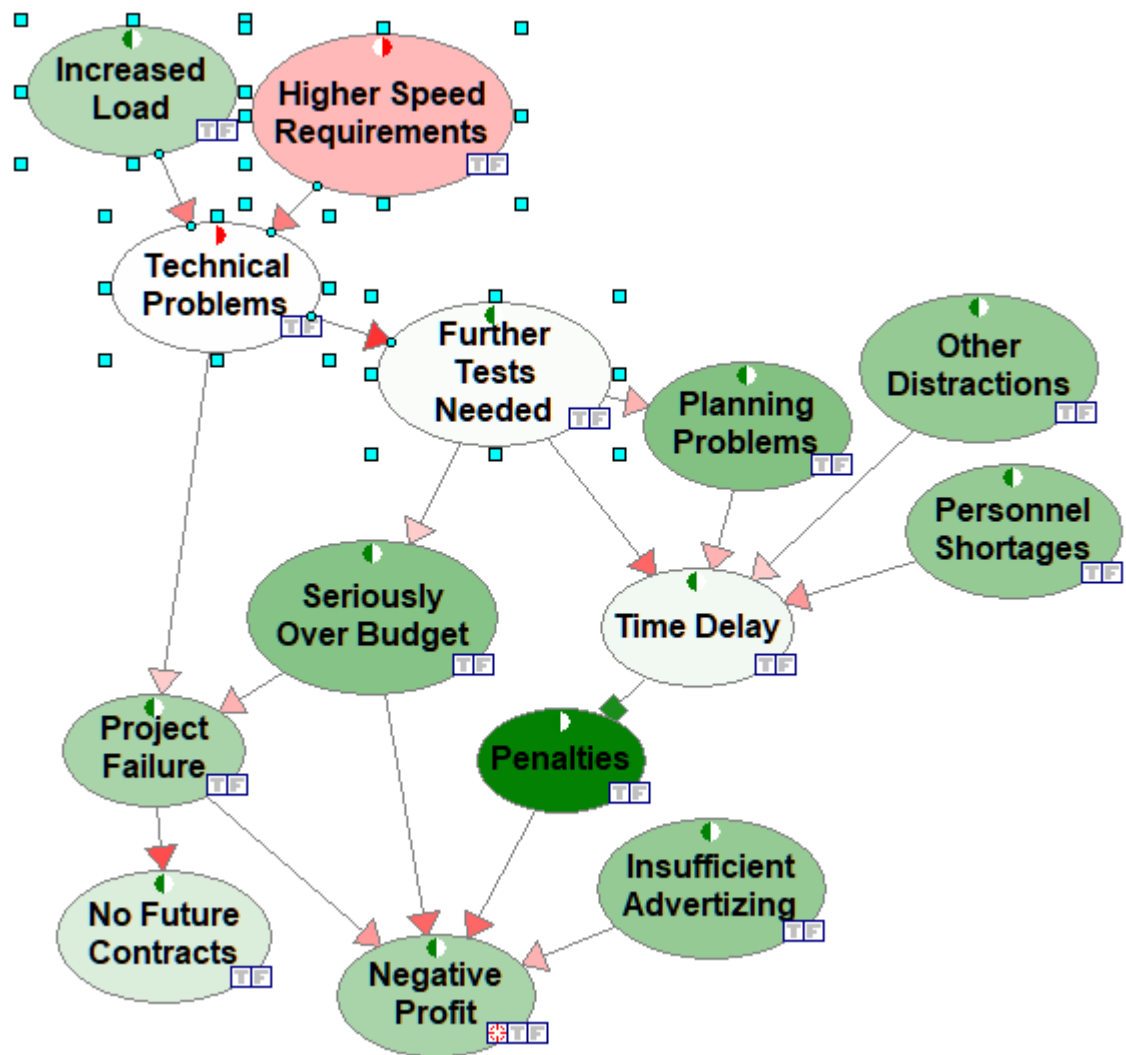
A node or a set of nodes can be also moved between submodels. To move a node to a different submodel, drag and drop it into a submodel icon or into a submodel window.

Highlighting selected model elements

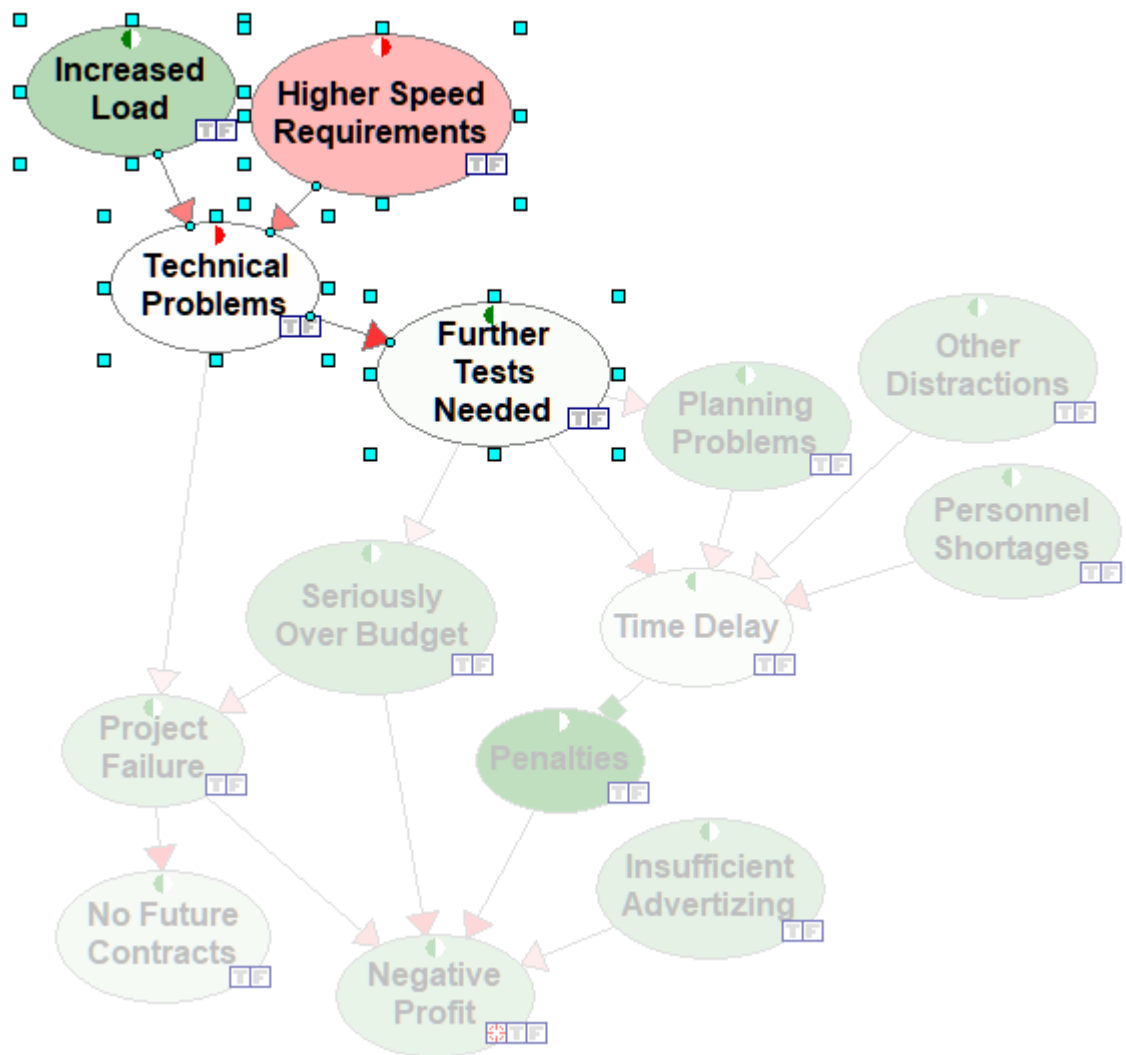
QGeNIe allows for highlighting the selected model elements (nodes, arcs, submodels, etc.) through the command *Highlight Selection* in the *Edit Menu* (short-cut *CTRL-L*) or through pressing the Highlight selection (🖋️) button. The following screen shot shows a rectangular selection of model elements in *Graph View*,



This results in selecting a group of nodes and arcs



When these elements are highlighted, the *Graph View* changes to the following



Pressing the *ESC* button or selecting *Clear Highlight* from the *Edit Menu* clears the selection.

Extending the workspace




QGeNIe is somewhat restrictive in how far the *Graph View* space stretches in every direction and there is typically not too much space beyond the borders. If you want to move model elements in any direction, please bring them close to the border and this will force QGeNIe to move the border.

Deleting model elements

To delete an element or a group of elements, select them and press the *Delete* key on the keyboard.

Deleting a node deletes all its incoming and outgoing arcs.

Copying, cutting, and pasting model elements

Model elements or group of elements can be copied or cut into the *Windows Clipboard*, and subsequently pasted into the same or a different *Graph View* window or into another application. To invoke any of these commands, please select them from the *Edit Menu* or from the *Standard Toolbar* (shown above, buttons , , and , respectively). Nodes pasted into the same window will preserve their incoming arcs but will lose their outgoing arcs. The reason for this is simple - the nodes need their parents in order to preserve their definition, which is typically conditional on its parents. On the other hand, preserving the outgoing arcs would mean that their child nodes would have double set of parents, i.e., the original nodes and their copies.


Clipboard supports multiple formats. QGeNIe stores data simultaneously in three formats:

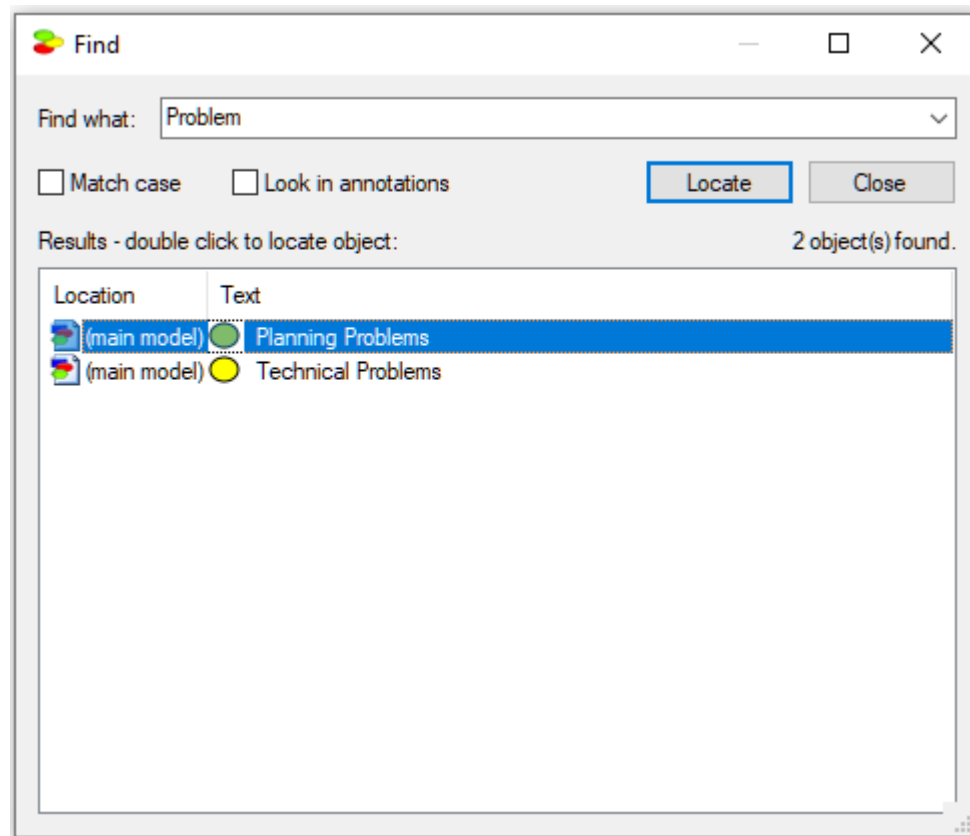
1. Native QGeNIe format for cut, copy, and paste operations between models
2. Standard text, for example names of selected nodes and comments
3. Standard bitmap, which is used for selected objects that have a graphical component
4. Picture (enhanced metafile), used for objects that have a graphical component

Each time you invoke *Copy* or *Cut*, data in all three formats are sent to the *Clipboard*. When pasting into a different application, all you need to do is select *Paste special* from that application's *Edit* menu (e.g., in *Word* or *PowerPoint*). This should bring the dialog box with all format names. Plain Paste command does different things in different programs - it pastes data as text into Word by default but may paste a bitmap image in *Paint*.

To copy a complete model as an image, first select all elements in the model using the *CTRL+A* shortcut. Then use the *CTRL+C* shortcut to copy the model to the clipboard. Subsequently, open the program in which you want to paste the model image (e.g., Adobe Photoshop or MS Word) and use *Paste special* and then *Bitmap* or (better yet!) *Picture* in *Word*. If you use *Photo Editor* or any graphics editing program, *Paste* will paste the model image by default.

Text-based search for model elements (Find command)

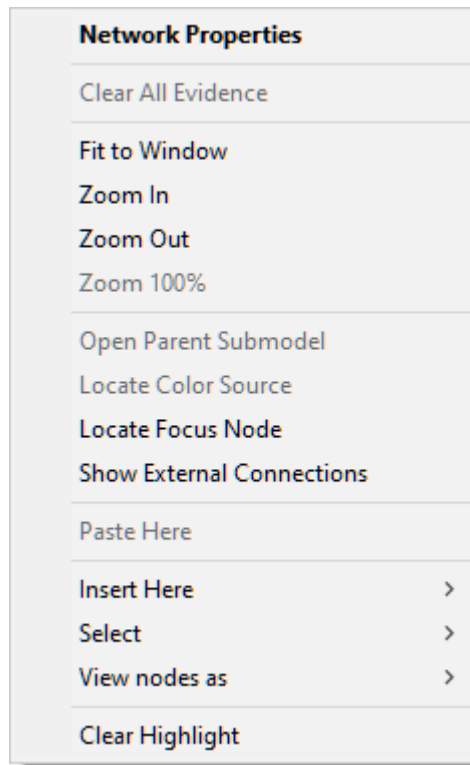
Find () button from the toolbar or selecting *Find* from the *Edit Menu* (shortcut *CTRL-F*) invokes the following dialog, which allows for finding model elements, such as nodes, through text search



The *Find* dialog allows for finding the text string specified in the *Find what* box in the names and identifiers of all elements of the models and submodels. It will also search within annotations if the *Look in annotations* check box is checked. *Match case* flag allows for additional customization of the search. Pressing the *Find* button starts the search. If any matches are found, they are displayed in the dialog box and the *Find* button changes into the *Locate* button. Selecting one of the results and pressing the *Locate* button locates the selected node in the [Graph View](#), centers it, and flashes three times. You can also locate a node by double clicking on one of the results.

Network Pop-up menu for Graph View

The *Network Popup Menu* for the *Graph View* can be accessed by right clicking on any clear area of the *Graph View*. Some of the options might be disabled depending on the properties of the network selected.



Most of the commands found here can be also invoked from the *Network Menu*.

Network Properties (the default operation) opens the [Network Properties](#) sheet for the network.

Clear All Evidence is the same command as in the corresponding command in the *Network Menu*.

Fit to Window makes the network as large or as small as it takes to fit entirely in the *Graph View*.

Zoom In zooms into the network. Every application of this command increases the zoom by 25%. The current zoom percentage is displayed on the top right of the [Standard Toolbar](#). A similar effect can be obtained by using the zoom tool from the [Standard Toolbar](#) or the [Tool Menu](#).

Zoom Out is the opposite of the *Zoom In* and it zooms out of the network. Every application of this command decreases the zoom by 25%. The current zoom percentage is displayed on the top right of the [Standard Toolbar](#).

Zoom 100% brings the model in the *Graph View* to its normal size.

Open Parent Submodel is enabled only if the current network in the *Graph View* is a submodel of another network. The result of this command is opening the *Graph View* window displaying the parent submodel.

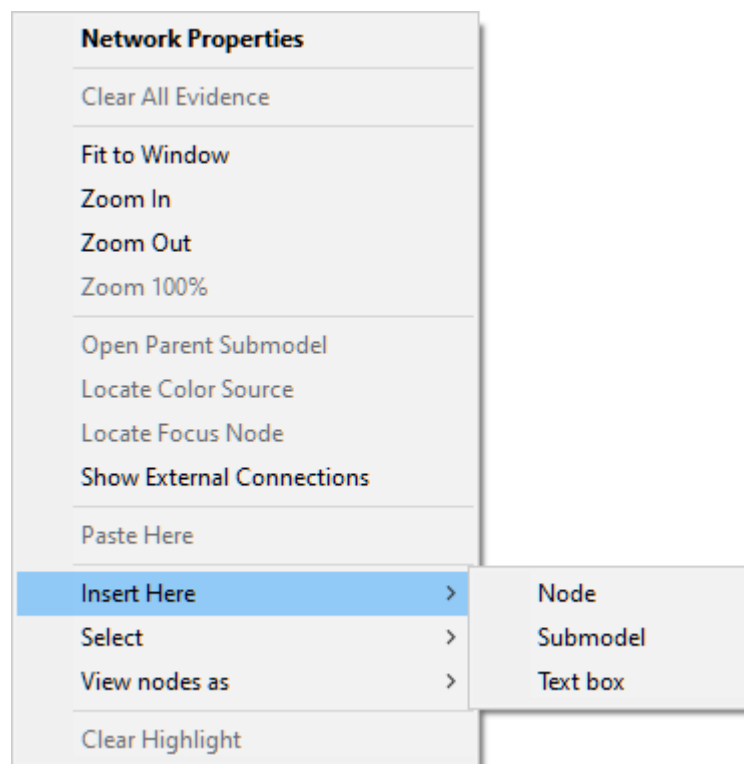
Locate Color Source is active only when the *Network Popup Menu* is executed inside a submodel. It finds the node that feeds its color to the submodel icon in the *Graph View*.

Show External Connections is a switch that helps in model navigation. When the switch is on, QGeNIe shows all nodes that are either direct predecessors (parents) or direct successors (children) of the nodes in the current submodel but are not present in the current submodel. QGeNIe will display these nodes at the margin of the *Graph View*.

Locate Focus Node finds the node designated as focus.

Paste Here pastes the contents of the clipboard onto the *Graph View* into the exact position of the mouse click that invoked the pop-up menu. This choice will be active only if the clipboard has data that have been entered using the *Cut* or *Copy* command within QGeNIe. You cannot *Cut* or *Copy* items from other programs into QGeNIe *Graph View*. You can *Cut* or *Copy* nodes between two running instances of QGeNIe.

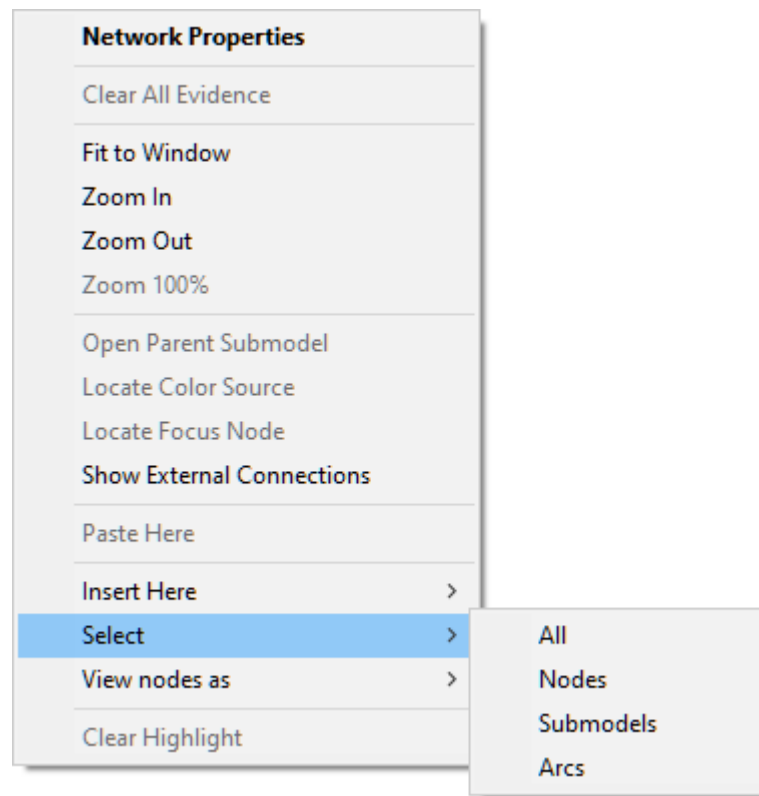
Insert Here submenu



The *Insert Here* submenu contains a list of all elements that can be drawn in the *Graph View*. Select any of the items on the list to place that item at the current cursor position. See [Components of QGeNIe models](#) section for more information on each item.

Select submenu

Selection of items enables certain operations to be performed on them without affecting other items that are not selected. Some options in QGeNIe will not be enabled unless some item has been selected.



The *Select* submenu contains the following options: *All* (select all items in the *Graph View*), *Nodes* (select all nodes in the *Graph View*), *Submodels* (select all submodels in the *Graph View*), and *Arcs* (select all arcs in the *Graph View*).

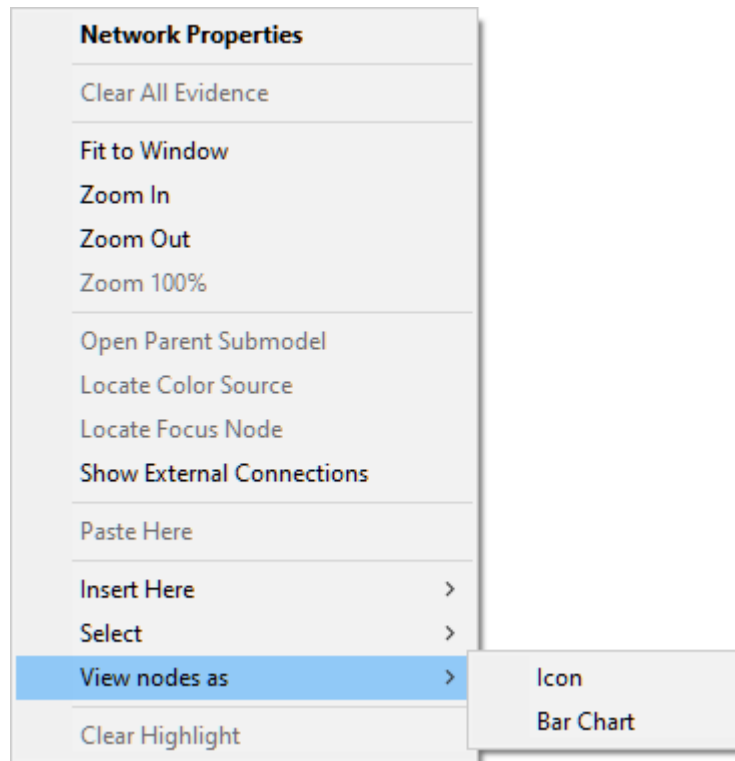
Selection modifiers

Simple selection, such as rectangular selection and the selections by color, by type and by the diagnostic type can be modified by pressing the *SHIFT* key or *CTRL* and *SHIFT* keys simultaneously. Effectively, three selection modes are available:

- *Default selection*: When neither *SHIFT* nor *CTRL* keys are pressed, leads to selection of only objects selected. The remainder of the model elements becomes unselected.
- *Add-to election*: When the *SHIFT* key is pressed but *CTRL* key is not pressed, the objects selected by the selection operation are selected, the remaining model elements do not change their selection status.
- *Invert selection*: When both the *SHIFT* and the *CTRL* keys are pressed, the objects selected by the selection operation invert their selection status, the remaining model elements do not change their selection status.

For example, consider the *Project* model. Let us select all model elements (nodes and arcs) with *Ctrl+A* first, then go to the *Edit Menu* and *Select* submenu. When keeping both *SHIFT* and *CTRL* pressed, we select *Arcs*. The selection status for arcs will invert and only nodes will remain selected.

View nodes as submenu

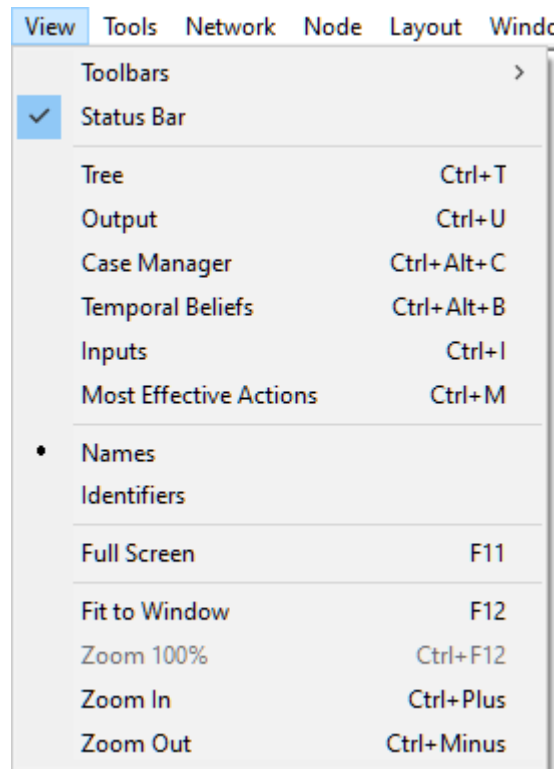


The *View nodes as* submenu is used to select how the nodes should be displayed in the *Graph View*. It is similar to the *View As* submenu in the [Node](#) menu but it applies to all nodes rather than to the selected nodes.

5.2.4 Tree view

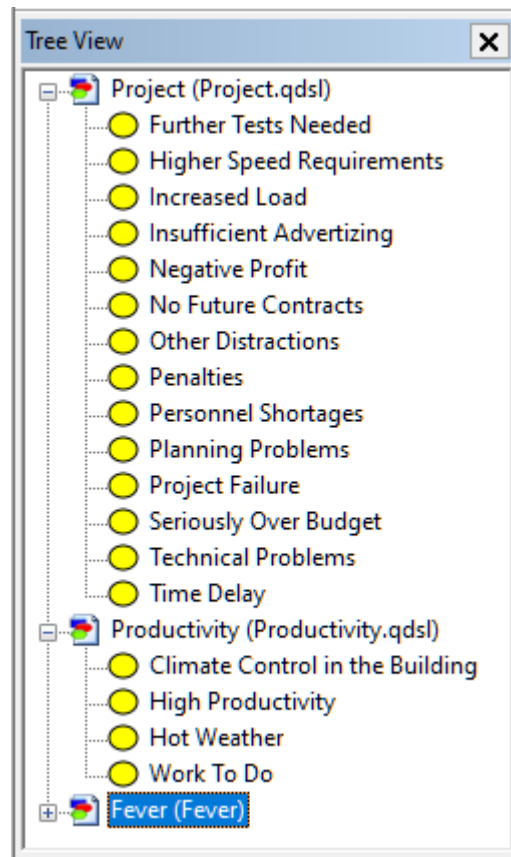
QGeNIe provides an alternative method of model navigation known as *Tree View*. The *Tree View* in QGeNIe is very similar to Windows tree view. It shows a hierarchical, alphabetically sorted list of all networks currently open, and all the nodes in the network. Most operations available in the [Graph View](#) can be also performed in the *Tree View*. Whatever changes are made in the *Tree View*, they are reflected immediately in the [Graph View](#). The *Tree View* can be also used to navigate in the [Graph View](#), for example to open submodel windows. Another important feature of the *Tree View* is that you can drag and drop nodes between different submodels and networks. We will illustrate the basic elements of *Tree View* functionality in this section.

Tree View can be displayed or hidden by checking or un-checking the *Tree* option in the *View Menu*. You can also use the keyboard shortcut *CTRL+T* to toggle *Tree View* display.



The *Tree View* panel can be detached from its position and placed anywhere on the screen by dragging it using its title bar. It snaps back into place if dragged close to the left or right border of the QGeNIe window.

Shown below is a typical *Tree View* panel.



The *Tree View* above shows three networks, *Project*, *Productivity* and *Fever*. A network or a [submodel](#) can be expanded or collapsed by clicking on \oplus or \ominus beside its name. *Fever* is not expanded (hence, \oplus is displayed beside it). *Project* and *Productivity* are fully expanded (hence, \ominus is displayed beside them). Nodes that are part of the networks are listed within each of the trees.

Note: Double clicking on the name of a network or submodel will also expand or collapse it.

Working with models, submodels, and nodes in the *Tree View*

Right clicking on the network name, node name, or submodel name will open the corresponding *Network Pop-up* menu, *Node Pop-up* menu or *Submodel Pop-up* menu. You can use these menus to change properties of the network, node, or submodel. Follow the links to each of the menus for more information on how to perform these operations.

Moving nodes between networks and submodels

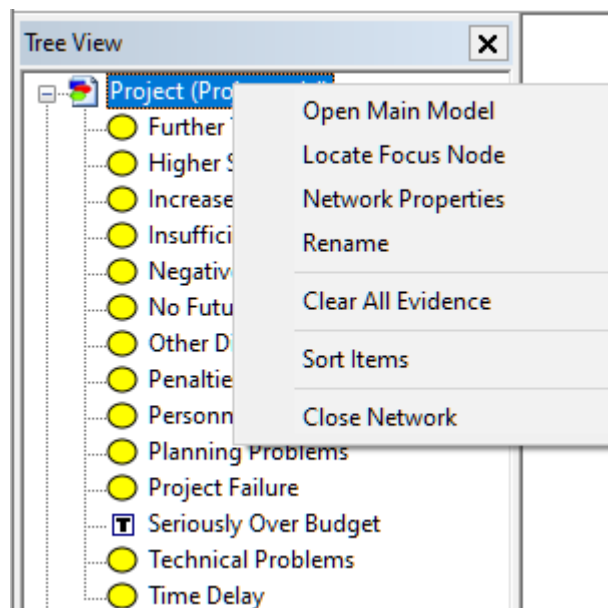
You can select any number of nodes and submodels in the *Tree View* and drag and drop them into any model or submodel in the *Tree View* or the *Graph View*. You can perform the drag and drop operations between different networks.

Note: If the nodes are being dropped in a submodel which is part of the same network, then the nodes are **moved** to their new location.

If the nodes are being dropped in a different network or a submodel in a different network, then the nodes are **copied** to their new location.

Network Pop-up menu in Tree View

The *Network Pop-up* menu in the *Tree View* can be invoked by right clicking on the network name in the *Tree View*.



Most of the choices are the same as in the *Network Pop-up* menu in the [Graph View](#).

Open Main Model opens the main network in the [Graph View](#).

Locate Focus Node finds the node designated as focus in the [Graph View](#).

Network Properties opens the [Network Properties](#) sheet.

Rename allows for changing the name of the network interactively.

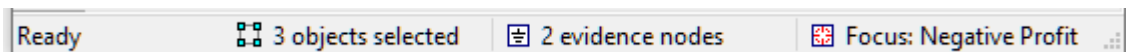
Clear All Evidence, dimmed if no evidence is present, allows for clearing all evidence.

Sort items causes the list of element names under the network to be sorted in alphabetical order.

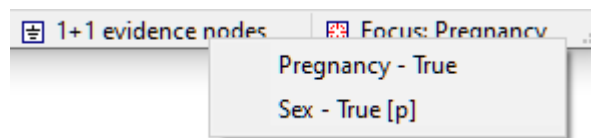
Close Network closes the network file. If any changes have been made to the network, then QGeNIe will warn you that you may lose the changes. You can also close the network by clicking on the *Close* button at the top right of the [Graph View](#) window or selecting *Close* option from the [File Menu](#).

5.2.5 Status bar

The *Status Bar* is a horizontal bar located at the very bottom of the main QGeNIe window. The *Status Bar* shows a short description of the command to be executed by the selected menu item or a tool on a toolbar on the left side and lists the number of model objects selected, the number of evidence nodes and on the right-hand side the focus node designated in the network.

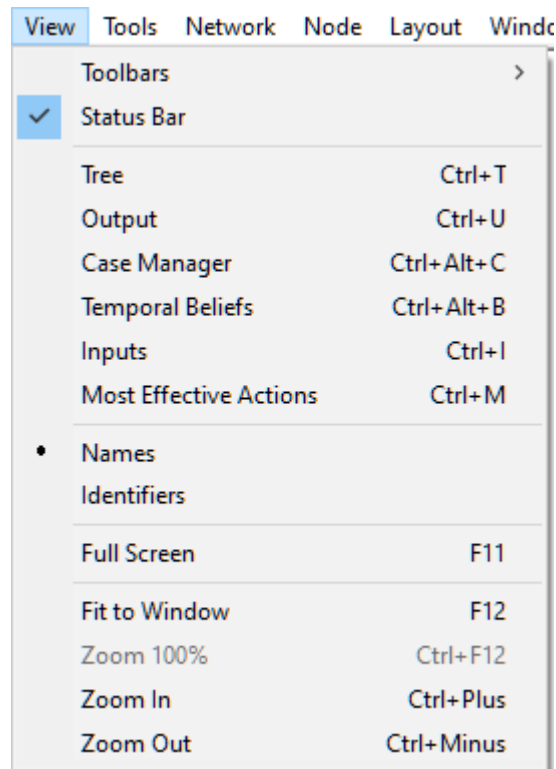


If there are any [evidence](#) nodes set in the network and any of the observed evidence propagates to other nodes, it will be indicated in the *Status Bar* as shown in the figure above. In the figure below, the text on the *Status Bar* *1+1 evidence nodes* indicates that there is one observation and one propagated evidence (i.e., evidence implied by the observations). The focus of the model is variable *Pregnancy*. The list of evidence nodes can be displayed by right-clicking on the text:



Any propagated evidence will be listed with a suffix [p]. To display only the observed evidence, right-click on the status bar when holding *CTRL* key. To display only propagated evidence, right-click on the status bar when holding the *SHIFT* key. Clicking a node name on the list of evidence or target nodes will locate the node in the *Graph View*.

Status Bar can be switched on and off by selecting or deselecting the *Status Bar* option from the *View Menu*. A check mark appears next to the menu item when the *Status Bar* is displayed.

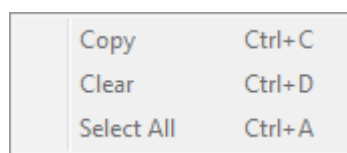


5.2.6 Output window

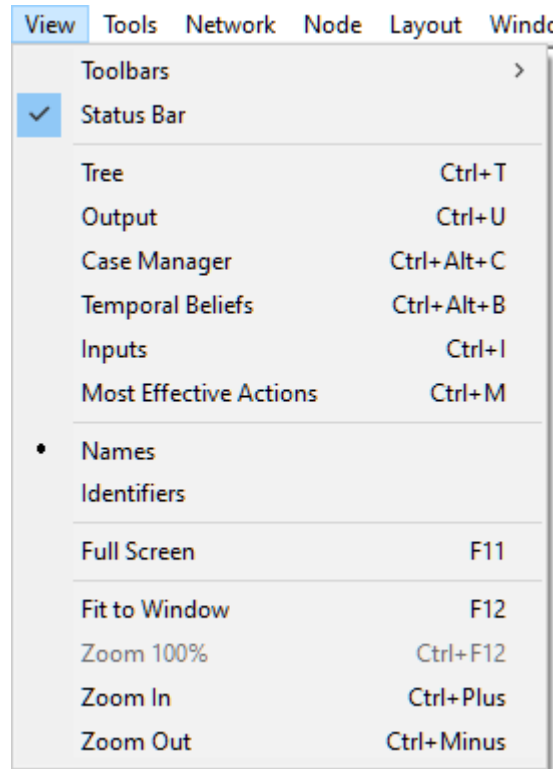
QGeNIe includes a *Output Window* that is used for notifying the user about possible problems with the model or program errors. The *Output Window* is usually shown in the bottom part of the screen, but can be moved to any location by the user.



You can perform selections of the messages or clear the contents of the *Output Window* through a context menu, available by right-clicking within the area of the window.

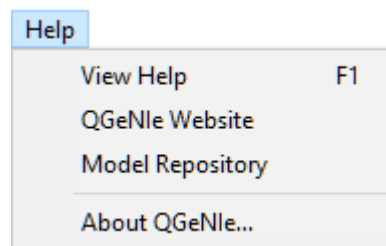


The *Output* window can be hidden or made visible by the user by changing the *Output* flag in the *View Menu* (shortcut *CTRL-U*).



5.2.7 Help menu

The *Help* menu offers commands providing assistance to the user:



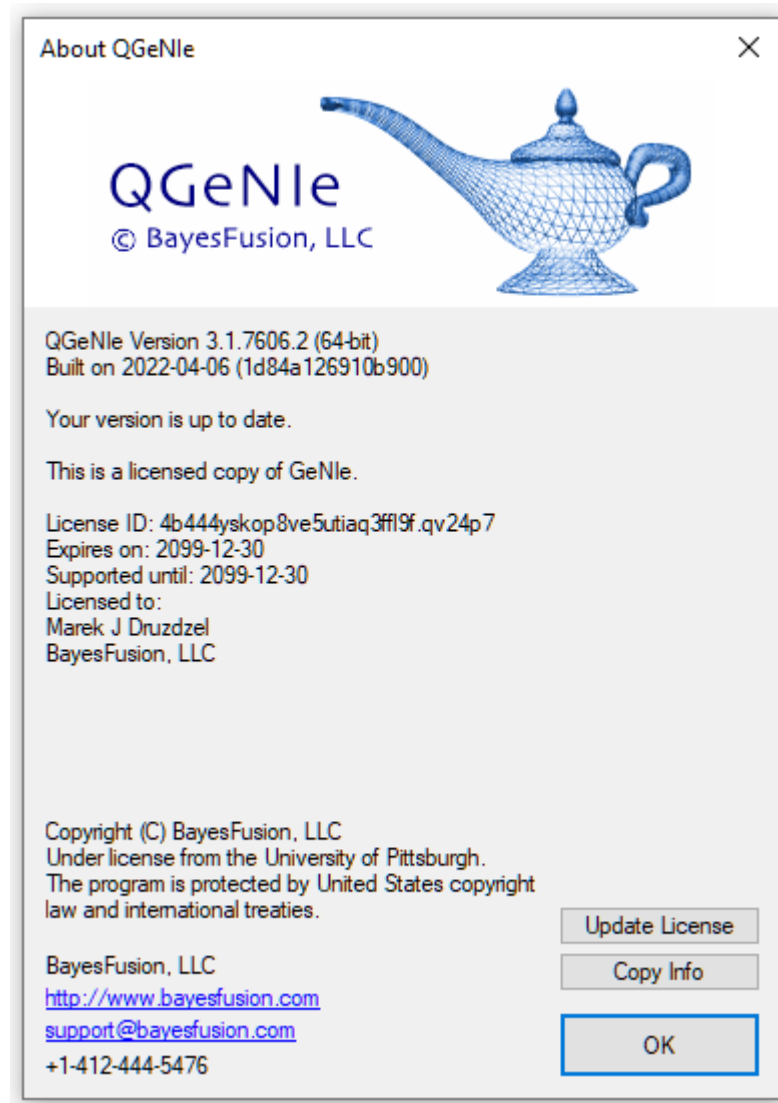
It contains four commands:

View Help (or *F1* key) invokes QGeNIe on-line help, which is the document that you are reading at the moment. QGeNIe on-line help is available in CHM, HTML and PDF formats. In addition to being distributed with the program, it is also available on BayesFusion, LLC's [support WWW pages](https://www.bayesfusion.com/). To exit the on-line help, simply close its window.

QGeNIe Website will take you to the official QGeNIe website at <https://www.bayesfusion.com/>.

Model Repository will take you to BayesFusion's interactive model repository through your default web browser.

About QGeNIe shows the following simple window:



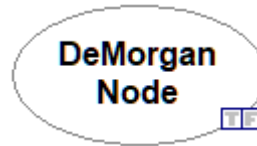
It displays a copyright notice, version number of your build of QGeNIe, license data, including license ID, its expiration date, support expiration date, the license holder's name and institution.

5.3 Components of GeNIe models

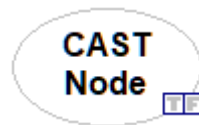
5.3.1 Node types

QGeNIe supports three types of nodes:

DeMorgan nodes, drawn as ovals and denoting uncertain variables representing DeMorgan gates. This is the fundamental node type in QGeNIe is created by default.



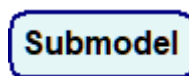
CAST nodes, drawn as ovals and denoting uncertain variables representing CAST gates. In order to create a CAST gate in QGeNIe, please hold the SHIFT key when clicking on the graph window when creating a node.



Both node types nodes represent propositional variables, i.e., variables that can take only two values, *True* and *False*. They are colored according to the marginal probability of the state *True*, unless this is reversed by a node setting, in which case the color shows the probability of the state *False*. The difference between the two node types is in the model of interaction of parents used in their definitions, [DeMorgan](#) and [CAST](#).

Note: QGeNIe does not allow for using both DeMorgan nodes and CAST nodes in the same model. CAST are legacy nodes that have been implemented in QGeNIe for the sake of compatibility with existing models. We recommend that all new models be based on DeMorgan gate.

Submodel nodes, drawn as rounded rectangles, denote submodels, i.e., conceptually related groups of variables. Submodel nodes are essentially holders for groups of nodes and helping with making models manageable.



To learn how to create nodes and arcs between them, see the introductory section on [Building a QGeNIe model](#).

5.3.2 The DeMorgan gate

This section gives a brief introduction to the DeMorgan gate (Maaskant 2006, Maaskant & Druzdzel 2008, Druzdzel 2009), which is the fundamental model of interaction among propositional variables in QGeNIe.

Those readers who were exposed to some logic in high school or college know that any logical function can be expressed in one of the Augustus De Morgan's canonical forms, an alternative (OR) of conjunctions (AND) or a conjunction of alternatives. It is also a fact stemming from so called De Morgan laws that the OR function and a negation can express the AND function. A combination of OR functions and negations can express any logical function. QGeNIe's DeMorgan gate offers essentially

an intuitive way of expressing any logical function and, in particular, a combination of ORs, ANDs, and negation.

Four basic influence types

The DeMorgan gate allows for modeling four basic types of influences that one variable (parent in the directed graph) can have on another (a child in the directed graph): (1) a positive influence (a *Cause*) and (2) a negative influence (a *Barrier*), both combining with other causes using a noisy OR interaction, (3) a required condition (a *Requirement*) and (4) a condition that prevents the effect from happening (an *Inhibitor*), both combining with other causes through an AND interaction.

We will now explain the meaning of the four types of causal influences and how they form any logical function when combined.

Cause

A cause is a parent that has a positive influence on the child. Please note that this influence does not need to be perfect. For example, smoking is generally believed to be a causal factor in lung cancer. Yet, incidence of lung cancer among smokers, while much larger than incidence of lung cancer among non-smokers, is still within a few percent. Hence, the conditional probability of lung cancer given that a person is a smoker is still fairly low. The cause increases the probability of the effect but does not need to be perfect in its ability to cause it.

Barrier

A barrier is a parent that decreases the probability of a child. For example, regular exercise decreases the probability of heart disease. While it is a well established factor with a negative influence on heart disease, it is unable by itself to prevent heart disease. One way of looking at a barrier is that it is dual to a cause: Absence of the barrier event is a causal factor for the child. One might go around the very existence of barriers by using negated versions of the variables that represent them. In the example above, one might define a variable *Lack of regular exercise*, which would behave as a cause of the variable *Heart disease*. This, however, might become cumbersome if *Regular exercise* participated in other interactions in a model. It might happen that it is a parent of both *Heart disease* and *Good physical shape*. Because *Regular exercise* decreases the probability of one and increases the probability of the other, Barrier, which is a negated Cause, is a useful modeling construct.

Requirement

A requirement is a parent that is required for the child to be present. There are perfect requirements, such as being a biological female is a requirement for being pregnant but there are also requirements that are in practice not completely necessary. For example, a sexual intercourse is generally believed to be a requirement for pregnancy, but it is not a strict requirement, as pregnancy may be also caused by artificial insemination.

Inhibitor

An inhibitor is a parent that prevents the child from happening. For example, rain may inhibit wild land fire. Like in the other types of interactions, the parent may be imperfect in inhibiting the occurrence of the child. Fire may start even if there is rain. Similarly to the relationship between causes and barriers, inhibitors are dual to requirements: Absence of an inhibitor event is a requirement for the child. One might go around the very existence of inhibitors by using negated versions of the

variables that represent them. In the example above, one might define a variable *No rain*, which would behave as a requirement for the variable *Wild land fire*. This, however, might become cumbersome if *Rain* participated in other interactions in a model. It might happen that it is a parent of both *Wild land fire* and *Good crop*. Because *Rain* is an inhibitor for the former and a requirement for the latter, Inhibitor, which is a negated Requirement, is a useful modeling construct.

For those readers who like a concise mathematical formulation, the four types of causes interact with their effect through the following logical formula:

$$e = (c_1 \vee c_2 \vee \neg b_1 \vee \neg b_2) \wedge r_1 \wedge r_2 \wedge \neg i_1 \wedge \neg i_2 ,$$

where:

- c_i s stand for *Causes*
- b_i s stand for *Barriers*
- r_i s stand for *Requirements*
- i_i s stand for *Inhibitors*

For the effect e to happen, all r_i s need to be present (one of them absent can bring down the effect) and all i_i s have to be absent (one of them present can bring down the effect). Any c_i or b_i can cause y but b_i work through their absent state, i.e., their absence can affect e . b_i are just negated c_i s and i_i s are just negated r_i s.

The DeMorgan gate is a canonical interaction model, similar to the Noisy-OR and Noisy-AND gates. In order for the DeMorgan model to be applicable in practice, the following conditions have to be fulfilled (see also Diez & Druzdzel 2006).

- There should be a causal mechanism for each parent such that the parent is able to impact the child variable in the absence of the other anomalies. If co-occurrence of two or more parents is necessary to impact the child, the model may not be suitable. Please note that the impact is different between *Causes/Barriers* and *Requirements/Inhibitors* but each of them is capable of impacting the child variable in separation of other parent variables.
- Are the causal mechanisms independent? This is quite likely the hardest condition to verify, because our knowledge of the domain may not be precise enough to verify that the causal mechanisms in question do not interact with one another. In practice, however, unless there are known interactions, it is not unreasonable to assume that independence of causal influences (also known as the ICI condition) holds and the model can be applied. Should this condition not hold, it is typically

possible to restructure the model in such a way that the ICI condition is approximately satisfied.

- In case of existence of other, unmodeled causes (a non-zero *Leak* parameter), are these unmodeled causes independent of all parents of the effect variable? Typically, we assume that this condition holds unless there is evidence against it. In statistical modeling, this condition is often referred to as "independence of error terms."

We should keep in mind that the above conditions have to be satisfied approximately, as the modeling effort involving QGeNIe graphs is an attempt to approximate reality and the methodology is robust to minor violations of assumptions.

Parameter Elicitation

Of essence to model builders are the questions that are asked of an expert when eliciting the parameter for each link type. These questions have to be clear so as to obtain reliable parametrization.

It is important to realize that the DeMorgan model is an Independence of Causal Influences (ICI) model. A prominent member of the ICI family is the Noisy-OR model (Pearl 1988, Henrion 1989). This means in practice that the parents influence the child independently of each other. Influence of each parent can be specified in separation from the influences of the remaining parents. When specifying the influence of a single parent on the child, it is assumed that each remaining parent is in its "distinguished state." The distinguished state is the state in which the parent has no effect on the child. Because the effects of the four types of parents are different, their distinguished states are also different. In our experience, this leads to considerable confusion among QGeNIe users, so we would like to discuss this on an example and caution the readers to be very careful in selecting the distinguished state of any node during parameter elicitation.

Cause

The distinguished state of a cause is the state in which the cause has no effect on the child. In QGeNIe, it is always the state *False*. For example, *Not being a smoker* has no effect on *Lung cancer* in the example introduced above. *Not being a smoker* is the distinguished state in this interaction. So is *Having no cancer*.

Barrier

The distinguished state of a barrier is also the state in which the cause has no effect on the child. In QGeNIe, it is always the state *True*. For example, regular *Exercise* may be thought as not influencing the risk of *Heart disease*, and so it is the distinguished state in this interaction. So is *No heart disease* in the child node.

Requirement

The distinguished state of a requirement is the state that is necessary for the effect to take place at all. For example, *Being a female* is a requirement for becoming pregnant and it is the distinguished state in this interaction. In QGeNIe, it is always the state *True*. *No pregnancy* is the distinguished state of the child node.

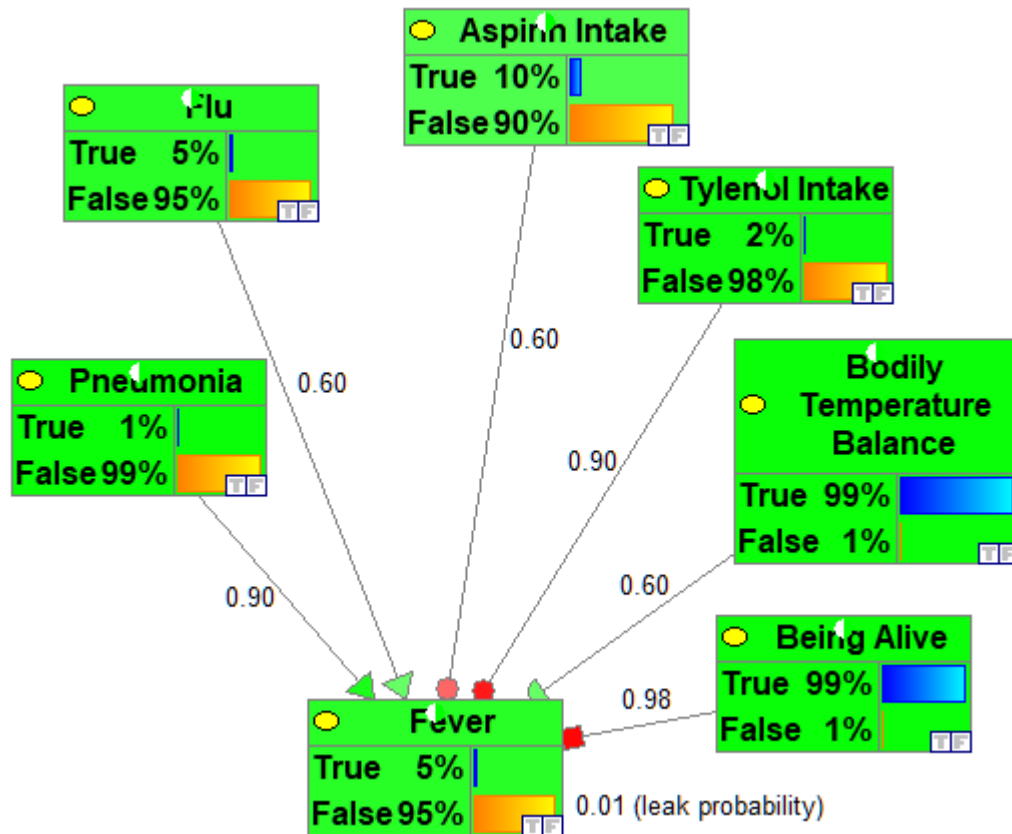
Inhibitor

The distinguished state of an inhibitor is the state that has no effect on the child, i.e., the inhibiting factor being absent. In QGeNIe, it is always the state *False*. For example, *Rain* is an inhibitor of *Wild land fire*. Its distinguished state is *No rain*, which does not influence the wild land fire.

Now, for each type of interaction, the parameter p_i associated with a link from a Cause or a Barrier i corresponds to the probability of the effect e happening if all parents are in their distinguished states (i.e., not acting upon e) and node i is not in its distinguished state. The parameter p_i associated with a link from a Requirement or an Inhibitor i corresponds to the probability of the effect e **not** happening if all parents are in their distinguished states (i.e., not acting upon e) and node i is not in its distinguished state. There is also an additional parameter, called *Leak* which expresses the probability of the effect given that all parents are in their distinguished states. Rather than stating the questions to the expert in the abstract, we explain them based on a simple example model below.

Example 1

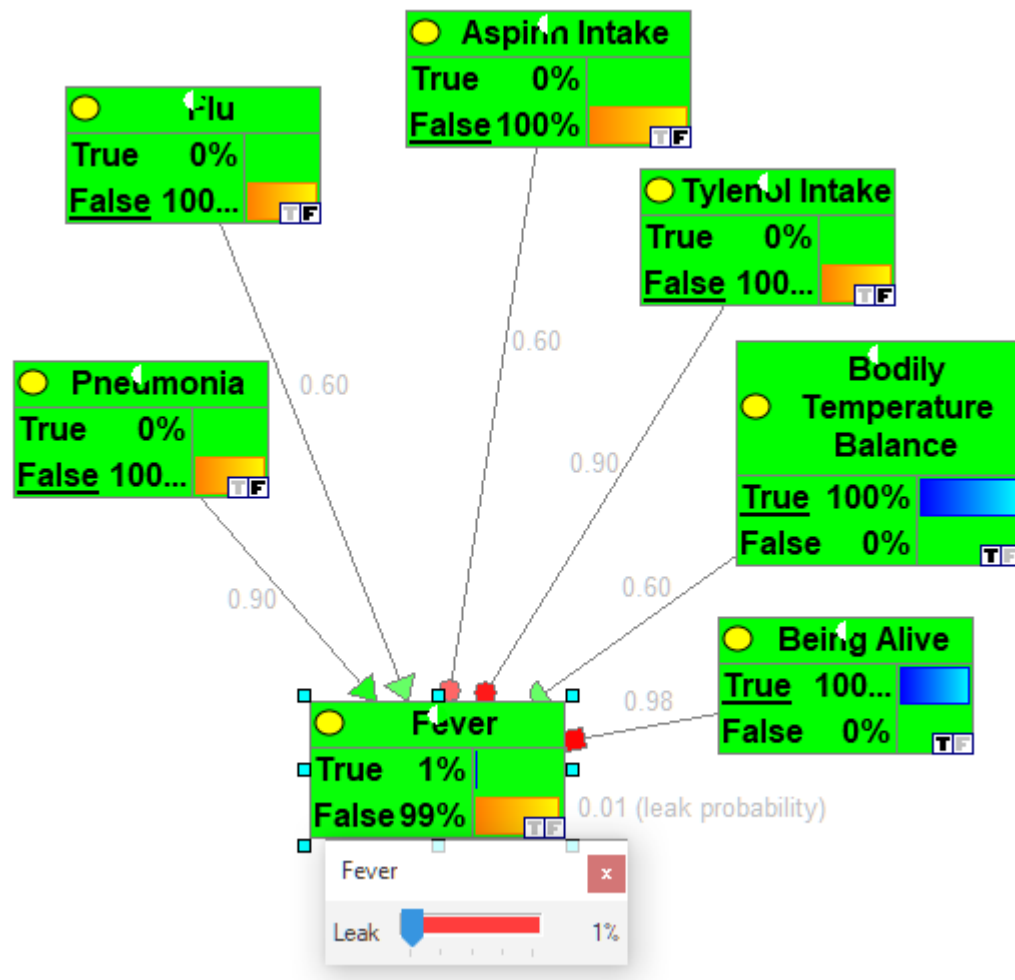
Consider the following network based on the DeMorgan gate with two causes (*Pneumonia* and *Flu*), one barrier (*Bodily Temperature Balance*), one requirement (*Being Alive*), and two inhibitors (*Aspirine Intake* and *Tylenol Intake*):



We list questions for each of the causes in the above model with references to the states of all causes underlined for more clarity.

The "leak" parameter

In order to elicit the leak parameter (0.01 in the above figure), the knowledge engineer has to ask the following question: *What is the probability of fever if an alive patient with bodily temperature balance, has neither pneumonia nor flu, and has taken neither aspirin nor Tylenol?* QGeNIe helps in the elicitation process by dimming the entire model except for the nodes involved and marking the nodes with the states that should be stated in the question.



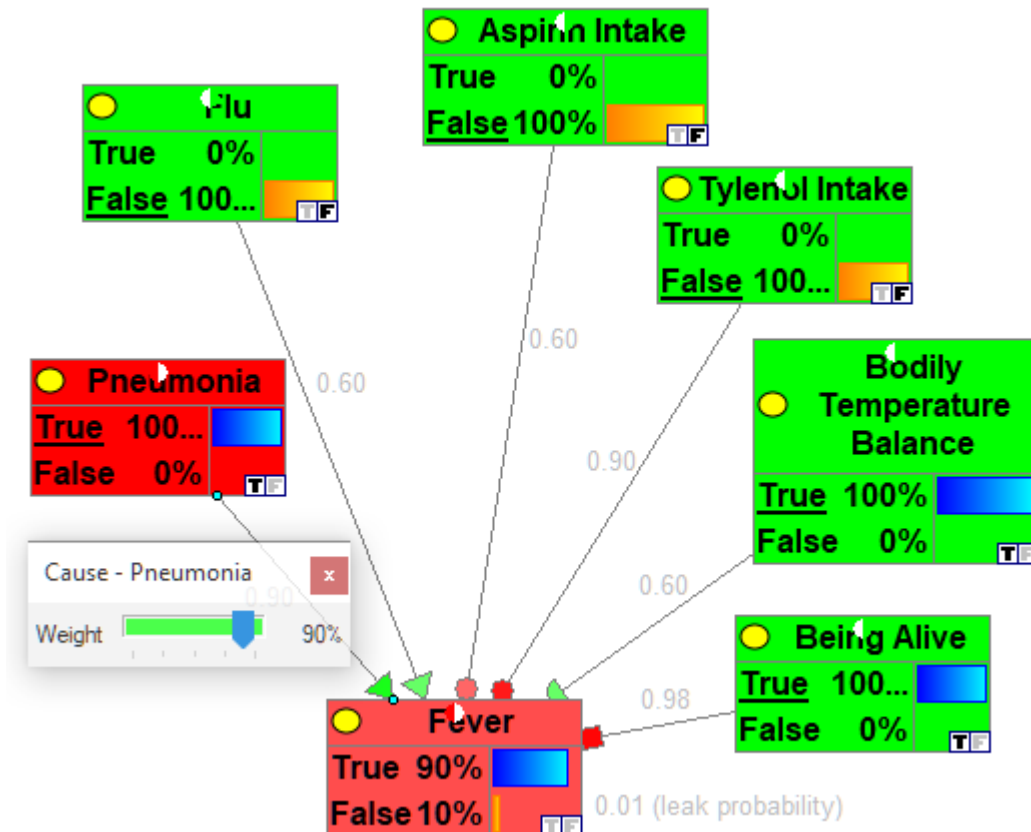
Please note that all causes in the screen shot below are instantiated to F, the barrier is instantiated to T, requirement to T, and inhibitors to F. The conditions in the question include reference to the distinguished states of each of the causes of *Fever*. The leak expresses the probability that fever happens due to other, unmodeled causes.

We list questions for each of the causes in the above model with references to the states of all causes underlined for more clarity.

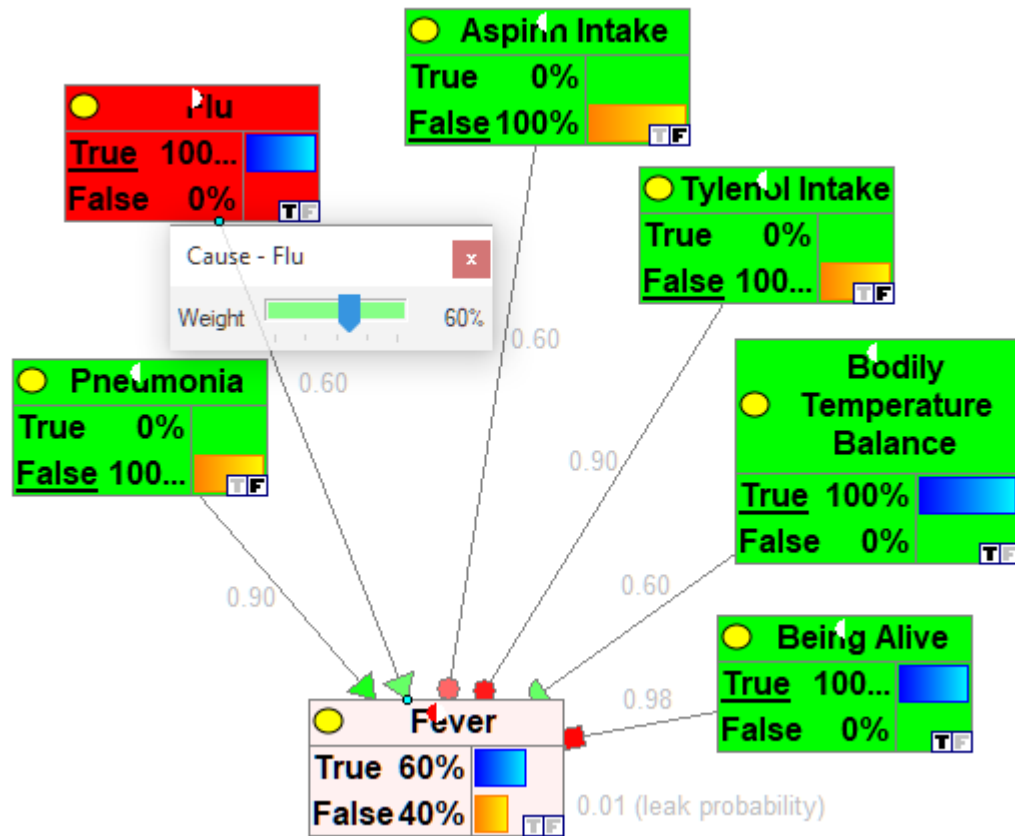
Causes

Questions asking for the causal strength of a cause C of the node *Fever* are framed in such a way that only C is present and all other variables are in their distinguished states.

In order to elicit the strength of the influence of the node *Pneumonia* on *Fever* (0.9 in the figure), the knowledge engineer has to ask the following question: *What is the probability of fever if an alive patient with bodily temperature balance, has pneumonia but no flu, and has taken neither aspirin nor Tylenol and no other unmodeled causal factors that may influence fever are present?* QGeNIe helps in the elicitation process by dimming the entire model except for the nodes involved and marking the nodes with the states that should be stated in the question.



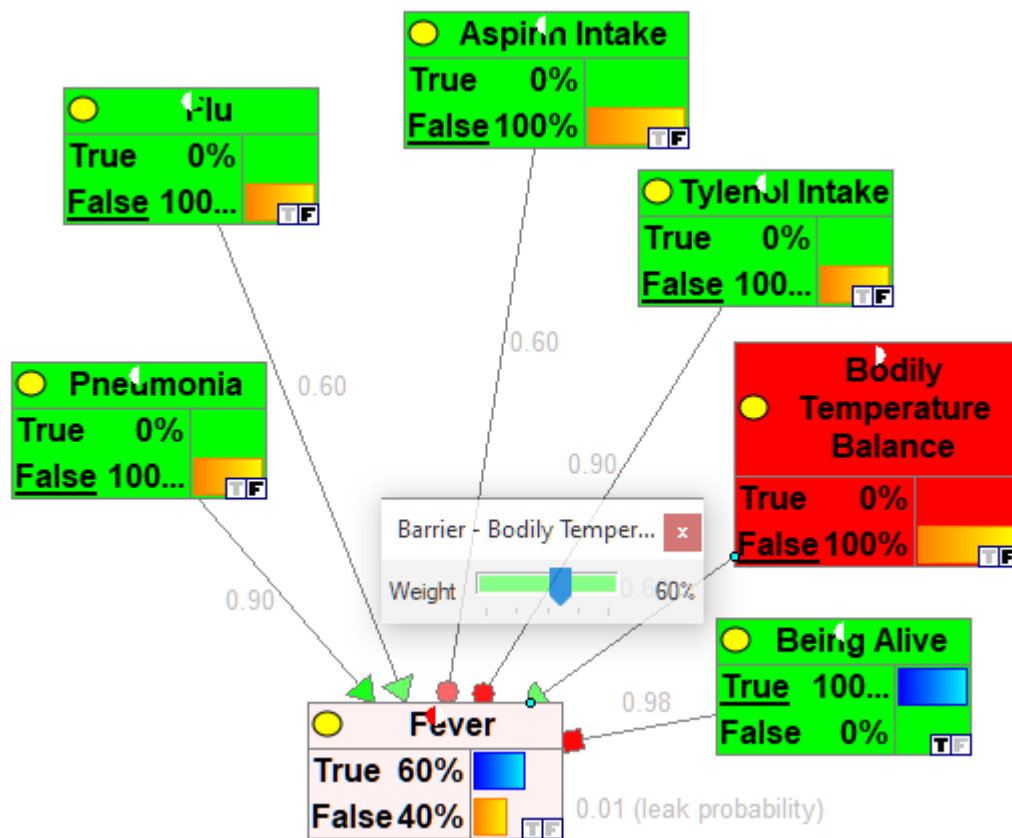
In order to elicit the strength of the influence of the node *Flu* on *Fever* (0.6 in the figure), the knowledge engineer has to ask the following question: *What is the probability of fever if an alive patient with bodily temperature balance, has flu but no pneumonia, and has taken neither aspirin nor Tylenol and no other unmodeled causal factors that may influence fever are present?* QGeNIe helps in the elicitation process by dimming the entire model except for the nodes involved and marking the nodes with the states that should be stated in the question.



Barrier

Questions asking for the causal strength of a barrier B of the node *Fever* is framed in such a way that only B is absent (please note that the distinguished state of a barrier is its presence, which does not impact the effect) and all other causes and barriers are in their distinguished states.

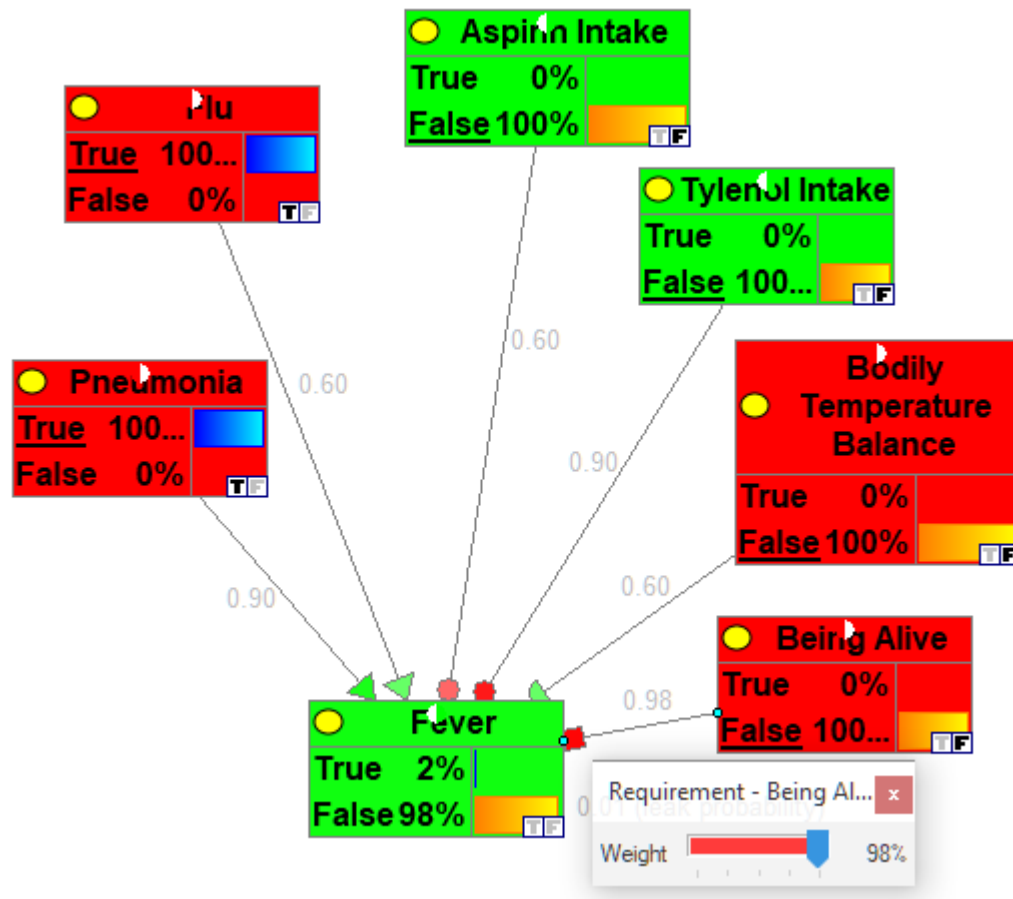
In order to elicit the strength of the influence of the node *Bodily Temperature Balance* on *Fever* (0.6 in the figure), the knowledge engineer has to ask the following question: *What is the probability of fever if an alive patient with abnormal temperature balance, who has neither flu nor pneumonia, and has taken neither aspirin nor Tylenol and no other unmodeled causal factors that may influence fever are present?* QGeNIe helps in the elicitation process by dimming the entire model except for the nodes involved and marking the nodes with the states that should be stated in the question.



Requirement

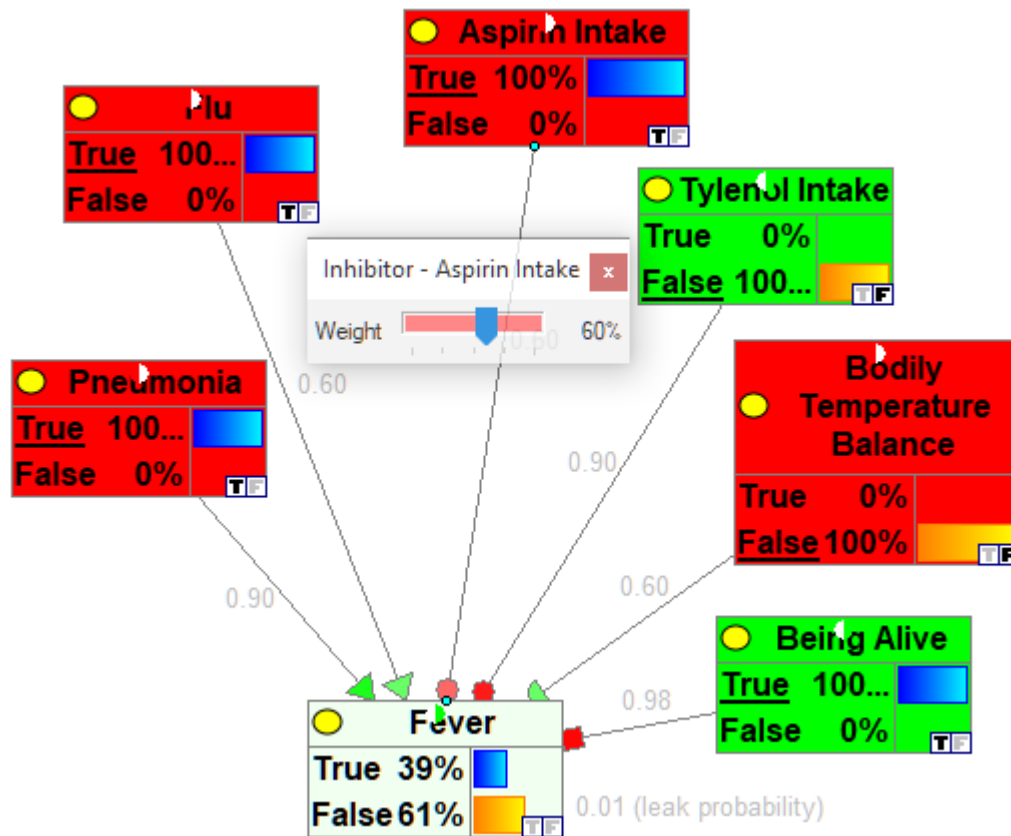
Questions asking for the causal strength of a requirement B of the node *Fever* is framed in such a way that only B is absent (please note that the distinguished state of a barrier is its presence, which does not impact the effect) and all other causes and barriers are in their distinguished states.

In order to elicit the strength of the influence of the node *Being Alive* on *Fever* (0.98 in the figure - please note that we allow for a patient to be just dead, with the body temperature still being high), the knowledge engineer has to ask the following question: *What is the probability of no fever in a dead patient with abnormal temperature balance, who has both flu and pneumonia, and has taken neither aspirin nor Tylenol and no other unmodeled causal factors that may influence fever are present?* Please note that in this case all causes and barriers have to be in active (rather than distinguished) state and the question is asking for the probability of *no fever*! This is just the effect that requirements interact with other causes through the AND function. QGeNIe helps in the elicitation process by dimming the entire model except for the nodes involved and marking the nodes with the states that should be stated in the question.

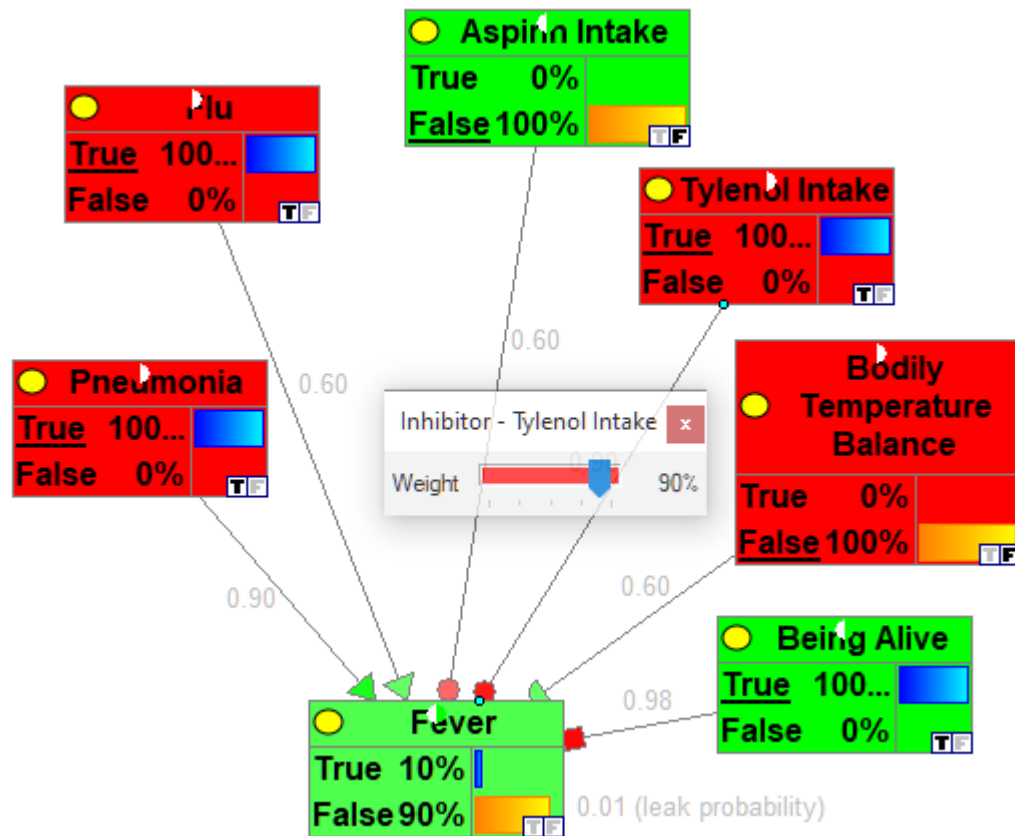


Inhibitors

In order to elicit the strength of the influence of the node *Aspirine Intake* on *Fever* (0.6 in the figure), the knowledge engineer has to ask the following question: *What is the probability of no fever if an alive patient with abnormal temperature balance, who has both flu and pneumonia, and has taken aspirin but not Tylenol and no other unmodeled causal factors that may influence fever are present?* QGeNIe helps in the elicitation process by dimming the entire model except for the nodes involved and marking the nodes with the states that should be stated in the question.



In order to elicit the strength of the influence of the node *Tylenol Intake* on *Fever* (0.9 in the figure), the knowledge engineer has to ask the following question: *What is the probability of no fever if an alive patient with abnormal temperature balance, who has both flu and pneumonia, and has taken Tylenol but not aspirin and no other unmodeled causal factors that may influence fever are present?* QGeNIe helps in the elicitation process by dimming the entire model except for the nodes involved and marking the nodes with the states that should be stated in the question.

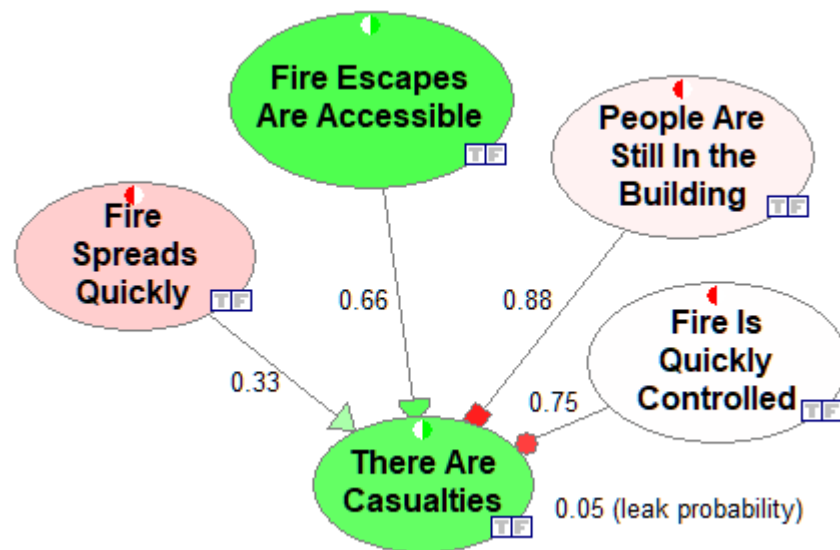


Please note that in both cases all causes and barriers have to be in active (rather than distinguished) state and the question is asking for the probability of *no fever*! This is just the effect that inhibitors interact with other causes through the AND function.

Each of the above questions can be adjusted to the needs of particular context, i.e., things can be rephrased or omitted if they do not make sense.

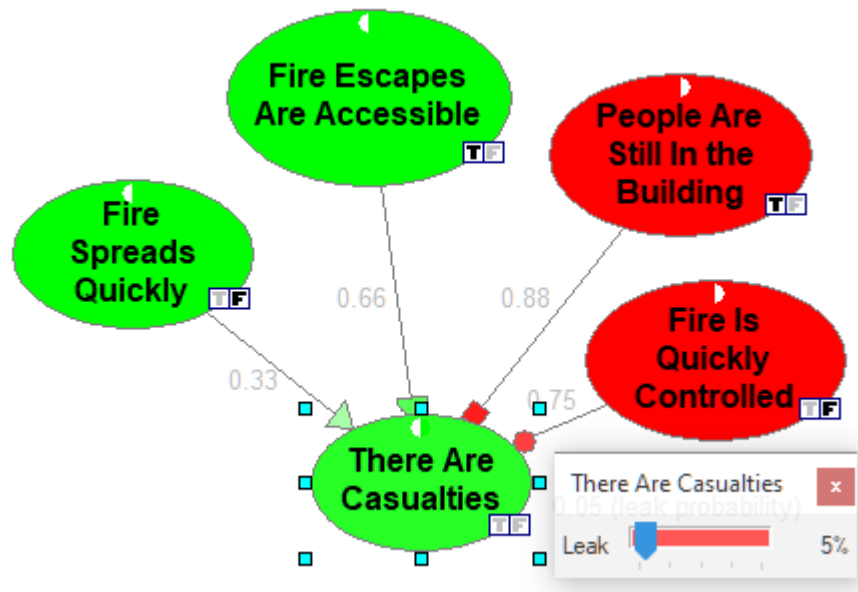
Example 2

Consider the following network based on the DeMorgan gate with one cause (*Fire Spreads Quickly*), one barrier (*Fire Escapes Are Accessible*), one requirement (*People Are Still In the Building*), and one inhibitor (*Fire Is Quickly Controlled*):



The "leak" parameter

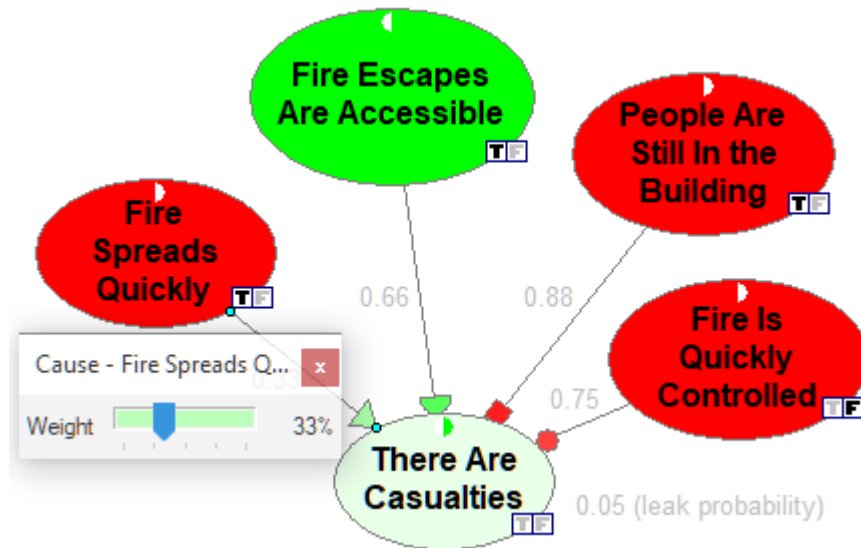
In order to elicit the leak parameter (0.05 in the above figure), the knowledge engineer has to ask the following question: *What is the probability of casualties if the fire does not spread quickly, fire escapes are accessible, people are still in the building, and fire is not quickly controlled?* Please note that the conditions in the question include reference to the distinguished states of each of the causes of *There Are Casualties*. The leak expresses the probability that casualties happen due to other, unmodeled causes. QGeNIe helps in the elicitation process by dimming the entire model except for the nodes involved and marking the nodes with the states that should be stated in the question.



Cause

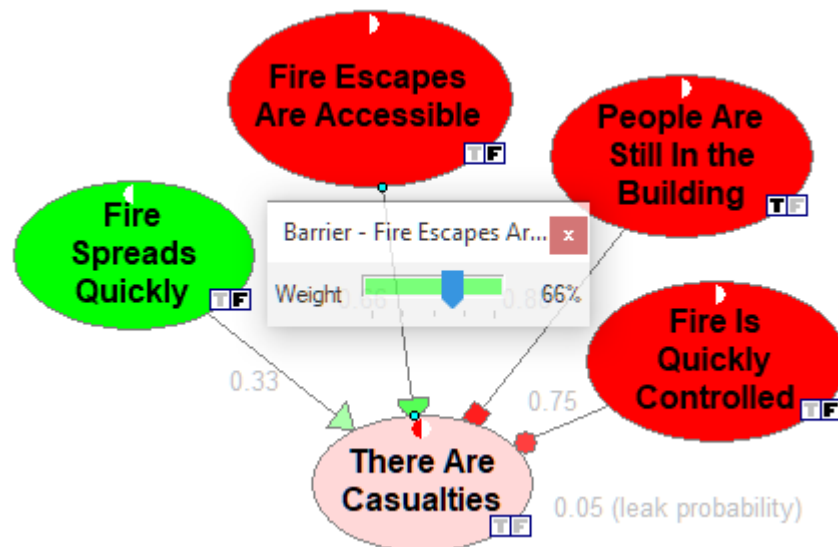
Question asking for the causal strength of a cause *C* of the node *There Are Casualties* is framed in such a way that only *C* is present and all other variables are in their distinguished state.

In order to elicit the strength of the influence of the node *Fire Spreads Quickly* on *There Are Casualties* (0.33 in the figure), the knowledge engineer has to ask the following question: *What is the probability of casualties if the fire spreads quickly, fire escapes are accessible, people are still in the building, fire is not quickly controlled, and no other unmodeled causal factors are present?* QGeNIe helps in the elicitation process by dimming the entire model except for the nodes involved and marking the nodes with the states that should be stated in the question.



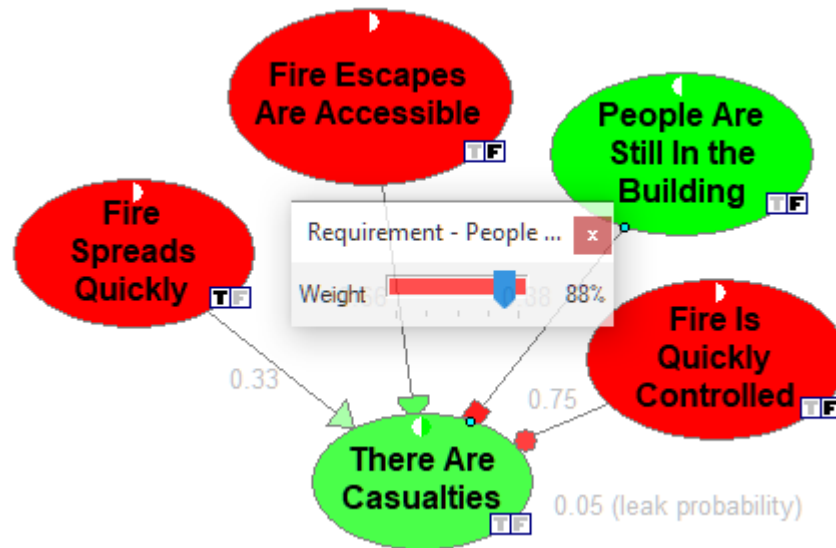
Barrier

In order to elicit the strength of the influence of the node *Fire Escapes Are Accessible* on *There Are Casualties* (0.66 in the figure), the knowledge engineer has to ask the following question: *What is the probability of casualties if the fire does not spread quickly, fire escapes are not accessible, people are still in the building, fire is not quickly controlled, and no other unmodeled causal factors are present?* QGeNIe helps in the elicitation process by dimming the entire model except for the nodes involved and marking the nodes with the states that should be stated in the question.



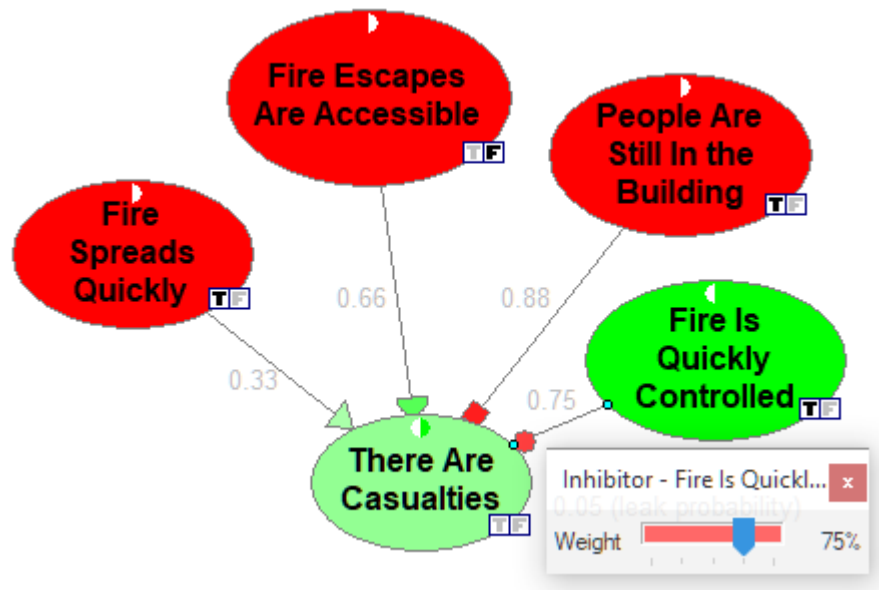
Requirement

In order to elicit the strength of the influence of the node *People Are Still In the Building* on *There Are Casualties* (0.88 in the figure), the knowledge engineer has to ask the following question: *What is the probability of no casualties if the fire spreads quickly, fire escapes are not accessible, there are no people in the building, fire is not quickly controlled, and no other unmodeled causal factors are present?* Please note that in this case all causes and barriers have to be in active (rather than distinguished) state and the question is asking for the probability of *no casualties*! This is just the effect that requirements interact with other causes through the AND function. QGeNIe helps in the elicitation process by dimming the entire model except for the nodes involved and marking the nodes with the states that should be stated in the question.



Inhibitor

In order to elicit the strength of the influence of the node *Fire Is Quickly Controlled* on *There Are Casualties* (0.75 in the figure), the knowledge engineer has to ask the following question: *What is the probability of no casualties if the fire spreads quickly, fire escapes are not accessible, there are people in the building, fire is quickly controlled, and no other unmodeled causal factors are present?* Please note that in this case all causes and barriers have to be in active (rather than distinguished) state and the question is asking for the probability of *no casualties*! This is just the effect that requirements interact with other causes through the AND function. QGeNIe helps in the elicitation process by dimming the entire model except for the nodes involved and marking the nodes with the states that should be stated in the question.



Each of the above questions can be adjusted to the needs of particular context, i.e., things can be rephrased or omitted if they do not make sense.

5.3.3 The CAST gate

In addition to the DeMorgan gate, QGeNIe implements another interaction model: the CAST gate (Chang 1994). CAST stands for CAusal STrengths logic. We have attempted to find a probabilistic interpretation of the CAST gate and we think that it is challenging with non-obvious meaning of its parameters. Still, CAST gate has been used widely, especially in military applications, through two programs popular in 1990s, SIAM and Causeway, both apparently no longer available. While interesting and useful in practice, we believe that in the context of probabilistic models, the CAST gate is semantically less clear than the DeMorgan gate and we found that it sometimes causes problems with knowledge elicitation from experts, especially around CAST's *baseline probability* (discussed below). We have implemented it in QGeNIe for the sake of completeness and compatibility with existing models and software, although we do not allow for combining DeMorgan and CAST gates in the same model, so the user has to choose whether he/she is modeling using DeMorgan gates or the CAST gates.

We focus in this section on the CAST gate, providing definitions and descriptions of the CAST parameters. Because this section is necessarily brief, we refer interested readers to the paper by Chang (1994).

Baseline probability

Similarly to the DeMorgan gate, every CAST node is characterized by a parameter called *baseline probability*. For nodes without parents, this parameter has a fairly clear probabilistic interpretation and is simply the prior probability of the proposition represented by the node.

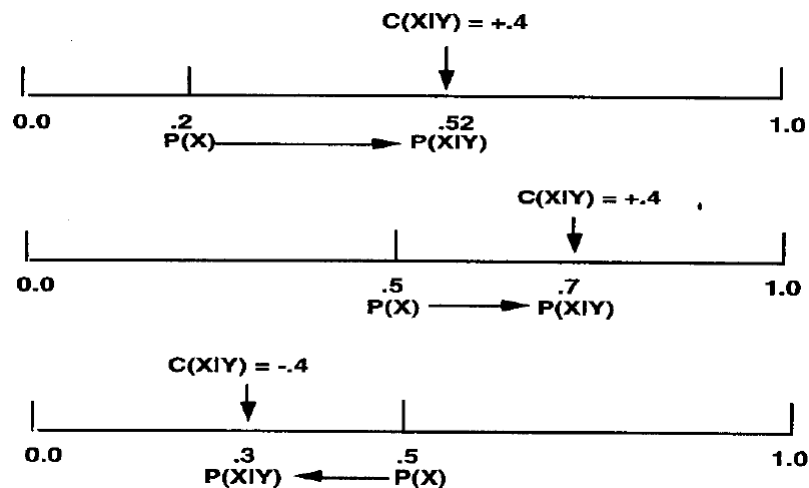
However, when the node has parents, the baseline probability can no longer be interpreted as the prior probability of the proposition. It cannot be given the interpretation of a leak parameter. Only jointly with a specification of the influence of parents, causal strengths (described in the next paragraph), it is possible to derive conditional probability distributions over the node, something that a probabilistic model needs.

It would be fair to state that the baseline probability $P(X)$ is the probability of a proposition X when it is isolated from influences of any other modeled variables, so it is close to the marginal probability of X .

Causal strengths

Every arc connecting a pair of CAST nodes is described by two parameters, which are called *causal strengths*. Causal strengths take values from the interval $[-1..1]$. They are not probabilities but rather an expression of fractional change in the baseline probability of the child node as an effect of the parent node being *True* or *False* respectively. Because they can take both positive and negative values, they are convenient in describing positive or negative character of influences.

The following figure from (Chang 1994, Figure 2.3) illustrates the meaning of causal strengths associated with an influence of a variable Y on a variable X :



Causal strength $C(X|Y)$ expresses the percentage of change of the baseline probability toward the extremes (0 and 1). When positive, its value tells us the percentage of change from the current probability p toward the probability 1.0. When negative, it expresses the percentage of change from the current probability p toward the probability 0.0. The top part of the figure shows what happens when the baseline $P(X)=0.2$ and $C(X|Y)=0.4$. In this case, the probability of X changes from its original value of 0.2 to $0.2+(1.0-0.2)*0.4=0.52$. The middle part of the figure shows what happens when the baseline $P(X)=0.5$. In this case, the probability of X changes from its original value of 0.5 to $0.5+(1.0-0.5)*0.4=0.7$. Finally, the bottom part of the figure shows what happens when the baseline $P(X)=0.5$ gets affected by a negative causal strength $C(X|Y)=-0.4$. In this case, the probability of X changes from its original value of 0.5 to $0.5-(0.5-0.0)*0.4=0.3$.


Aggregation of causal strengths

Causal strengths, similarly to parameters in the DeMorgan gate, describe individual influences that neighboring nodes have on each other. This allows for a concise description of how a node is effected by its parents. When multiple parents are present, the CAST logic determines how their influences are combined with each other. Combining positive influences is quite straightforward and so is combining negative influences. Calculation of a mixture of positive and negative influences is more problematic, although resting on reasonable heuristic assumptions.

5.3.4 Submodels

QGeNIe allows for placing groups of nodes into submodels. Submodels are special types of nodes that host sub-graphs of the entire graph and make the [Graph View](#) structured hierarchically. Submodeling facilitates modularity in large models. We advise to make a generous use of submodels in case your models become larger than, say 50 variables. A large number of variables may not fit on the screen and may make the interaction with the model cumbersome. The internals of a submodel, along with its structure can be examined in separation from the entire model.

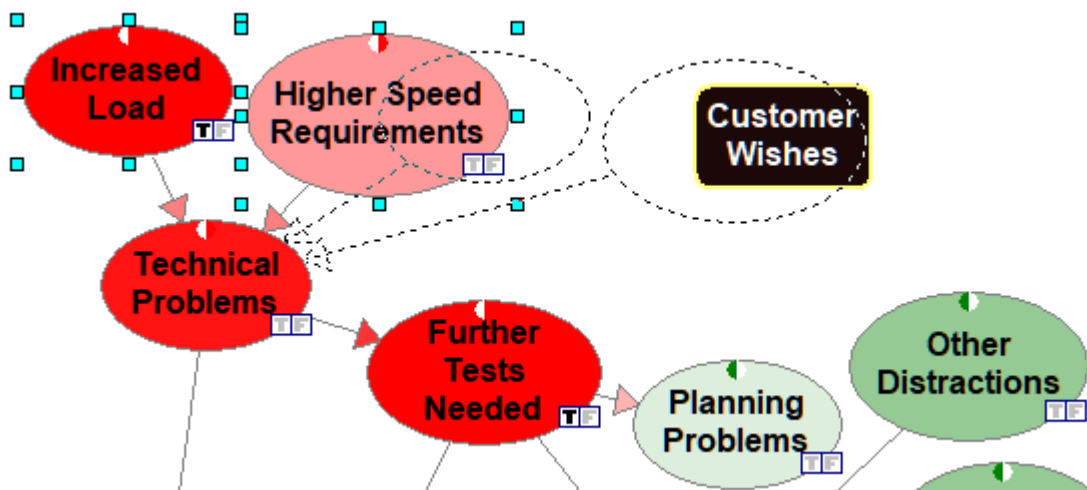
Creation of submodels, moving nodes, navigating through submodels

To create a submodel in QGeNIe, select *Submodel* from the [Tool Menu](#) or the *Submodel* () tool from the [Standard Toolbar](#) and click on the *Graph View*. You will see a new submodel.

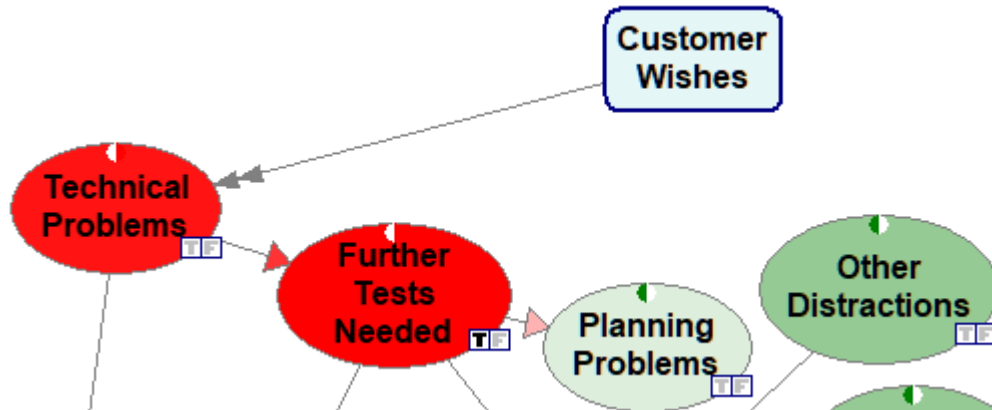


Submodel windows can be opened by double-clicking on the *Submodel* icon or right-clicking on the submodel icon and choosing *Open Submodel* from the *Submodel properties* menu.

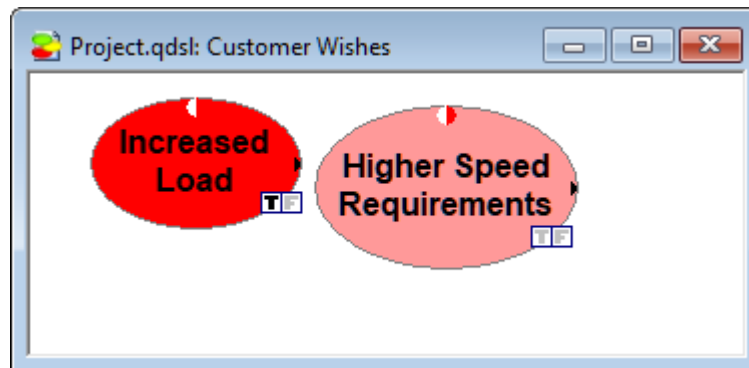
Nodes can be moved between submodels by selecting them in the source submodel, dragging, and dropping them in the destination submodel. For example, we might want to create a submodel for the variables *Increased Load* and *Higher Speed Requirements* in the following model. We do this by creating a submodel node, renaming it to *Customer Wishes*, and then dragging and dropping the nodes *Increased Load* and *Higher Speed Requirements* to the new submodel.



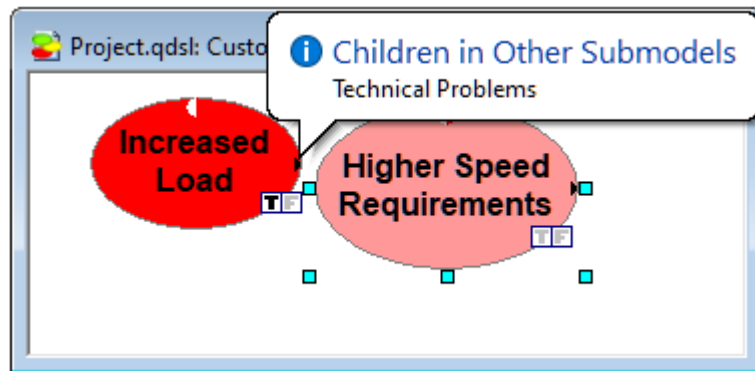
The resulting model will look as follows:



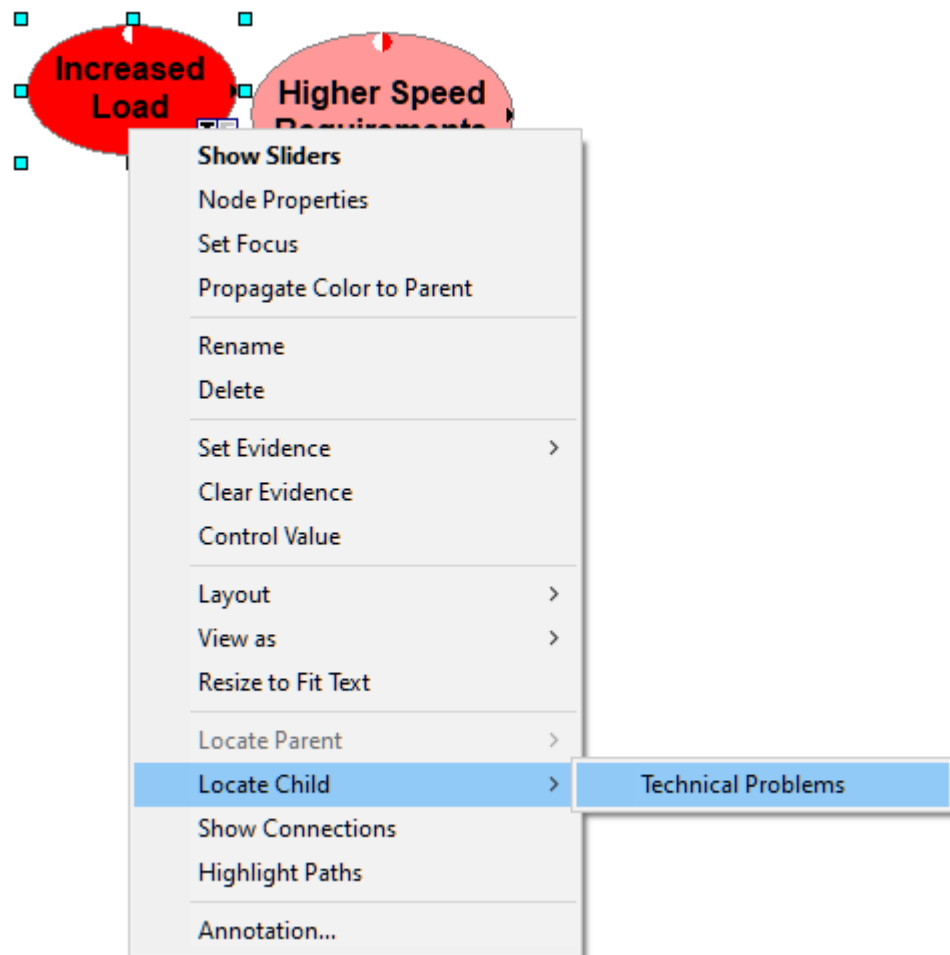
Submodels are opened by double-clicking on them. Double-clicking on the submodel *Customer Wishes* yields the following:



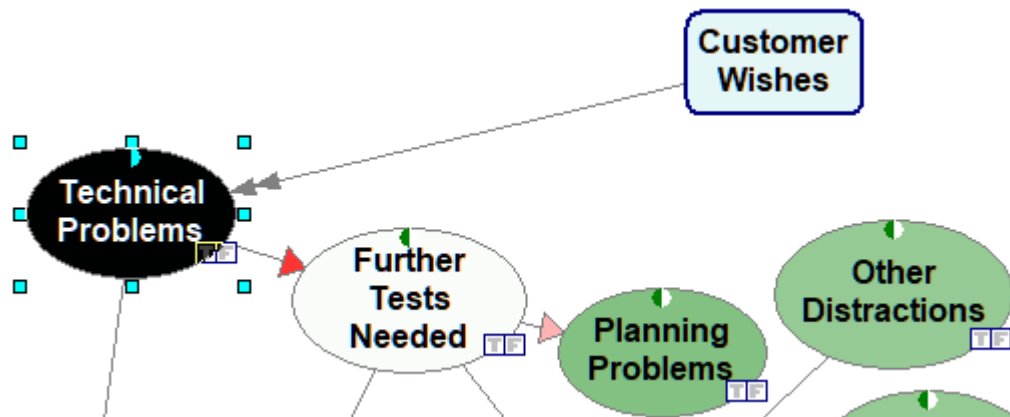
One thing that becomes less clear in submodels is the connections that a submodel has with the external world. QGeNIe does not normally show arcs that are coming from outside or that go to the outside world. It does let the user know that there are such connections. First of all, by showing these connections as coming into the submodel node (note the arcs from the submodel node *Customer Wishes* coming into the node *Technical Problems* at the main model level). It also adds small triangle-shaped marks on the left and right sides of the internal submodel nodes showing that there are incoming and outgoing arcs respectively. The user can examine these connections by placing the cursor over the small triangle. This will display the name of the child of the node in another submodel as follows:



You can locate the child of this node by right clicking and choosing *Locate Child* from the *Node Pop-up* menu:



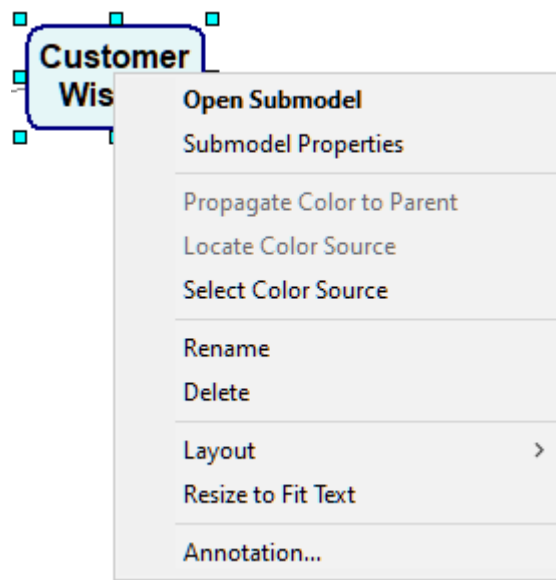
Alternatively, right-clicking on the small triangle on the right-hand-side shows a pop-up menu with a list of children of the node in other submodels. Both actions lead to finding the node and flashing it on the screen as shown below:



It is possible to add arcs between nodes that are located in different submodels in the very same way that arcs are added between nodes in the same submodel. When more than one arc is drawn between a submodel and a node, then QGeNIe draws a double arrow arc from the submodel to the node as shown above. All the above functions can be also performed through QGeNIe *Tree View*.

Submodel properties

Submodel properties sheet can be displayed by right clicking on the name of the submodel in the [Tree View](#) or right clicking on the submodel icon in the [Graph View](#). This will display the *Submodel Pop-up* menu. Select *Submodel Properties* from the menu.



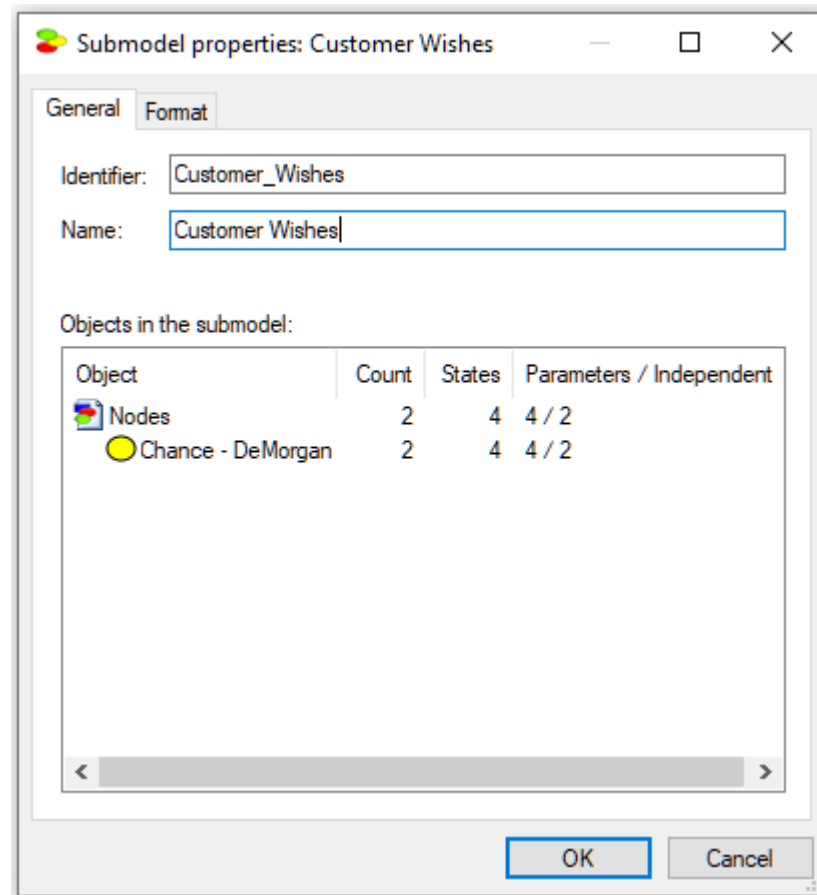
Note : Double clicking on the submodel will open the graph view of the submodel, it will not open the *Submodel properties* sheet.

The *Submodel properties* sheet consist of two tabs: *General* and *Format*. The *General* tab allows to change the identifier and the name of the submodel, the *Format* sheet allows to change the graphical

properties of the submodel icon and is identical to the property sheet described in node property sheets.

General tab

The *General* tab displays the *Identifier* and the *Name* of the submodel, along with the submodel's basic statistics.



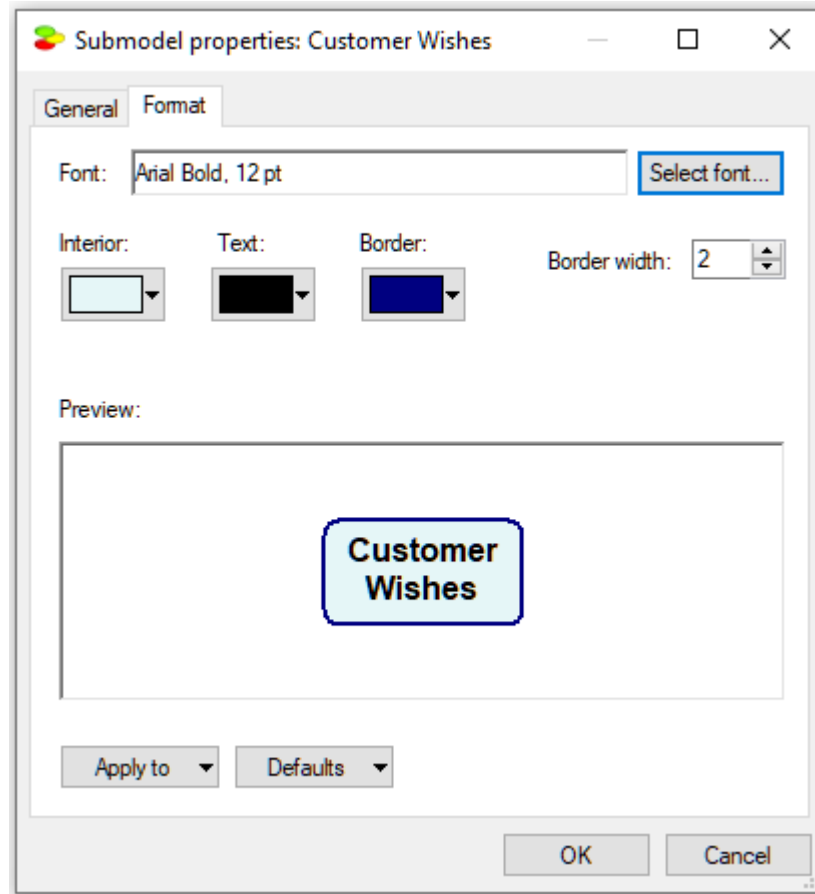
Identifier displays the identifier for the submodel, which can be modified by the user. Identifiers must start with a letter, and can contain letters, digits, and underscore (`_`) characters. The identifier for the network shown above is *Customer_Wishes*.

Name displays the name for the submodel, which is specified by the user when the submodel is first created. There are no limitations on the characters that can be part of the name. The name for the network shown above is *Customer Wishes*.

The *Objects in the submodel* lists counts of various types of objects and numerical parameters in the submodel. They give an idea of the submodel's complexity.

Format tab

The *Format* tab allows to modify the visual properties of the submodel icon, i.e., how the submodel icon is displayed in the *Graph View*.



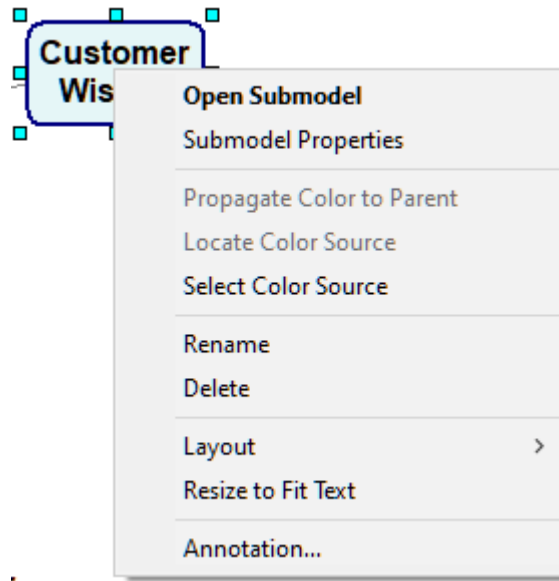
The *Format* tab allows for specifying the physical appearance of the submodel node in the *Graph View*. The user can specify the font, its size, border width, the color of the text, border, and interior of the submodel node.

Other submodel operations

Submodel Popup menu is slightly different for the *Graph View* and the *Tree View*.

Submodel Pop-up menu for the Graph View

The *Submodel Pop-up* menu for the [Graph View](#) can be displayed by right clicking on the submodel icon in the [Graph View](#).



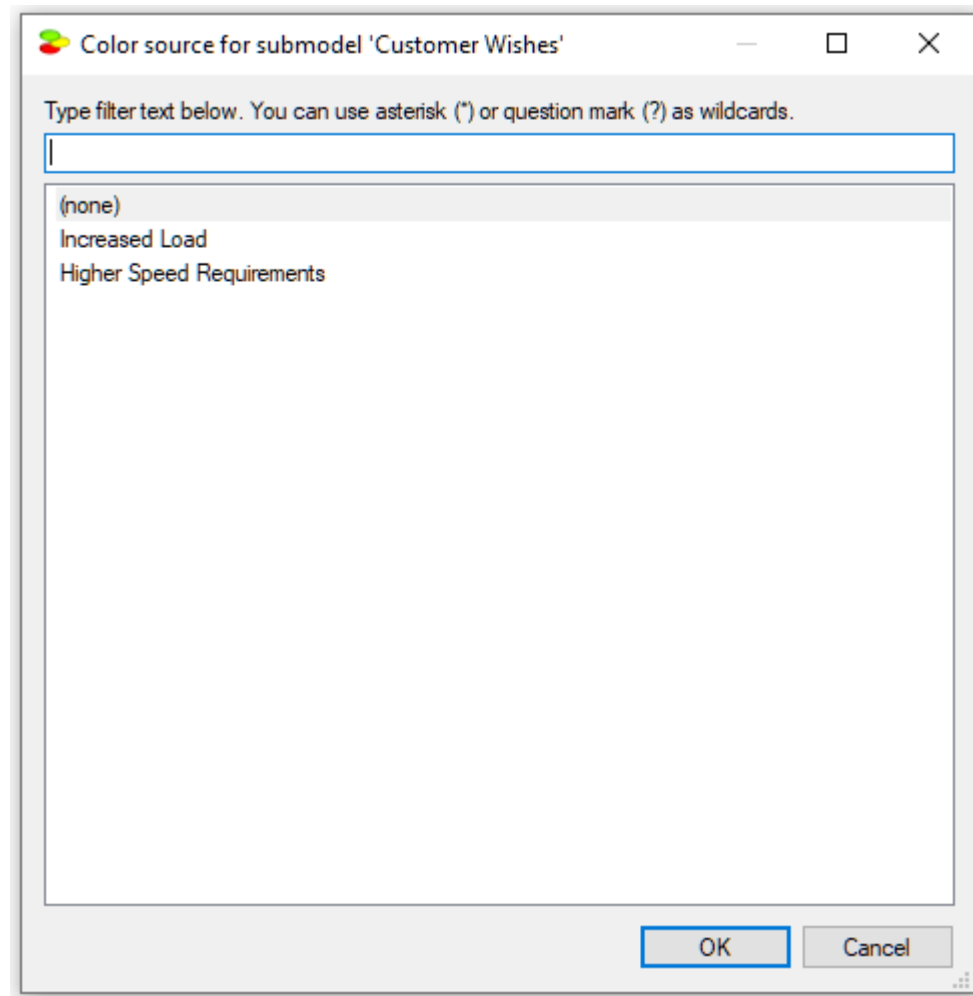
Open Submodel opens the submodel in a new [Graph View](#) window.

Submodel Properties opens the *Submodel Properties* sheet.

Propagate Color to Parent allows for passing the color of the node (this includes a submodel) icon to the parent submodel icon. This is useful when we want to summarize the effect of a submodel in one node to which we pass the color of a selected outcome node from inside the submodel.

Locate Color Source is an operation opposite to *Propagate Color to Parent* and it allows to find the node that is passing its color to the current submodel icon.

Select Color Source is an operation symmetric *Propagate Color to Parent* and it allows to find the node that will pass its color to the current submodel icon. When selected, the following dialog pops up.



The list of nodes in the dialog contains all nodes inside the submodel. The can be searched using wildcard characters of * and ?.

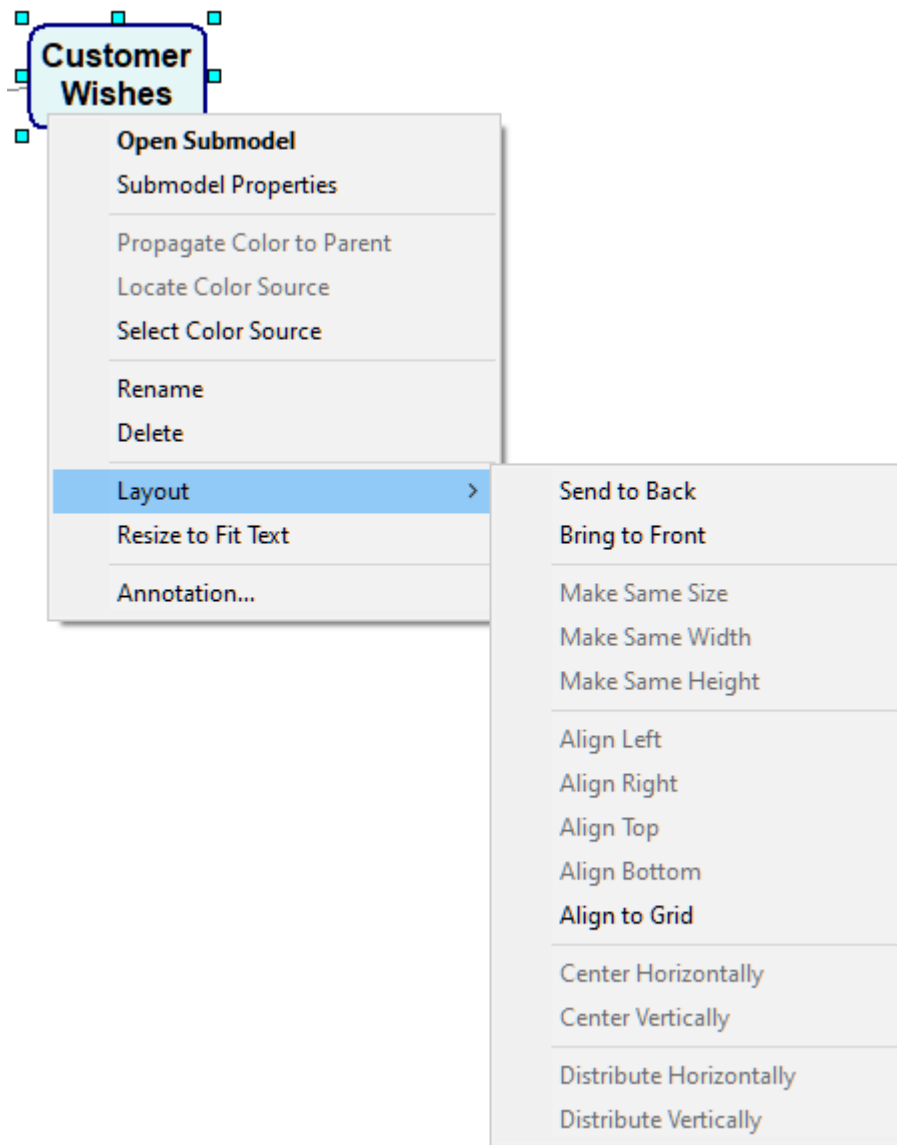
Rename allows you to rename the submodel by placing the submodel icon in edit mode. You can also rename a submodel by modifying the *Name* field in *Submodel Properties* sheet.

Delete deletes the selected submodel.

Resize to Fit Text resizes the submodel icon so that it fits the entire submodel name.

Annotation... opens up the annotation dialog so that you can add an annotation to the submodel (see [Annotations](#) section for more information).

Layout submenu



Most of the commands on the *Layout* submenu are the same as those in the [Layout Menu](#). The only commands here that are not found in the *Layout Menu* are:

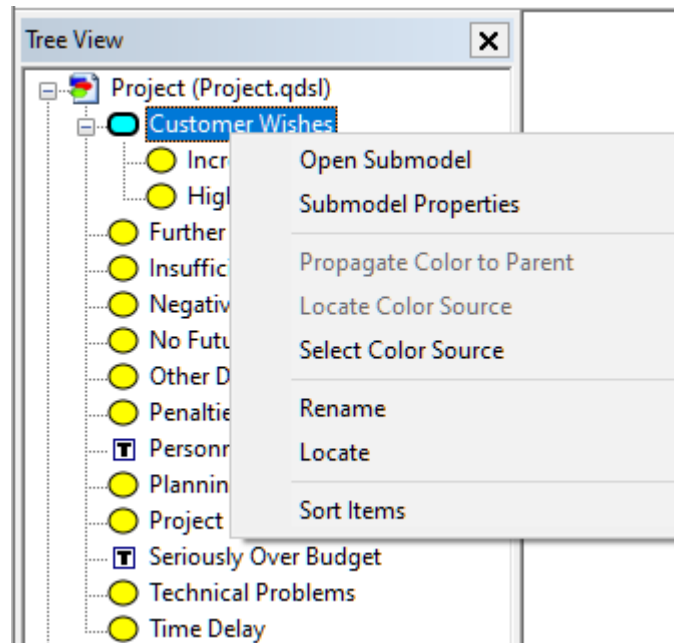
Make Same Size (enabled only if two or more items are selected in the *Graph View*) resizes the selected items so that they are the same size as the item that was right clicked.

Make Same Width (enabled only if two or more items are selected in the *Graph View*) resizes the selected items so that they are the same width as the item that was right clicked.

Make Same Height (enabled only if two or more items are selected in the *Graph View*) resizes the selected items so that they are the same height as the item that was right clicked.

Submodel Popup menu for the *Tree View*:

The *Submodel* pop-up menu for the [Tree View](#) can be displayed by right clicking on the submodel name.



The two commands that are not available in the [Graph View](#) are:

Locate locates the submodel in the [Graph View](#) of its parent model or submodel. Once located, the submodel icon flashes several times on the screen to attract user attention.

Sort Items sorts the list of nodes or submodels of the current submodel listed in the tree view in alphabetical order.

5.3.5 Arcs

An arc in a [Bayesian network](#) denotes an influence, i.e., the fact that the node at the tail of the arc influences the value (or the probability distribution over the possible values) of the node at the head of the arc.

One remark about editing models is that QGeNIe does not allow moving arcs between nodes, i.e., it is not possible to select and drag the head or the tail of an arc from one node to another. If this is what you want, the way to accomplish this task is to first delete the original arc and then create a new arc.

QGeNIe displays also arcs between nodes and submodels. An arc from a node N to a sub-model S means that at least one node in S depends on N . An arc from a sub-model S to a node N means that N depends on at least one node in S . An arc from a sub-model S_1 to a sub-model S_2 means that there is at least one node in S_2 that depends on at least one node in S_1 . Arcs between sub-models can be double-headed, in which case the relations listed above is reciprocal. For example, a double-headed arrow between a node N and a sub-model S means that there is at least one node in S that depends on N and that there is at least one node in S that influences N . QGeNIe does not show arcs that are coming from

the outside of the current sub-model window. Existence of arcs coming from outside of the current sub-model and ending in a node in the current sub-model is marked by a small triangle on the left-hand side of the node. Existence of arcs originating in a node in the current sub-model and ending in a higher-level sub-models is marked by a small triangle on the right-hand side of the node. These links can be followed by right-clicking on the small triangles.

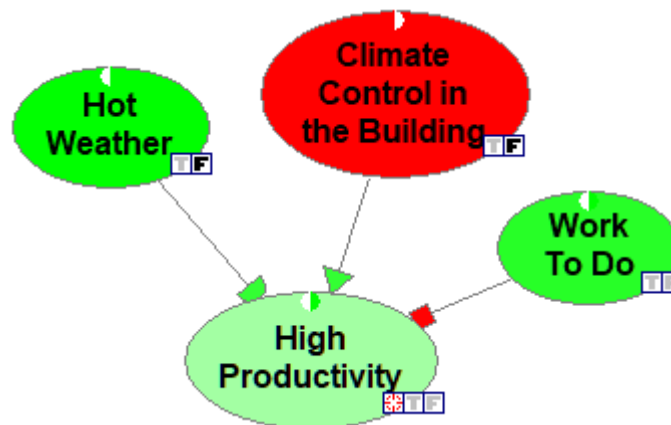
Cycles in the graph, i.e., directed paths that start and end at the same point, are forbidden (unless the graph is dynamic, such as a Dynamic Bayesian Network, which is covered in a separate section). QGeNIe will not allow you to draw cyclic graphs. Please note that even though QGeNIe will enforce that the underlying graph is acyclic, you may still be able to observe cyclic graphs involving submodel nodes. This is due to the meaning assigned to arcs between submodels.

5.3.6 Node status icons


Each node in the *Graph View* is marked by one or more *node status icons*. These are tiny icons displayed in the lower right corner of the node icon. There are four different node status icons: *Observed*, *Implied*, *Controlled* and *Focus*. We will explain their meaning on simple examples.

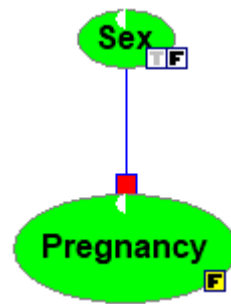
Observed Status Icon

The *Observed* status icon (e.g., ) is displayed bold when the user enters evidence into a node and it signifies that the node is an evidence node. Nodes *Hot Weather* and *Climate Control in the Building* in the following model are evidence nodes and are marked with the *Observed* status icons:



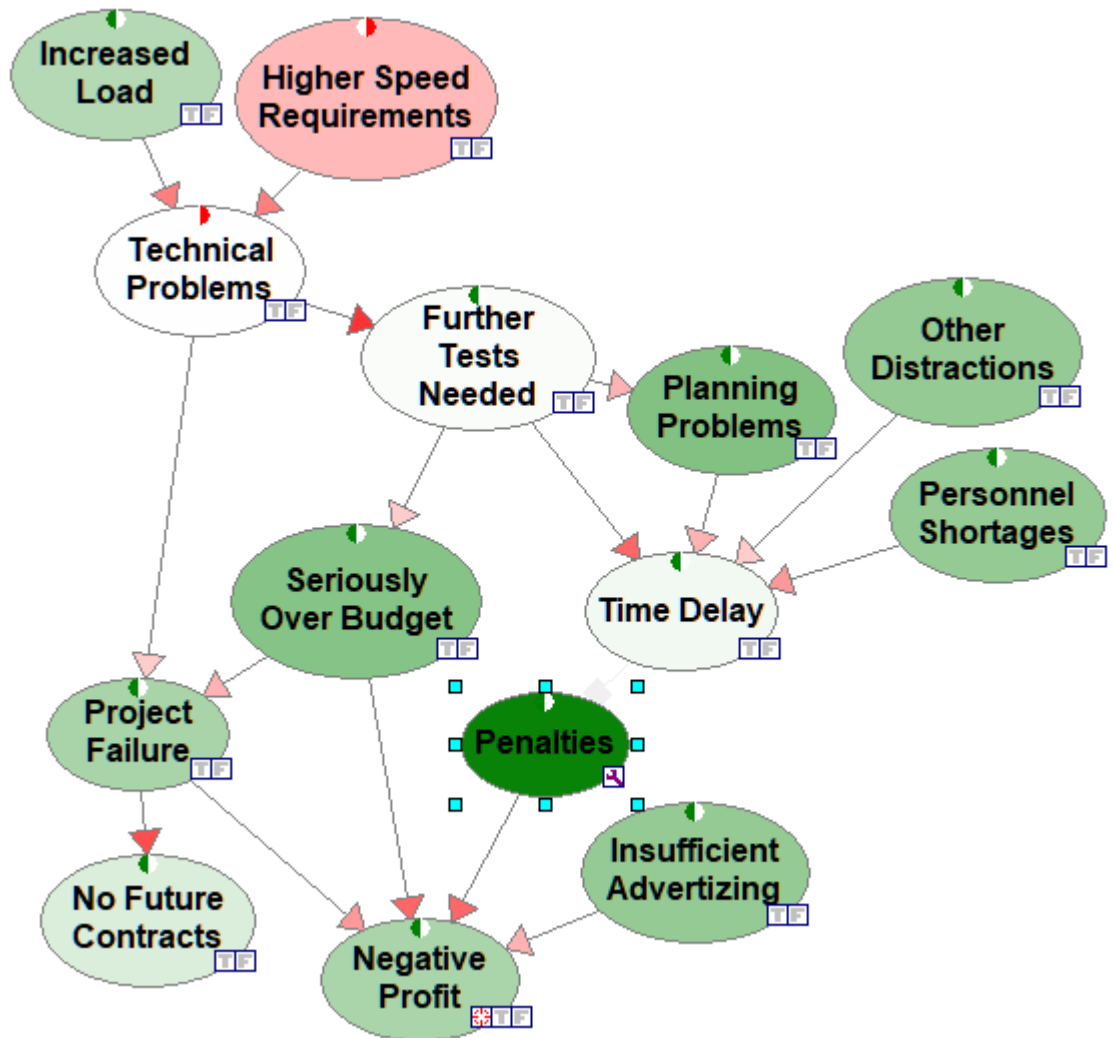
Implied Status Icon

Sometimes, observing a node implies the values of other nodes. For example, observing that a patient is male in a medical decision support system will imply that he is not pregnant. This is possible because the program realizes that for a male patient pregnancy is impossible. QGeNIe marks the nodes whose values are implied by observations of other nodes by the *Implied* status icons. The *Implied* status icon is identical to the *Observed* status icon, but have yellow background (e.g., ). In the following example, the *Sex* variable has been observed to be *False* and the value of the *Pregnancy* variable has been determined by GeNIe to be *False*.




Controlled Status Icon

Controlling the value of a node is different from observing it (see [Changes in structure](#) section). When a node is controlled, QGeNIe marks it with the *Controlled* status icon (🔧). Node *Penalties* in the model below has been manipulated and marked as *Controlled*.




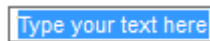
Focus Status Icon

The *Focus* status icon () has to do with QGeNIe's support for calculations of the value of information and value of manipulation. Node *Negative Profit* in the above model has been designated as *Focus*.

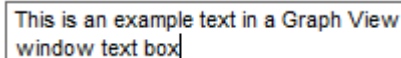
5.3.7 Text boxes

You can put an arbitrary text in the *Graph View* window. This text may be useful as a comment explaining the details of the model. To add a text you need to create a *Text box*.

Select the *Text box* () button from the toolbar or *Tools* menu (note that the cursor shape changes). The *Text box* button will become recessed. Move the mouse to a clear portion of the [Graph View](#) and click the left mouse button. You will see a rectangle appear on the screen:



You can type any text inside the box. You can use any of the regular editing tools, such as cursor movement, selection, *Cut*, *Copy*, and *Paste*. After you have typed your text, press *Enter* or click anywhere outside of the box. The box may look as follows:

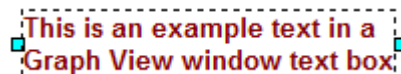


You can change the font type, style, size, and color using appropriate tools from the [Format Toolbar](#). Shown below are some effects of using the [Format Toolbar](#):

**This is an example text in a
Graph View window text box**

You can always come back to editing the text in the box by double-clicking on the box. Double-clicking makes the text visible for editing again.

You can select the box by single-clicking on it. A selected box shows its boundaries and two small squares at its left and right boundary.



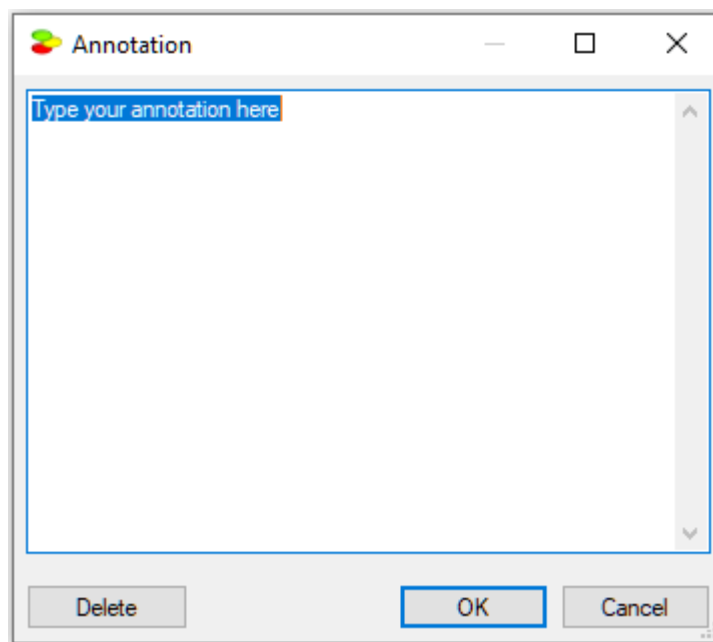
You can re-size a selected box by dragging on one of these small squares. You can delete it by pressing the *Delete* key. You can drag the box to a new location by clicking on it and holding the mouse button down while moving it to a new location.

5.3.8 Annotations

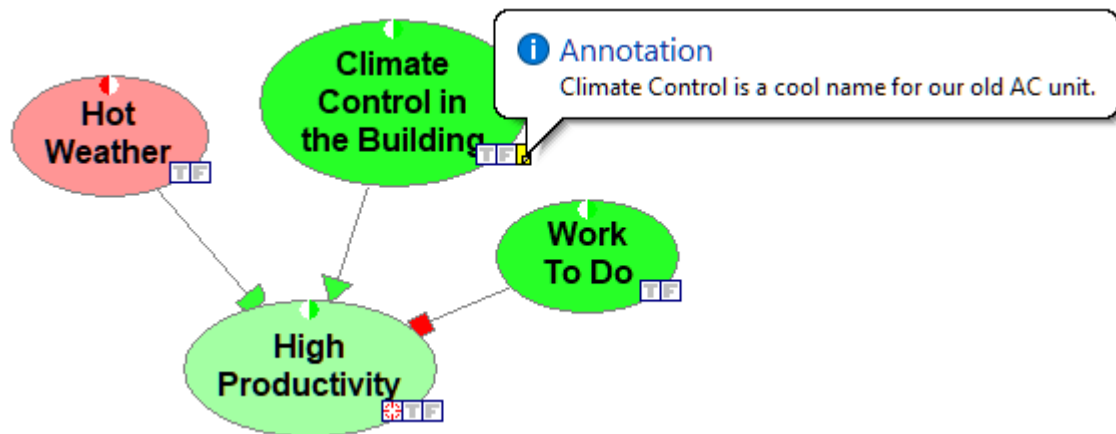
QGeNIe supports annotations for nodes and states of nodes. Following the idea that one of the main goals of a model is documenting the decision making process, annotations are useful for explaining function of nodes and states, or to note down just about anything the user feels is important regarding the node or state.

Annotations for nodes

You can specify annotations for nodes by right clicking on the node and selecting *Annotation* from the *Node Pop-up* menu. This will display the annotation box as shown below:



Enter the annotation in the white blank space and click on *OK* to save it. Once an annotation has been saved against a node, QGeNIe displays a small yellow note (📝) beside the status icon of the node. To view the annotation for a node, hover the cursor over the note. QGeNIe will display the annotation as follows:

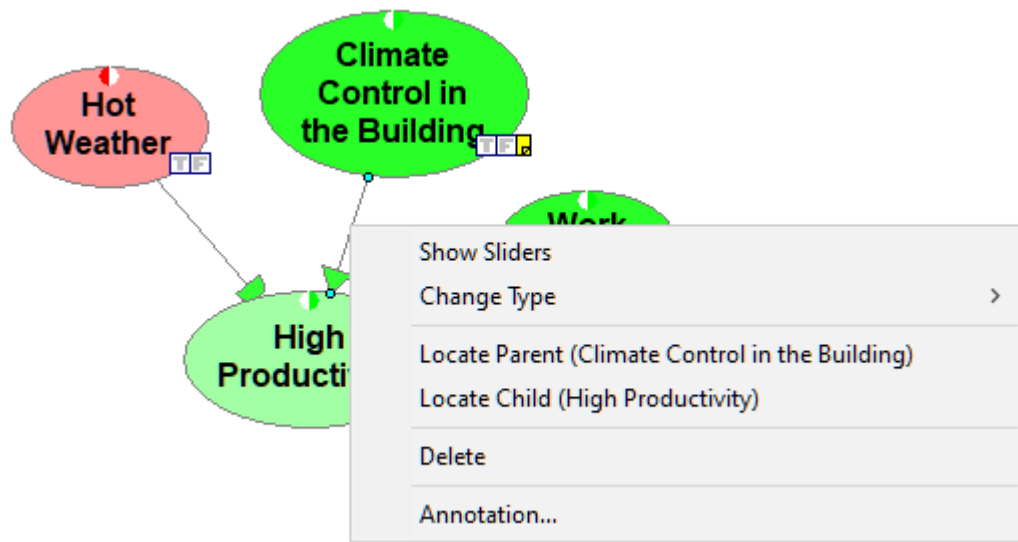


Double click on the note to display the annotation box, which will allow you to edit the annotation.

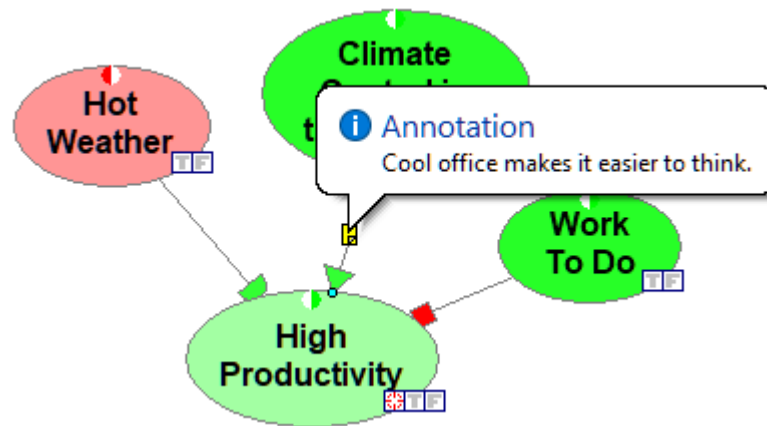
To delete an annotation, double click on the note and delete all contents of the annotation box.

Annotations for arcs

You can annotate arcs between nodes as well. To annotate an arc, right click on it and choose *Annotate* from the pop-up menu that shows:



Choosing *Annotate* brings up the annotation window. Hovering over the yellow stick-it-note shows the text of the annotation.

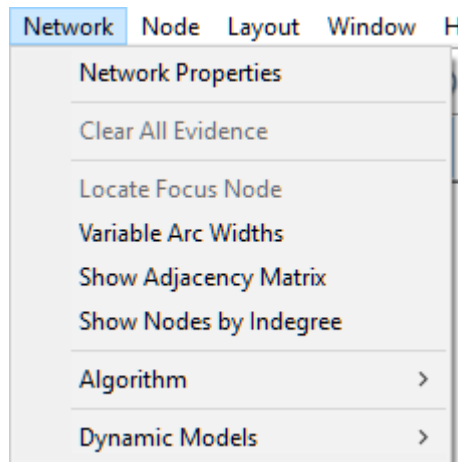


5.4 Model and component properties

5.4.1 Network properties

Network properties sheet summarizes all properties that are specified at the model level. It can be invoked in three ways:

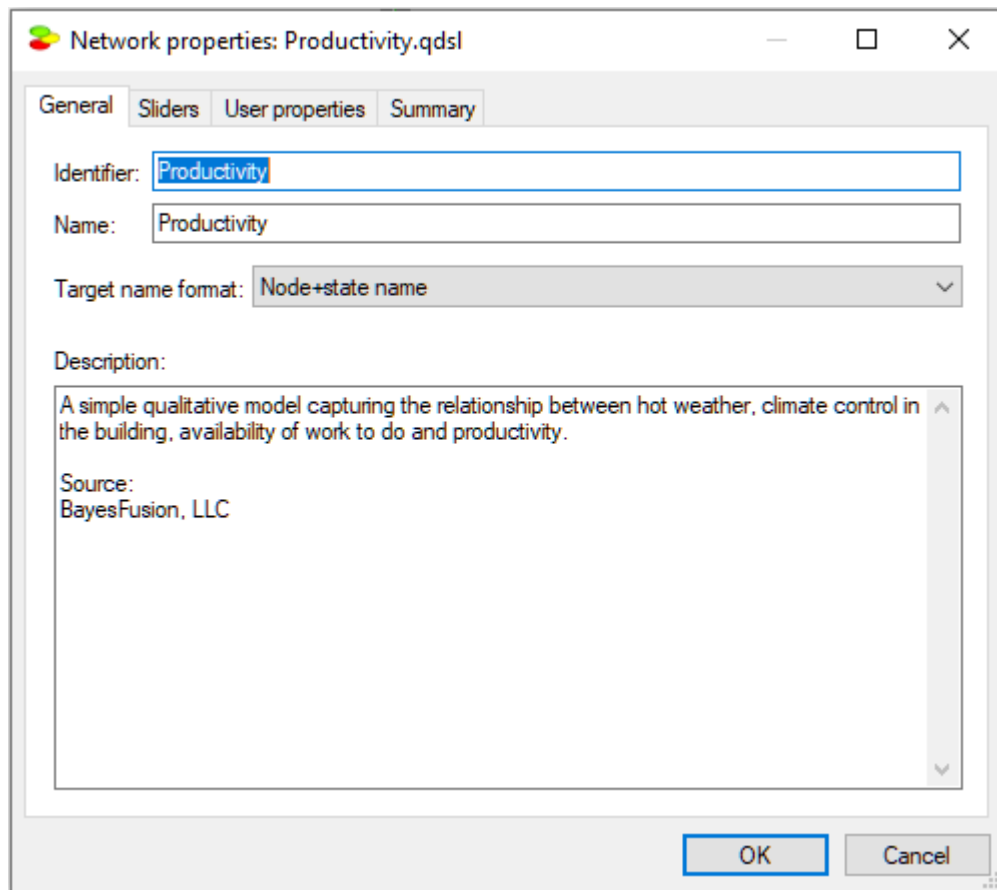
1. Double clicking on a clear area of the network in the graph view.
2. Right clicking on the name of the network in the *Tree View* or right clicking on a clear area of the network in the *Graph View*. This will display the *Network Pop-up* menu. Select *Network Properties* from the menu.
3. Select *Network Properties* from the [Network Menu](#) as shown below.



The *Network properties* sheet, once opened, consists of several tabs.

General tab

The *General* tab of *Network properties* (shown below) consists of the following fields:



The screenshot shows a dialog box titled "Network properties: Productivity.qdsl". It has four tabs: "General", "Sliders", "User properties", and "Summary". The "General" tab is selected. It contains the following fields:

- Identifier:** A text box containing "Productivity".
- Name:** A text box containing "Productivity".
- Target name format:** A dropdown menu showing "Node+state name".
- Description:** A large text area containing the text: "A simple qualitative model capturing the relationship between hot weather, climate control in the building, availability of work to do and productivity."
- Source:** A text box containing "BayesFusion, LLC".

At the bottom right, there are "OK" and "Cancel" buttons.

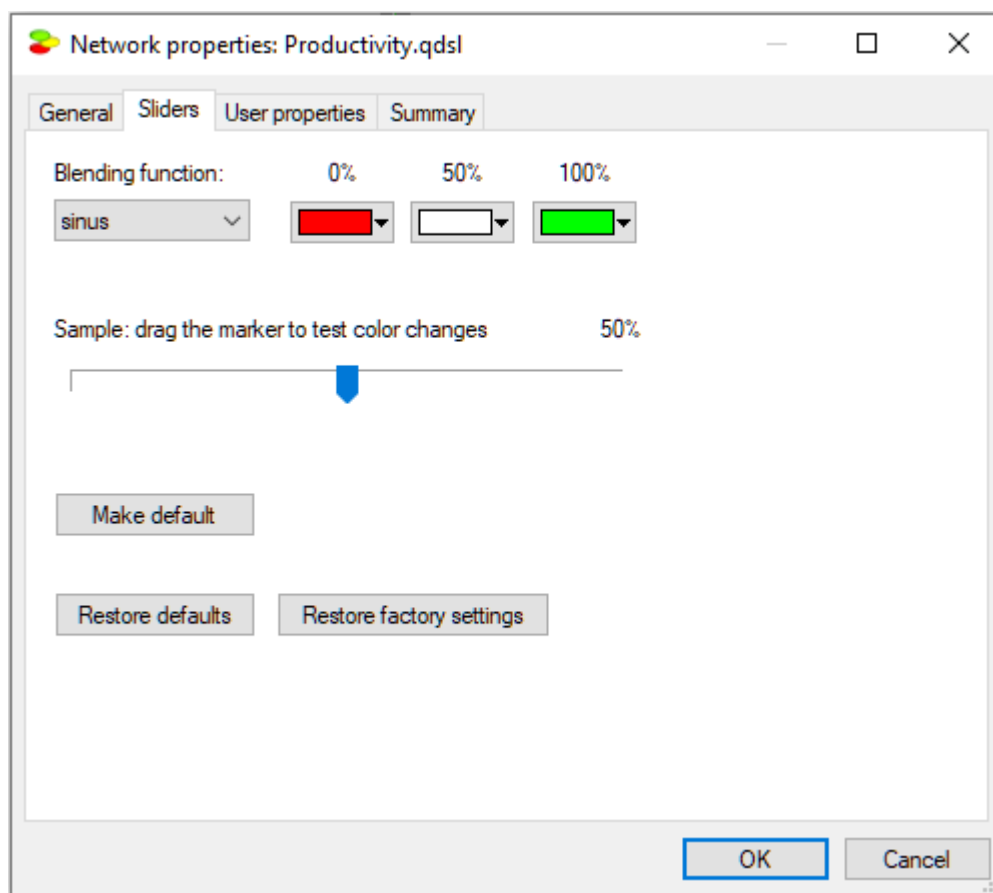
Identifier displays the identifier for the network, which is user-specified. Identifiers must start with a letter, and can contain letters, digits, and underscore () characters. The identifier for the network shown above is *Productivity*.

Name displays the name for the network, which is also user-specified. There are no limitations on the characters that can be part of the name. The name for the network shown above is also *Productivity*.

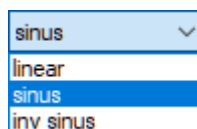
Description is a free text describing the network. Please note that a model is a documentation of the problem and use descriptions generously.

Sliders tab

The *Sliders* tab of *Network properties* (shown below) allows for choosing the colors representing various degrees of belief in the propositions represented by model variables.

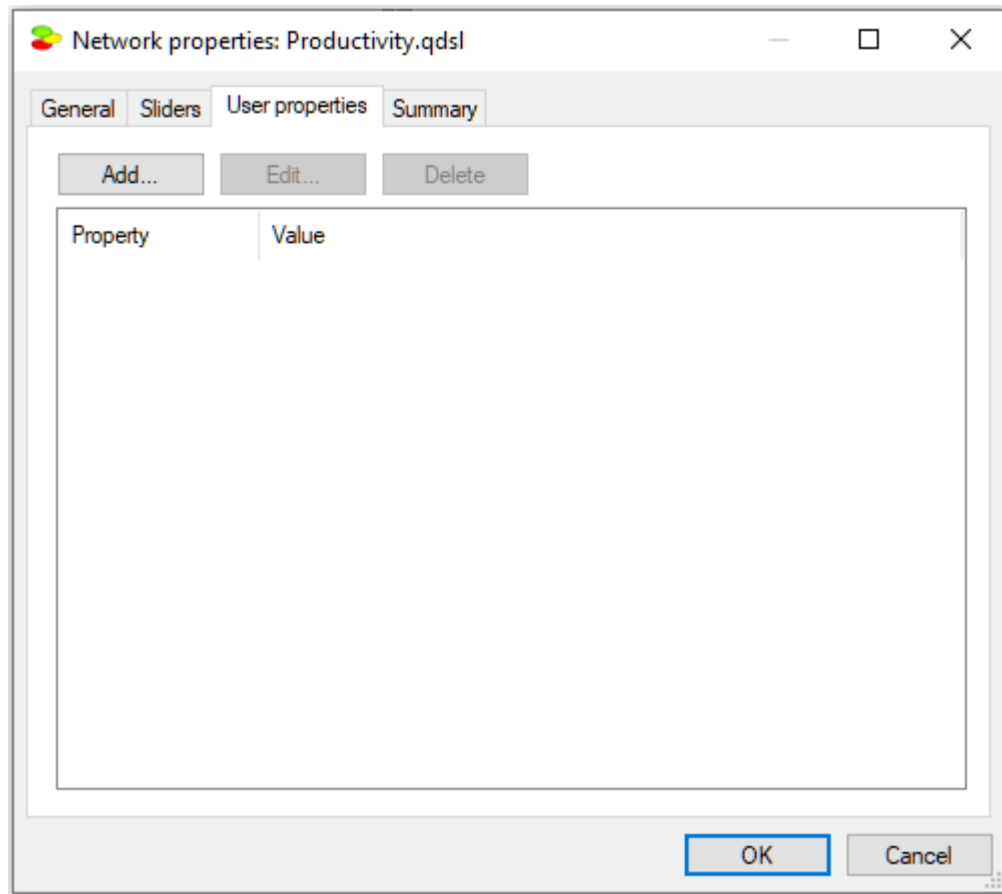


The default colors, which we recommend, range from deep red to deep green, representing probability zero and one respectively. Should you change them and later realize that you would prefer to return to the original settings, you can always use the *Restore defaults* to return to the colors defined by previous pressing of the *Make default* button or *Restore factory settings* to return to the red-green scheme built into the program. Color transitions can be selected through the *Blending function* pop-up menu:



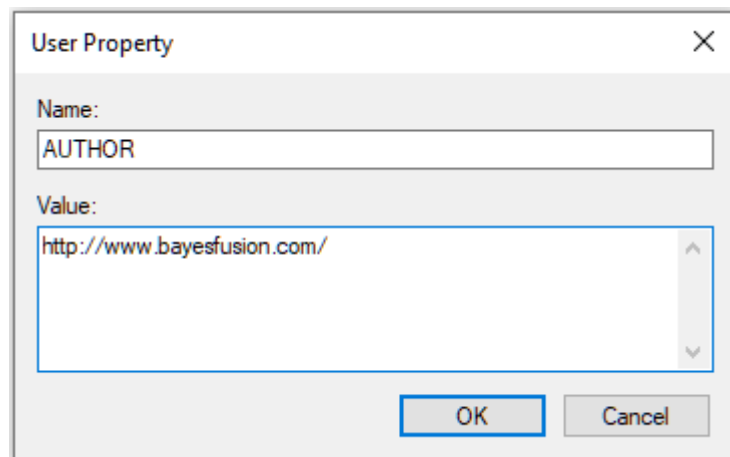
User properties tab

User properties tab allows the user to define properties of the model that can be later retrieved by an application program using [SMILE](#).



For example, we can add a property *AUTHOR* with the value "<http://www.bayesfusion.com/>". Neither QGeNIe nor SMILE use these properties and they provide only placeholders for them. They are under full control and responsibility of the user and/or the application program using the model. QGeNIe and SMILE support only editing and storing/retrieving them.

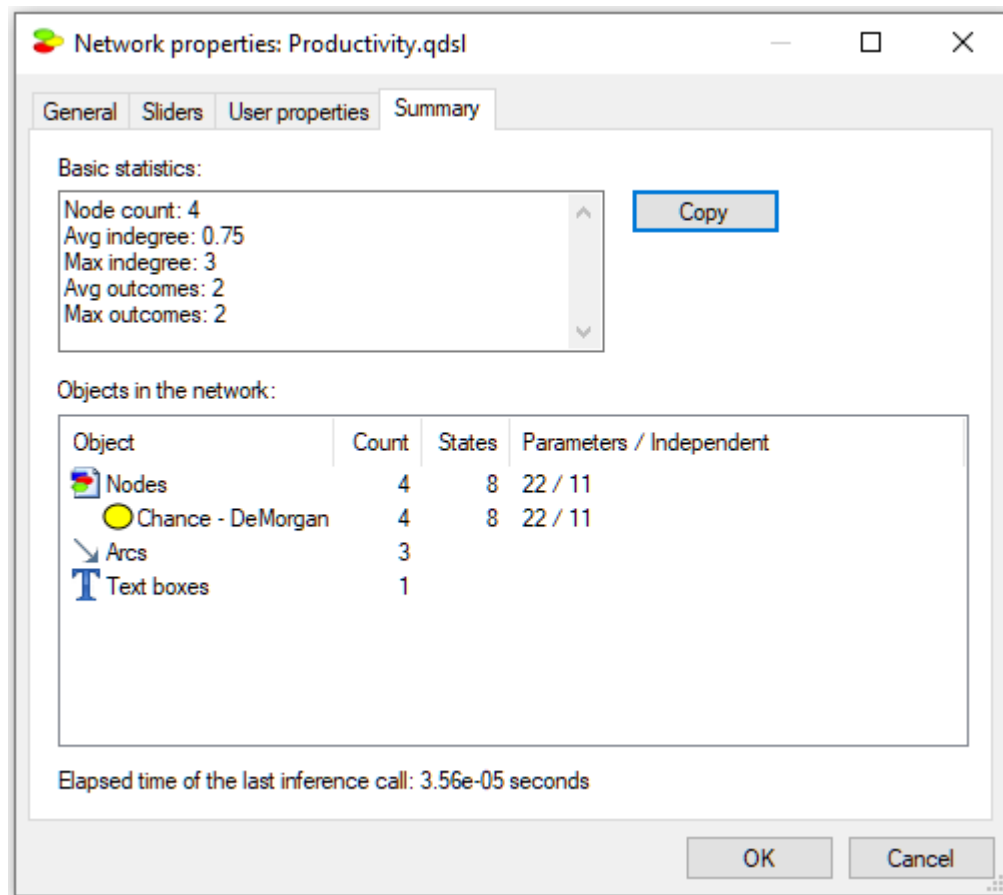
Button *Add* invokes the following dialog that allows for defining a new user property:



Buttons *Edit* and *Delete* allow for editing and removing a selected property, respectively.

Summary tab

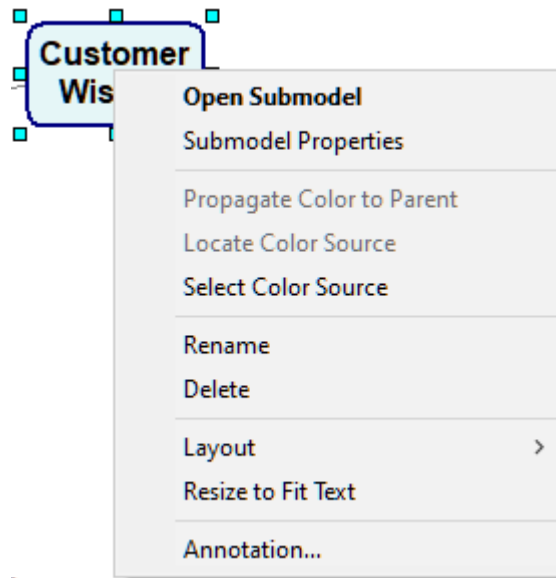
Summary tab contains summary statistics of the network, as illustrated below:



Statistics focus on the structural properties of the network, such as the number of nodes of each type in the network, the average and the maximum in-degree (the number of parents of a node), the average and the maximum number of outcomes of nodes, node counts by their diagnostic type, the number of arcs and the number of text boxes, and, finally, the number of states and parameters. Independent parameters take into account that some parameters are just complements, making sure that probabilities have to add up to 1.0. The *Productivity* model, shown in all illustrations in this section, contains 4 nodes, all of which are DeMorgan nodes. The number of arcs (3) and the average in-degree (0.75) give an idea of the structural complexity of the network. Elapsed time of the last inference call gives an idea of the difficulty that your computer system experienced with solving the model.

5.4.2 Submodel properties

The *Submodel properties* sheet can be displayed by right clicking on the name of the submodel in the [Tree View](#) or right clicking on the submodel icon in the [Graph View](#). This will display the *Submodel Pop-up Menu*. Select *Submodel Properties* from the menu.

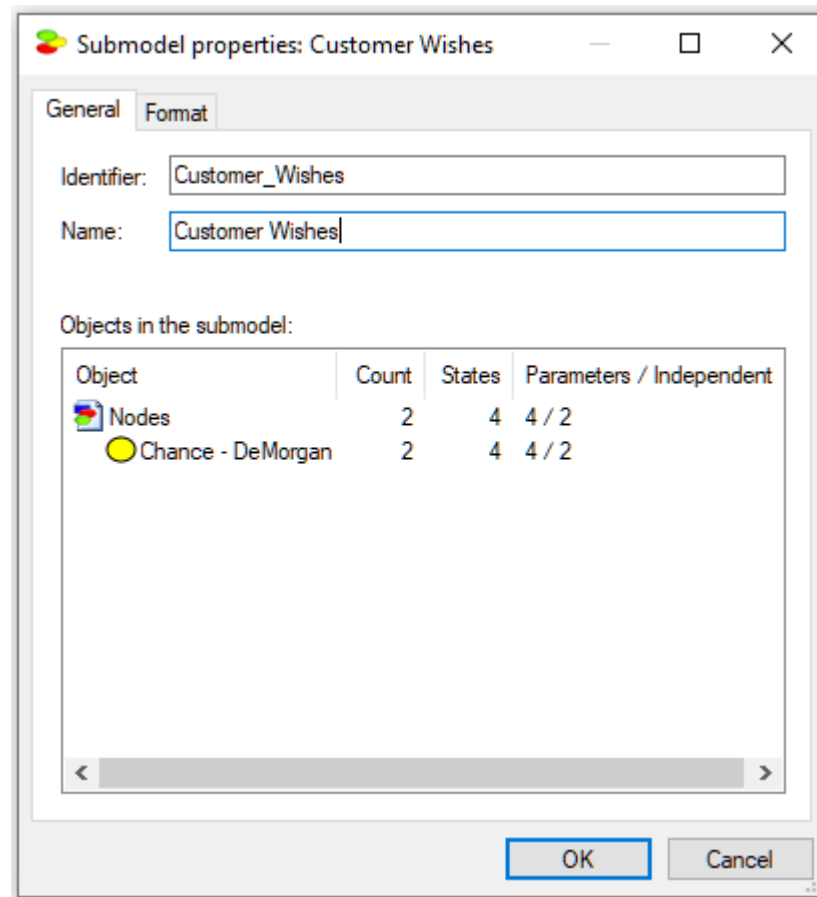


Note : Double clicking on the submodel will open the graph view of the submodel, it will not open the *Submodel properties* sheet.

The *Submodel properties* sheet consist of two tabs: *General* and *Format*.

General tab

The *General* tab displays the *Identifier* and the *Name* of the submodel, along with the submodel's basic statistics.



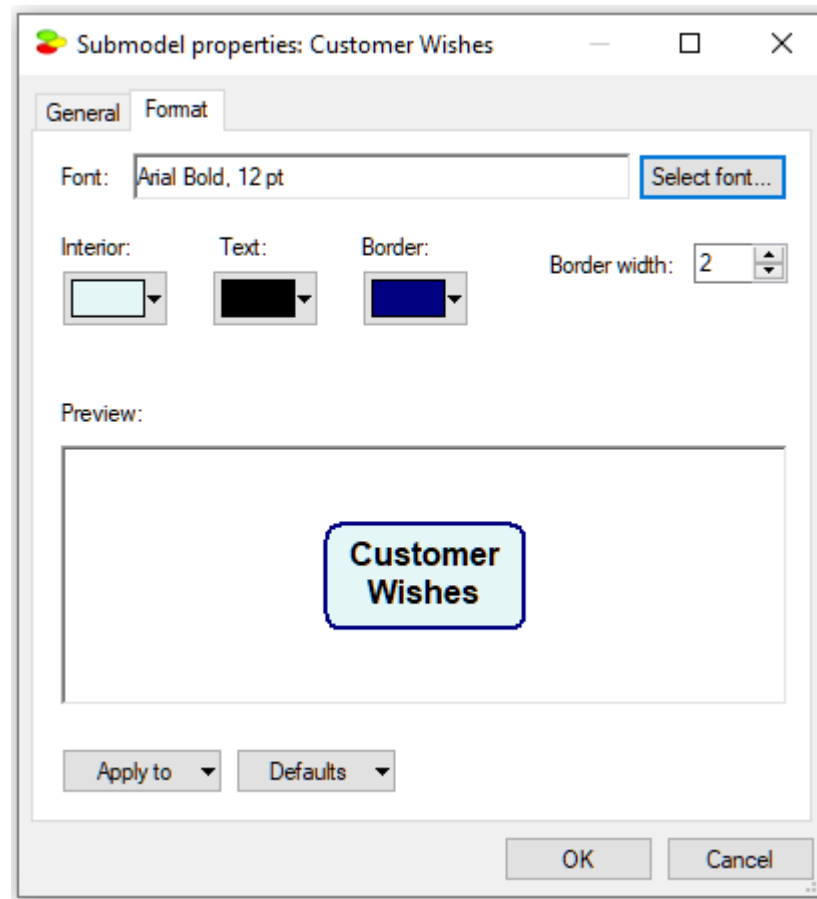
Identifier displays the identifier of the submodel, which is user-specified. Identifiers must start with a letter, and can contain letters, digits, and underscore () characters. The identifier for the network shown above is *Customer_Wishes*.

Name displays the name for the submodel, which is user-specified. There are no limitations on the characters that can be part of the name. The name of the submodel shown above is *Customer Wishes*.

The *Objects in the submodel* lists counts of various types of objects and numerical parameters in the submodel. They give an idea of the submodel's complexity.

Format tab

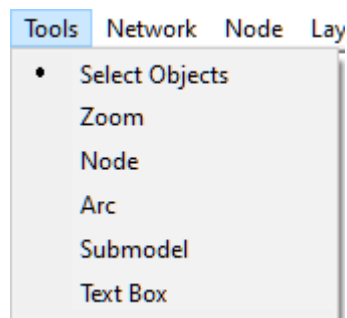
The *Format* tab allows to modify the visual properties of the submodel icon, i.e., how the submodel icon is displayed in the *Graph View*.



The *Format* tab is similar in function to the *Format* tab of the [Node properties](#) sheet.

5.4.3 Tools menu and Standard toolbar

Tools menu is the main bag of tools for building models.



Most of these tools are replicated in the *Standard Toolbar*, which is a bar with buttons that offer quick mouse shortcuts for a number of menu commands.







It is also accessible in a floating form

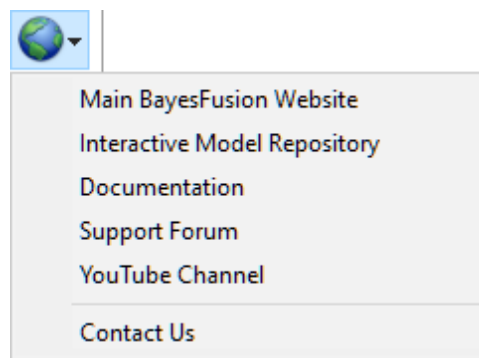


Standard Toolbar can be made invisible using the toggle command *Toolbar* on the *View Menu*. It can be also moved to any position within QGeNIe application window. To move the toolbar from a locked position, click on the vertical bar at the left edge of the toolbar and drag it to its destination. Besides the standard buttons for opening, closing, and saving a file, this toolbar has buttons for selecting various tools found in the *Tool Menu*.

We review here *Standard Toolbar* tools that mimic the *Tools* menu tools and allow for creating objects in the *Graph View* window. The drawing tool currently selected is marked by a dot on the left side of its name in the *Tools* menu. Each of the *Tool* menu tools can be also selected using a corresponding *Standard Toolbar* icon. While choosing a tool from the *Tools* menu, selects the tool for a single editing action (with the exception of the *Select Objects* tool, which is the default tool), the tools on the *Standard Toolbar* work also in *sticky mode*. When a drawing tool on the *Standard Toolbar* is double-clicked, it remains selected until it is deselected (single-clicked upon) or another tool is explicitly

selected. The tool *Node* (), draws a node in the *Graph View*. *Submodel* () draws a submodel, *Text Box* () allows for creating on-screen comments, and *Arc* () allows for creating an arc between two nodes.

The BayesFusion internet resources button () is a short-cut to the *Internet Resources* menu:



Clicking on any of the links leads to opening a corresponding web page in your default web browser:

Main BayesFusion Website:

The main page of BayesFusion's Internet web site.

Interactive Model Repository:

The site of BayesFusion's interactive model repository, powered by [BayesBox](#).

Documentation:

A page with links to various documents, manuals, etc.

Support Forum:

BayesFusion's support forum, a place where you can ask questions and browse previous questions and our specialists' answers.

YouTube Channel:

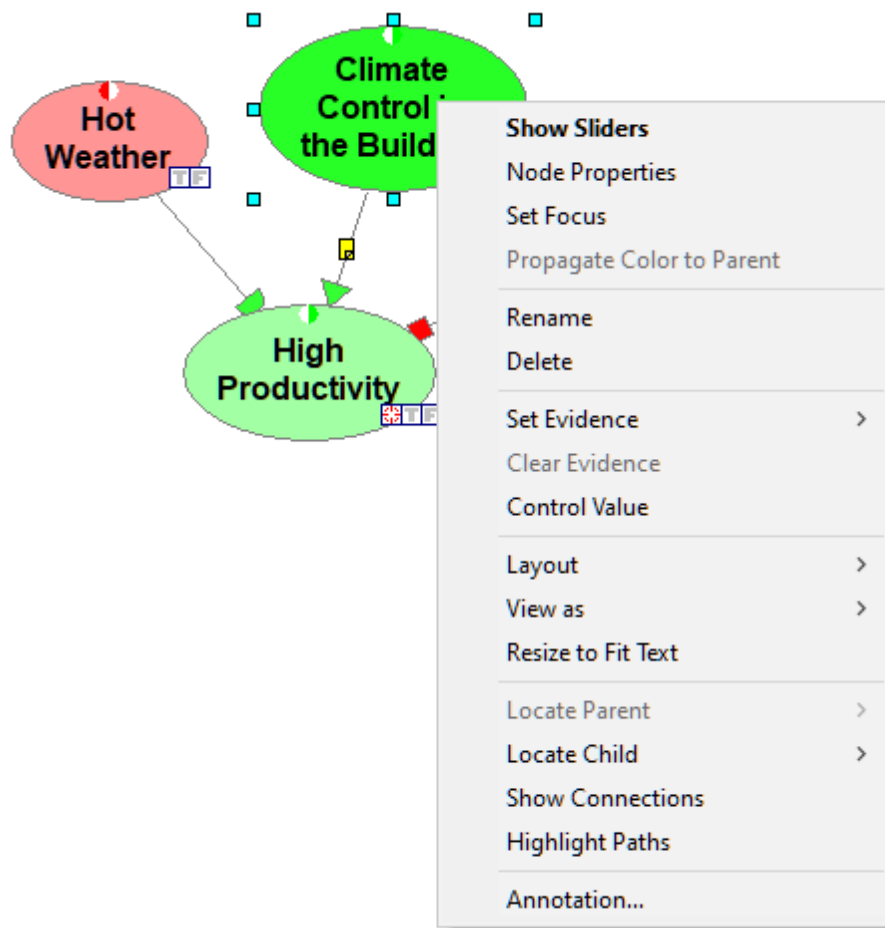
BayesFusion's YouTube channel with movies and instructional videos.

Contact Us:

While we can be reached most reliably by EMail, this is a link to a web-based contact form that can be used to send us an Email message.

5.4.4 Node properties

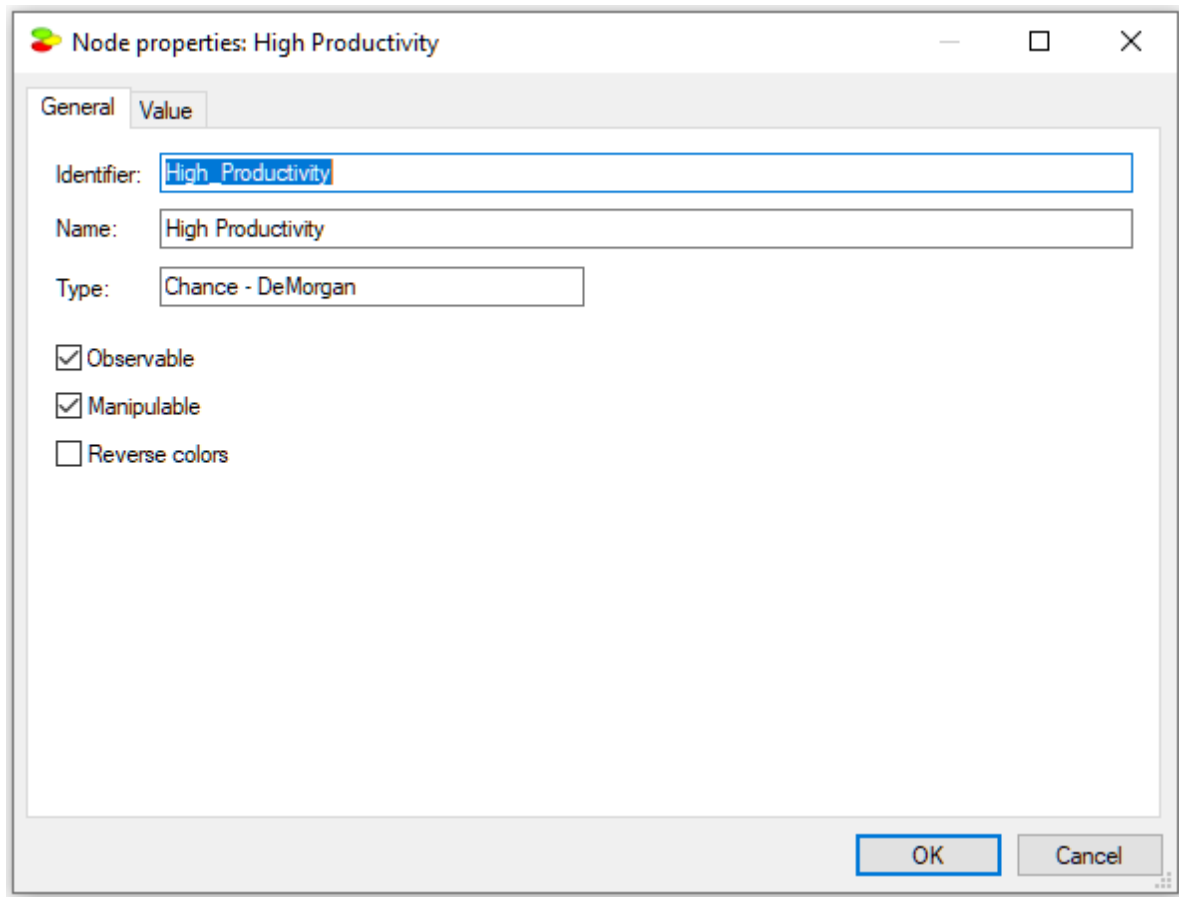
Node properties sheets allow for modifying properties of model nodes. They can be opened right clicking on the name of the node in the [Tree View](#) or right clicking on the icon of the node in the [Graph View](#). This will display the *Node Pop-up* menu. Select *Node Properties* from the menu.



The *Node properties* consist of two tabs: *General* tab and *Value* tab.

General tab

The following image shot shows a snapshot of the *General* tab:



The *General* tab contains the following properties:

Identifier displays the identifier for the node, which is user-specified. Identifiers must start with a letter, and can contain letters, digits, and underscore (`_`) characters. The identifier for the network shown above is *High_Productivity*.

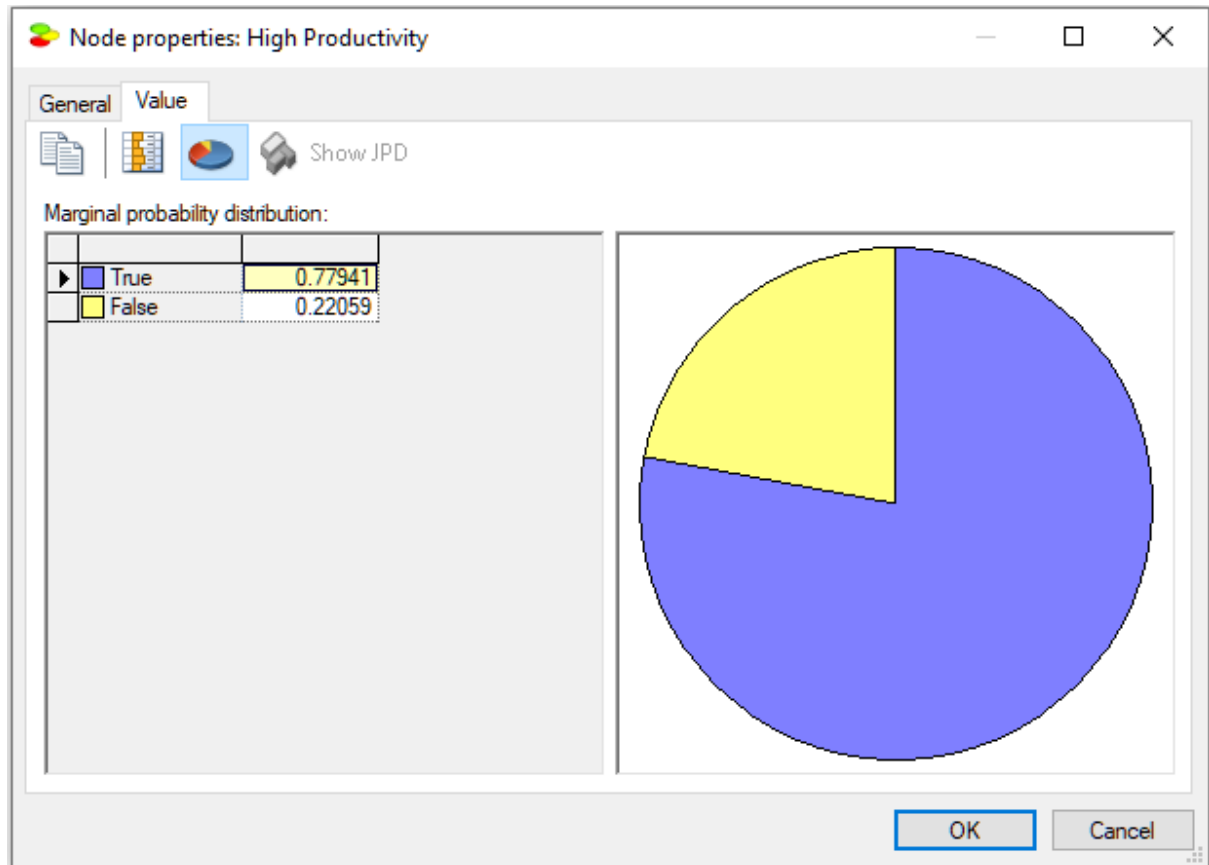
Name displays the name for the network, which is user-specified. There are no limitations on the characters that can be part of the name. The name for the network shown above is *High Productivity*.

Identifiers, which are meant to refer to nodes, may be too cryptic when working with a model, so we advise that QGeNIe users rely only on *Names*. The reason for having both, the *Identifier* and the *Name* is historical and has to do with the fact that nodes may be referred to in equations. In that case, the reference should avoid problems with parsing, which could easily appear with spaces or special characters. QGeNIe allows for creating and editing identifiers because it is common that QGeNIe models are exported to GeNIe, where identifiers become important.

Node *Type* (in the picture, it is *Chance - DeMorgan*) cannot be changed in the *General* tab and serves only informational purpose. Node type is set during node creation.

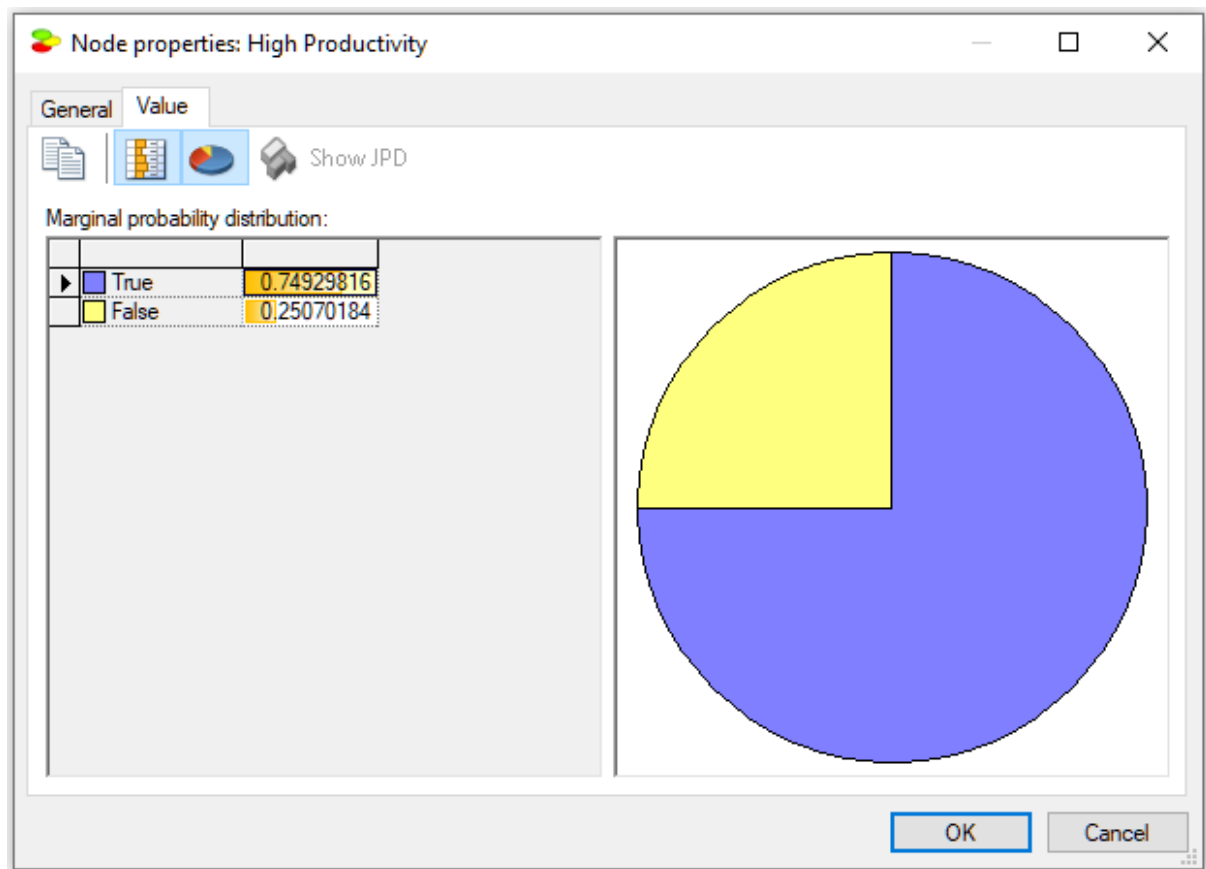
Value tab

The *Value* tab shows the marginal probability distribution over the two states of a QGeNIe node: *True* and *False*.



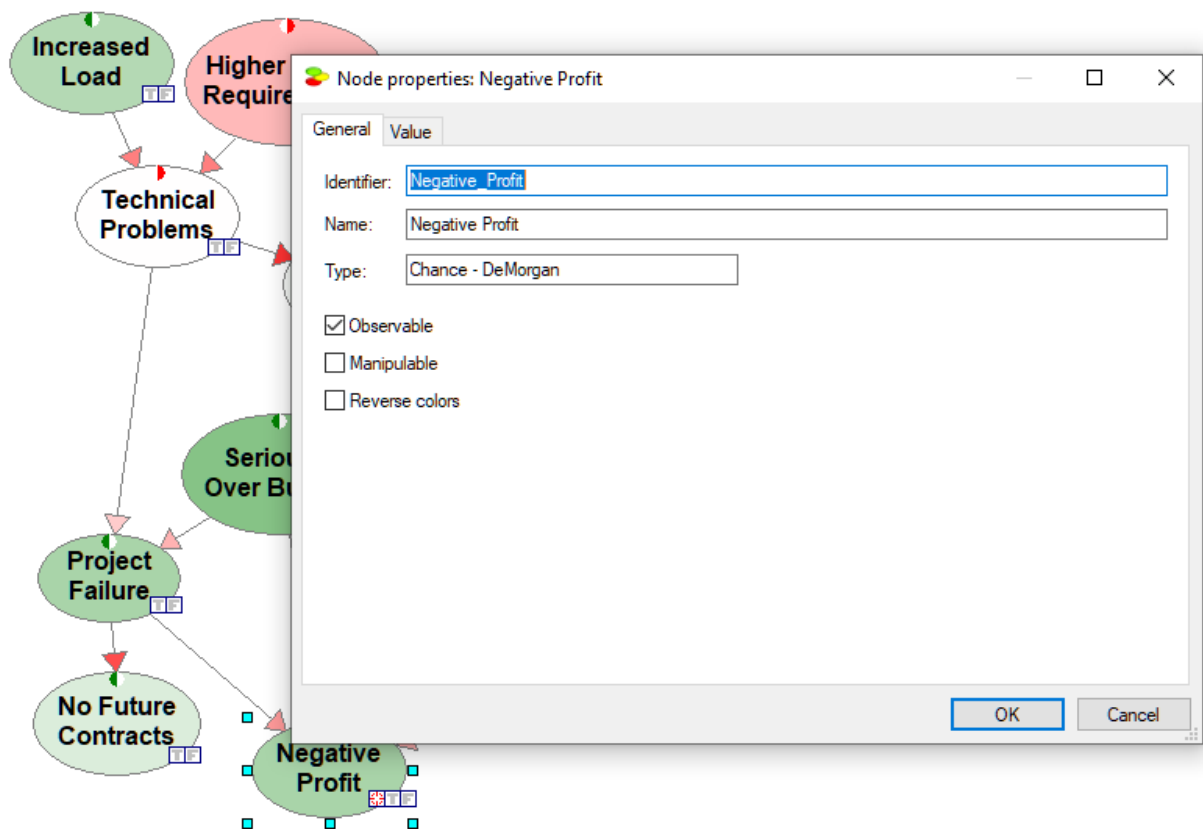
The *Piechart* (🥞) button turns on and off the display of this probability distribution as a pie chart.

The *Show QuickBars* (📊) buttons turns on and off graphical bars in the result spreadsheet (a list of states with their posterior probabilities).

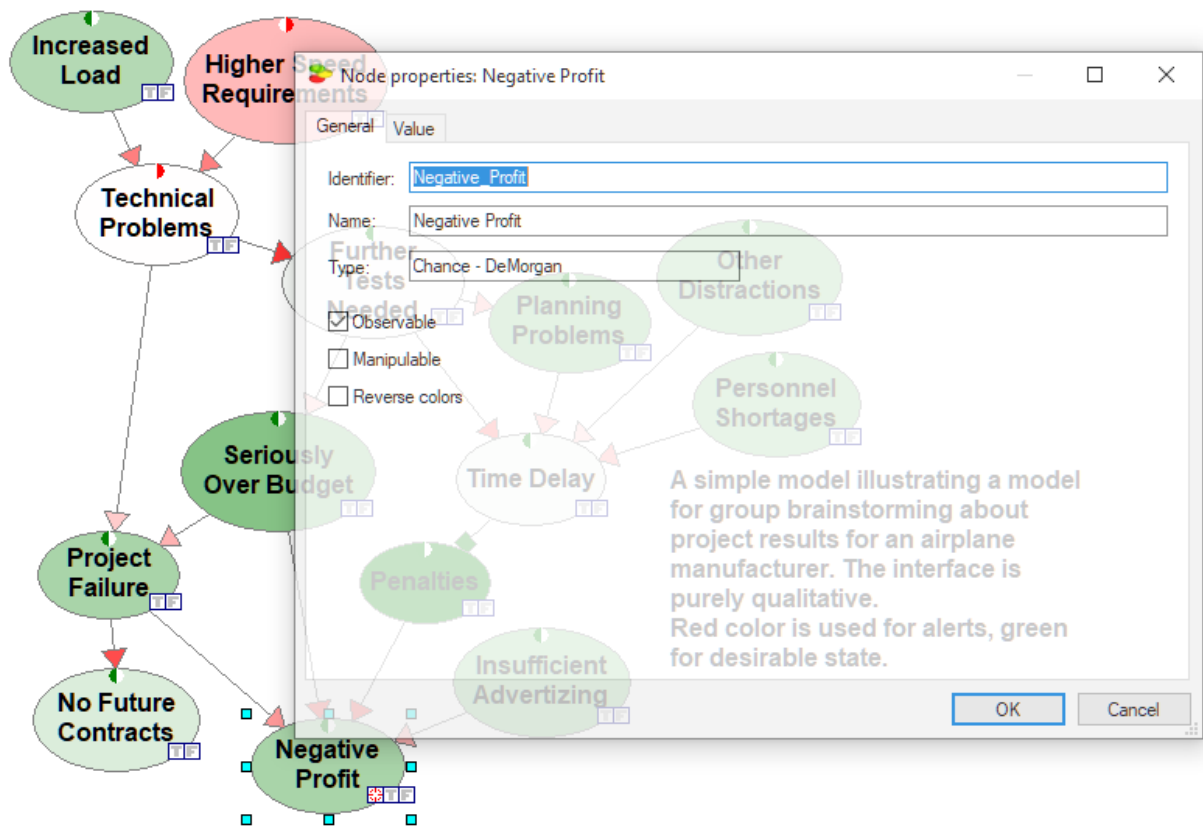


Transparent mode

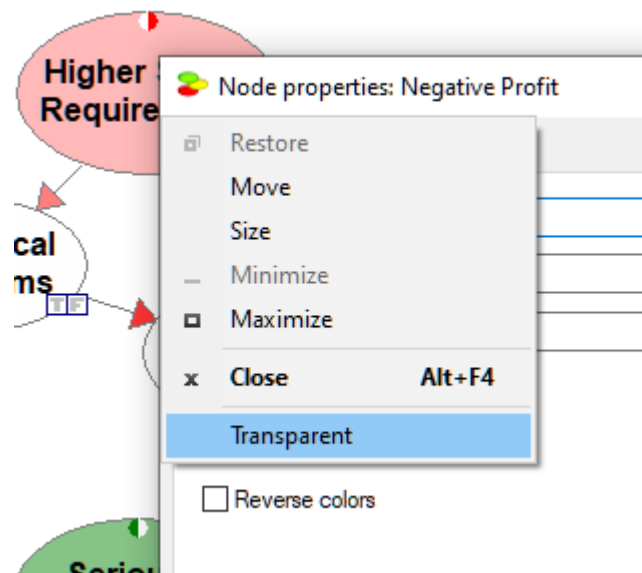
By default, the node property sheets are displayed in opaque mode, i.e., when open, they cover whatever is under them. Here is a screen shot of the *Project* model



It is possible to make the property sheets transparent, which may be handy when navigating through a model. Transparent mode allows to see what is under the property sheets. The same model in transparent mode looks as follows



To toggle between the opaque and transparent mode, check the *Transparent* flag in the *System Menu*, available by clicking on the icon in the upper-left corner of the node property sheets.

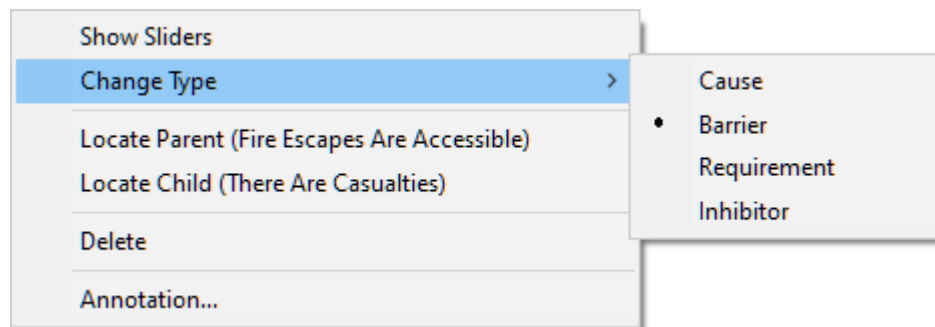


The setting hold only for the currently open property sheet and only as long as it is open.

The transparent mode may be especially useful when the property sheets are maximized, in which case it allows to see what else is on the screen and in the *Graph View* window.

5.4.5 Arc properties

Arc properties can be controlled through the *Arc Properties Menu*, invoked by right clicking on the arc in the [Graph View](#).



Show Sliders allows for entering the strength of influence parameter, described in detail in the [The DeMorgan gate](#) section.

Change Type submenu allows for changing the arc type. Arc types are described in detail in the [The DeMorgan gate](#) section. This operation is useful when we realize that the influence type that we have designated when constructing the model is incorrect. Please do make sure to revisit the strength of influence elicitation, as the meaning of the strength of influence parameter may have changed with the change of the arc type.

Locate Parent and *Locate Child* allow for finding the parent and the child node respectively. This operation is especially useful when the model is complex and the nodes are not clearly visible or in case the parent or the child are located in a submodel.

Delete removes the arc permanently.

Annotation opens up an [Annotation](#) dialog and allows for adding an annotation to the arc.

5.5 Visual appearance, layout, and navigation

5.5.1 Introduction

One of the major strengths of QGeNIe is its graphical user interface. QGeNIe users have repeatedly and consistently praised it for being pleasant, easy to use, and intuitive. It literally cuts the model development time by orders of magnitude. Because this is the bottleneck in applying decision-theoretic methodologies in practice, it translates directly to considerable savings. We have paid a lot of attention to detail and one of the reasons why it is so good is its physical appearance. While each of the sections

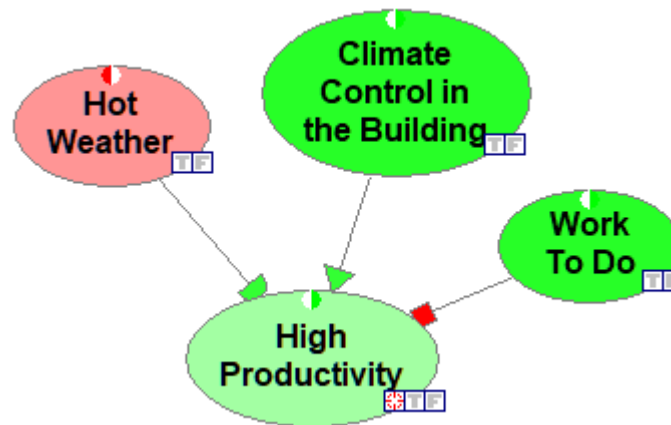
of this manual contains elements of graphical user interface, which we hope the reader will experience as pleasant and intuitive, this section points out some ways in which your interaction with the program can be enhanced and made more efficient.

5.5.2 Viewing nodes in the Graph View

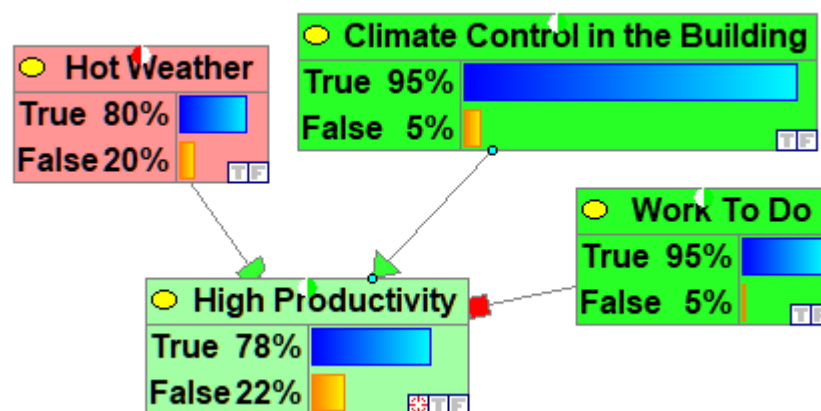
We review in this section two important aspects of viewing nodes in the *Graph View*.

Icons and bar charts

There are two ways of viewing nodes in the *Graph View*: (1) as icons, and (2) as bar charts. The following image shows the *Productivity network* from the [Hello QGeNIe!](#) section, in which every node is shown as an icon:

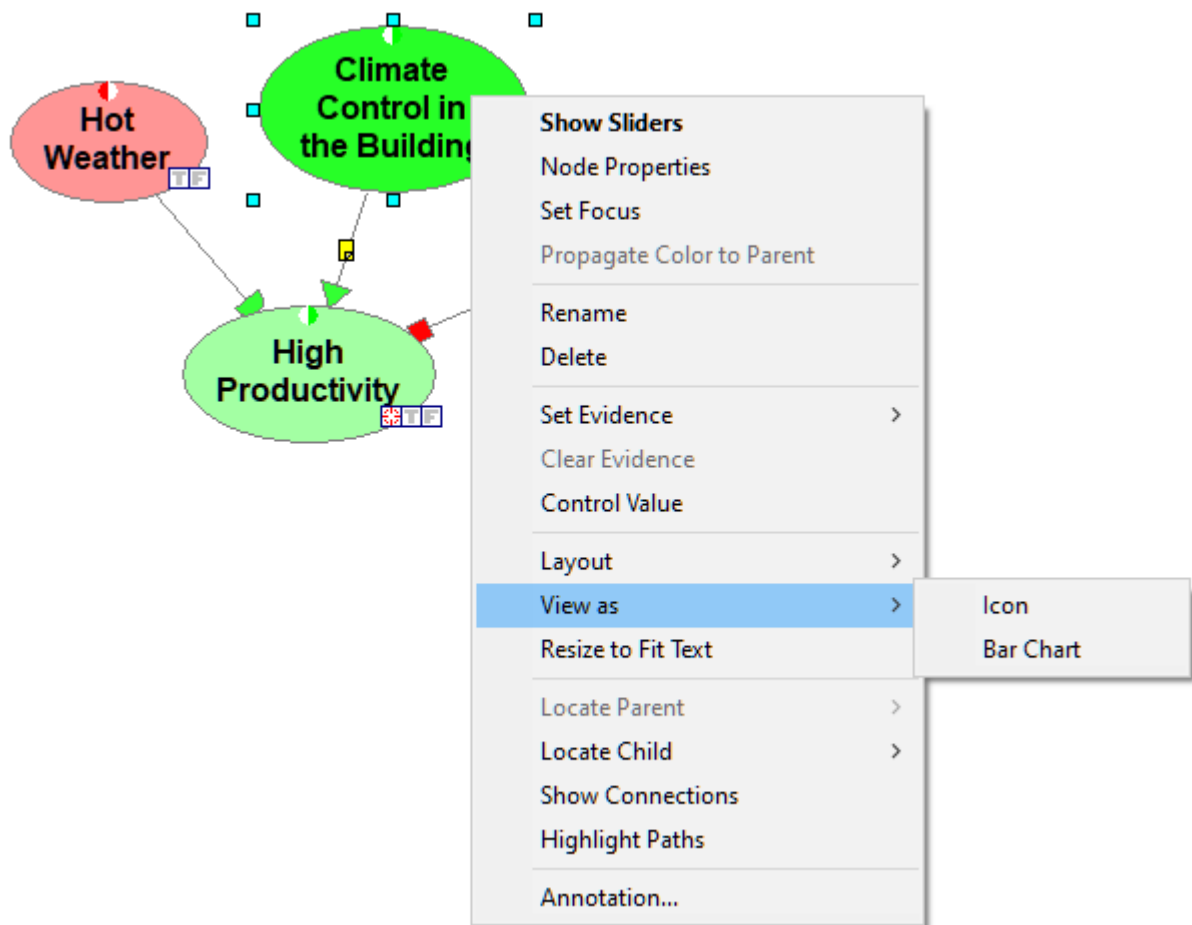


The following image shows the same network, in which every node is shown as a bar chart:



Bar charts display graphically the node's marginal probability distributions. The advantage of seeing a node as a bar chart is that we can see at any point in time its marginal probability distribution expressed by bars and percentages. The disadvantage is that a bar chart takes more space on the screen and may unnecessarily draw user's attention. We advise that those nodes, whose marginal probability

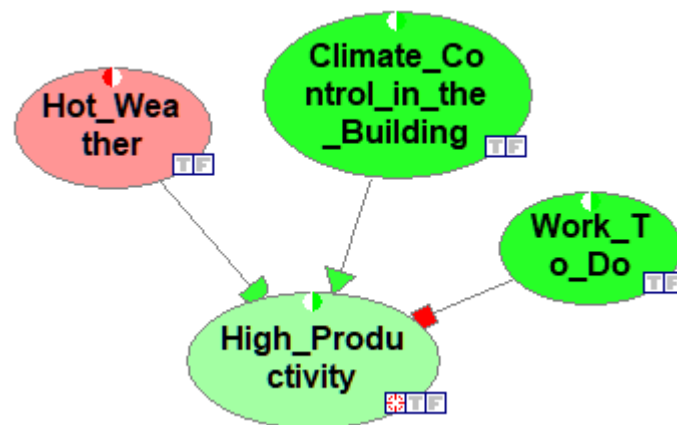
distributions are of interest, are viewed as bar charts and others as icons. To switch between the two view, select the nodes in question and use the *Node Pop-up* menu



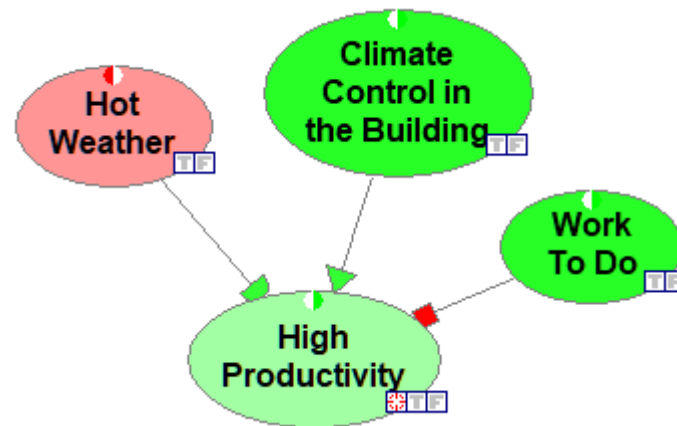
To view all nodes as bar charts or as icons, perform this operation when no nodes are selected.

Names and identifiers

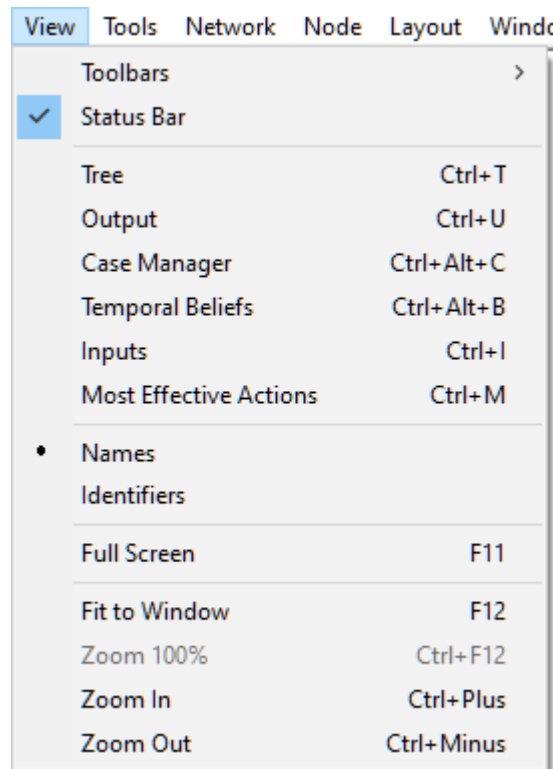
There is another important element of viewing nodes: Viewing their IDs or their names. IDs are short and play the role of variable names. QGeNIe uses them for the purpose of compatibility with GeNIe, and GeNIe, in turn, uses them for the sake of compatibility with equation-based variables, where reference to other nodes in a node's definition has to be through a unique identifier. Identifiers have to start with a letter followed by any combination of letters, digits, and underscore characters. Names are longer and have no limitations on the characters that they are composed of. A model viewed with identifiers looks cryptic.



A model viewed with names is more digestible to human users.




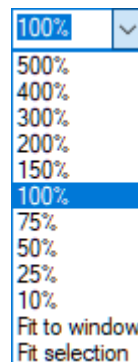
To switch between IDs and names, please use the *View Menu*



We advise that, unless there are important reasons for viewing them as identifiers, nodes be viewed by names. This is more readable for human users.

5.5.3 Zooming and full screen mode

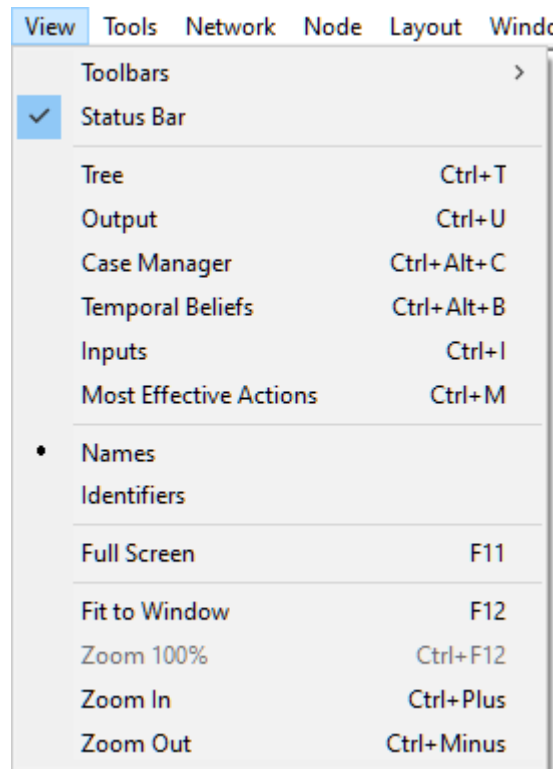
Zoom () is one of the navigational tools and allows for zooming in and out the *Graph View* by clicking with the left mouse button and the right mouse button respectively. An alternative way of zooming is through rolling the mouse wheel while holding down the *SHIFT* key. Yet another is pressing *CTRL*+ and *CTRL*-. The effective zoom percentage is displayed on right side of the *Standard Toolbar*. Zooming can be also performed directly through the *Zoom* menu.



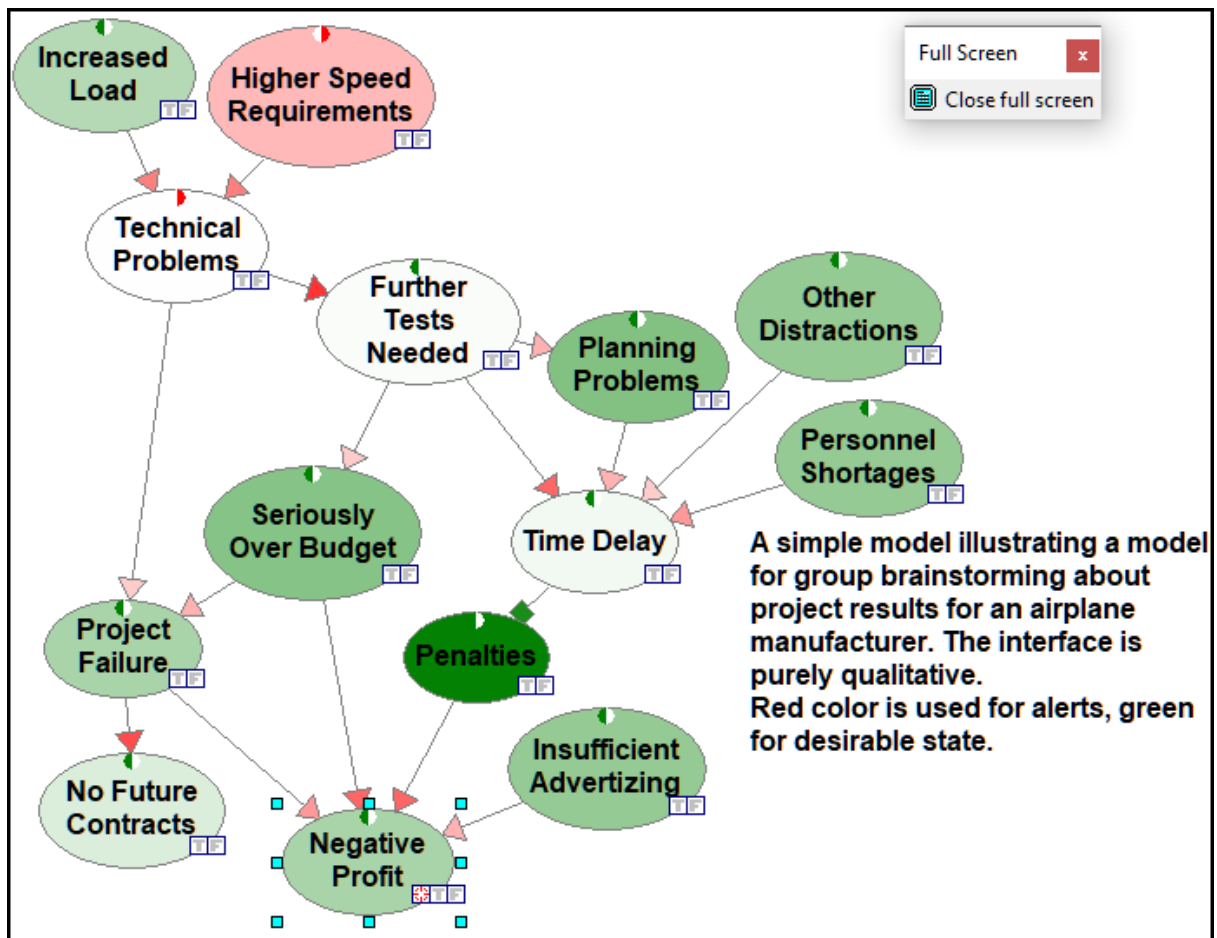
Zoom menu allows for choosing from a small set of predefined zoom percentage values. Two additional useful functions are *Fit to window* and *Fit selection* allow for further customization of the display. *Fit to window* selects the zoom value that makes the entire model visible in the *Graph View*. *Fit selection*

will select an optimal zoom value to make the selected model elements centered and visible in the *Graph View*.

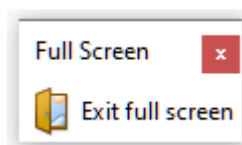
An additional functionality, full screen view is useful in case of limited screen space. To enter the full screen mode, press the *Toggle full screen* (🖥️) button or choose *Full Screen* from the *View Menu*.



The *Graph View* will be expanded to cover the full screen (physically!).



To exit the full screen mode, please press the *Close full screen* button in the following dialog that is present in the upper-right corner of the screen.



This will result in returning to the standard *Graph View*. It is also possible to remove this dialog without exiting the full screen mode, for example in case the dialog covers important parts of the screen. To close the dialog, click on the on the top-right corner of the window. Without the dialog available, you can still exit the full screen mode by pressing the *Esc* or *F11* keys.

5.5.4 Format toolbar and Layout menu

Note: All operations performed using the *Format* toolbar are applicable to the currently selected item in the *Graph View*. All new items created follow the current settings on the *Format* toolbar. *Tree View* cannot be changed using the *Format Toolbar*.

The *Format* toolbar includes tools for refining the aesthetic aspects of the *Graph View*. It can be made invisible using the toggle command *Toolbar-Format* on the *View Menu*. It can be also moved to any position on the screen in its free floating form. To move the toolbar from a locked position, click on the vertical bar at the left edge of the toolbar and drag it to its destination. The *Format* toolbar has buttons for changing font, color, and size of the text, and for alignment of text and various nodes in the model.

Here is the *Format* toolbar



In its free-floating form, the *Format* toolbar appears as follows



Font properties buttons



allow for selecting the font, its size, and appearance (Bold or Italic).

Text justification tools



allow for specification of the text alignment within text boxes, notes, and nodes.

Color tools



allow for setting the interior color of node and submodel icons, line color, and text color. This is similar to color selection in the *Format* tab of [Node Properties](#) sheet.

Line width pop-up tool is used to select the width of the boundary lines of the node/submodel icons.



Node/Submodel Align buttons



become active when at least two nodes are selected in the *Graph View*. They allow for aligning the drawing of nodes to each other or to the grid. Their functionality is self-explanatory and essentially similar to the functionality of most drawing software packages.

Show gridlines toggles display of the grid, i.e., a mesh of perpendicular horizontal and vertical lines in the background of the [Graph View](#). The grid is useful in drawing and aligning nodes.

Align to grid aligns the tops and left sides of selected objects to the nearest grid lines. At least one object must be selected for this option to be active.

Align left and *Align right* align the leftmost and rightmost points of the selected objects, respectively.

Align top, *Align bottom* align the tops and bottoms of the selected objects, respectively.

Center horizontally and *Center vertically* align the object centers in horizontal and vertical direction respectively.

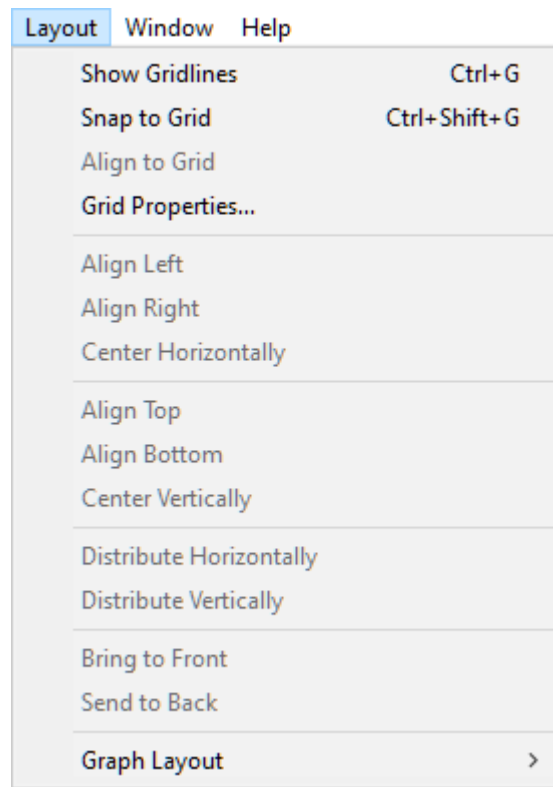
Distribute horizontally and *Distribute vertically* distribute evenly the selected objects respectively horizontally and vertically between the position of the farthest nodes. Both tools will be active only if at least three objects are selected.

Bring to front brings the selected object to the front so that none of its parts is covered by other objects.

Send to back sends the selected object to the back so that none of its parts covers any other objects.

Finally, *Resize to fit text* changes the size of the selected nodes so that the text displayed in the nodes shows in its entirety.

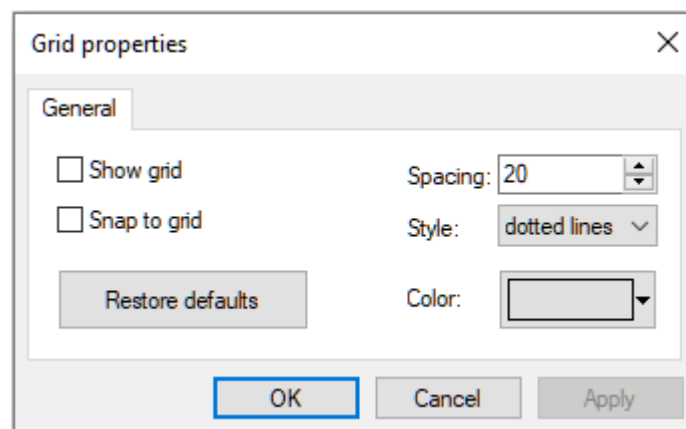
The functionality of *Node/Submodel Align* buttons is repeated in the *Layout* menu



There are two additional functions offered by the *Layout* menu that are not offered by the *Format* toolbar:

Snap to grid makes all new nodes that are created in the *Graph View* aligned to the grid. (If you want to align existing nodes to the grid, select them and use the *Align to Grid* tool, described above.) When *Snap to grid* is on, dragging of nodes, with the *Snap to Grid* option on, will not be smooth, as nodes will jump from one grid line to another.

Grid Properties... opens up the *Grid Properties* dialog shown below



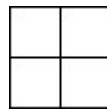
Show grid checkbox has a same function as the *Show Gridlines* option in the *Layout* menu. When checked, the grid lines are displayed in the *Graph View*.

Snap to grid checkbox has the same function as the *Snap to Grid* option in the *Layout* menu. When checked, all new nodes created will be automatically aligned to the grid.

Spacing defines spacing (in pixels) between the lines of the grid. A smaller value will result in a finer grid.

Style defines how the grid lines will be displayed:

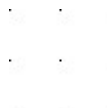
Solid lines makes the grid lines solid.



Dotted lines makes the grid lines dotted.



Dots makes only the intersection points of the grid lines displayed.

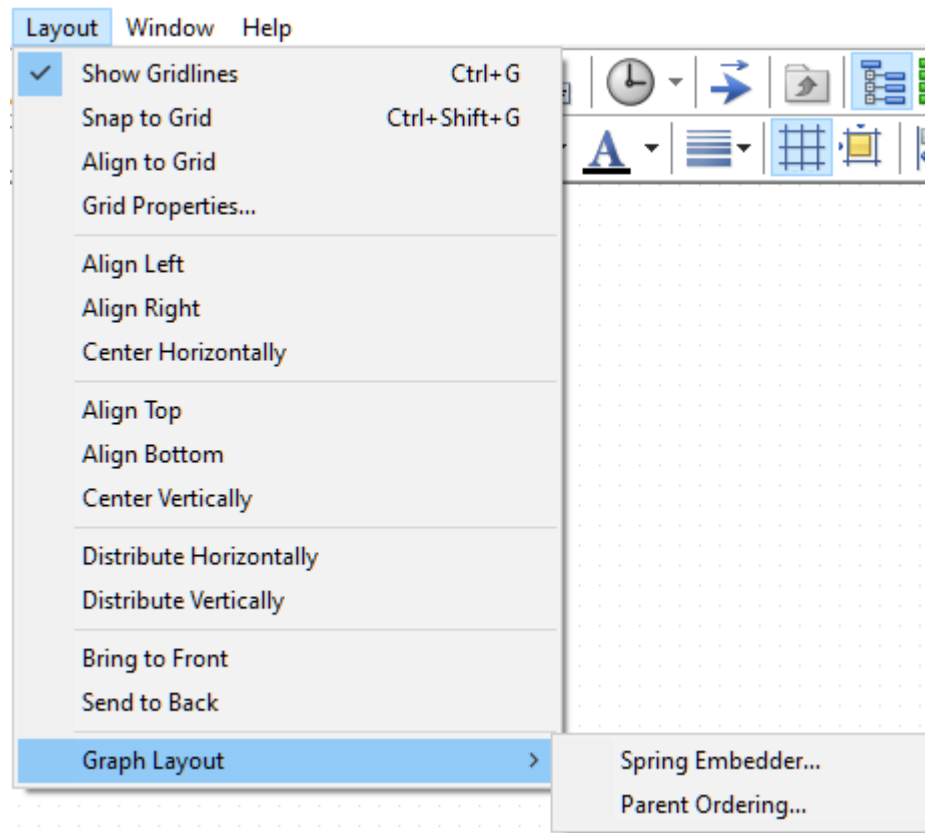


Color sets the color of the grid lines. You can select any color from the palette or define your own color. Grid color selection is similar to color selection for nodes. While darker colors are better when the grid style is dotted or dotted lines, we advise lighter colors for solid grid lines so that they do not clutter the *Graph View* unnecessarily.

Restore Defaults restores the grid display settings to the original factory settings.

5.5.5 Graph layout functions

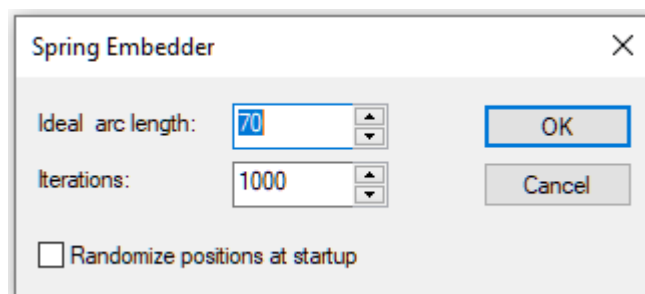
While it is not very likely, it is possible that a model constructed in the *Graph View* becomes too messy. For such cases, QGeNIe supplies functions that arrange nodes within the *Graph View* automatically using two graph layout algorithms: Spring Embedder and Parent Ordering. Layout algorithms are computationally complex and their output may be far from perfect from the point of view of a human user. It is generally a good idea to treat their output as a starting point for manual arrangement of nodes.



Spring embedder

The *Spring Embedder* algorithm (Quinn & Breuer 1979; Eades 1984) is a more sophisticated algorithm of the two and it yields fewer arc crossings and a generally better layout of the graph. It rearranges the positions of the nodes in such a way that the nodes do not overlap each other, arc crossings are minimized, and the layout is readable for the user.

Clicking on the *Graph Layout/Spring Embedder* opens the following dialog:



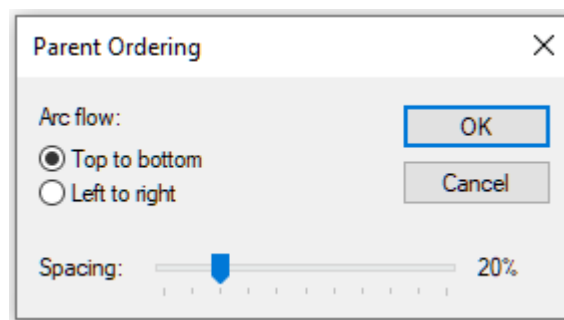
Ideal arc length specifies the ideal distance (in pixels) between two interconnected nodes. Please note that not all arcs will be of this length, the algorithm may modify this length so as to reduce overlaps.

Iterations specifies the number of iterations for the layout algorithm. Reducing this number, especially for very large networks, will translate directly to shorter execution time.

Randomize positions at startup, if checked, causes node positions to be randomized before running the layout algorithm. If it is not checked, the algorithm receives the original node positions.

Parent ordering

Parent Ordering is a simple algorithm for graph layout that essentially places the elements of the model in top-down or left-to right order starting with the parent-less ancestors and ending with childless descendants. The *Graph Layout/Parent Ordering* option opens the following dialog:



Top to Bottom: This places the nodes in the model in the top to bottom on the graph view.

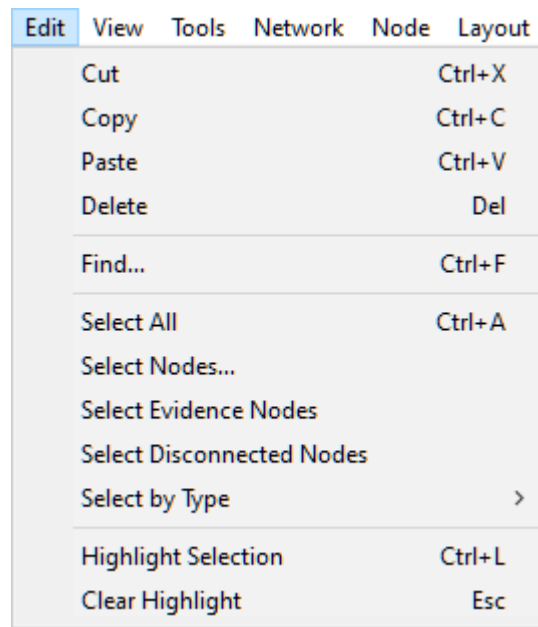
Left to Right: This places the nodes in the model from left to right on the graph view.

Spacing: This specifies the distance between nodes.

5.5.6 Selection of model elements

Often, when constructing models, we want to select their parts so as to change them as a group. The simplest way of selecting a model element is by left-clicking on them. Another common way is by selecting an area in the *Graph View* using the mouse. Everything in the area selected by the mouse will be selected. Adding new elements to the selected set can be accomplished by holding the SHIFT key when left-clicking on the mouse.

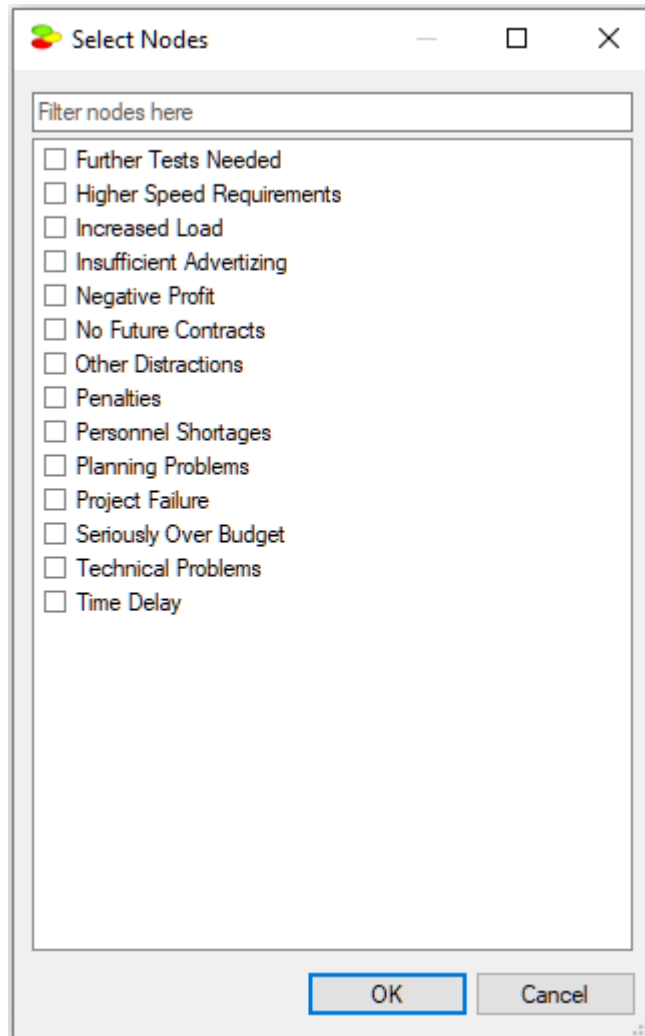
Edit Menu offers several other selection shortcuts that will may make your life easier:



All, with a shortcut *CTRL+A* works in most views and selects all elements (for example, nodes, submodels, and all arcs between them) of the current *Graph View*.

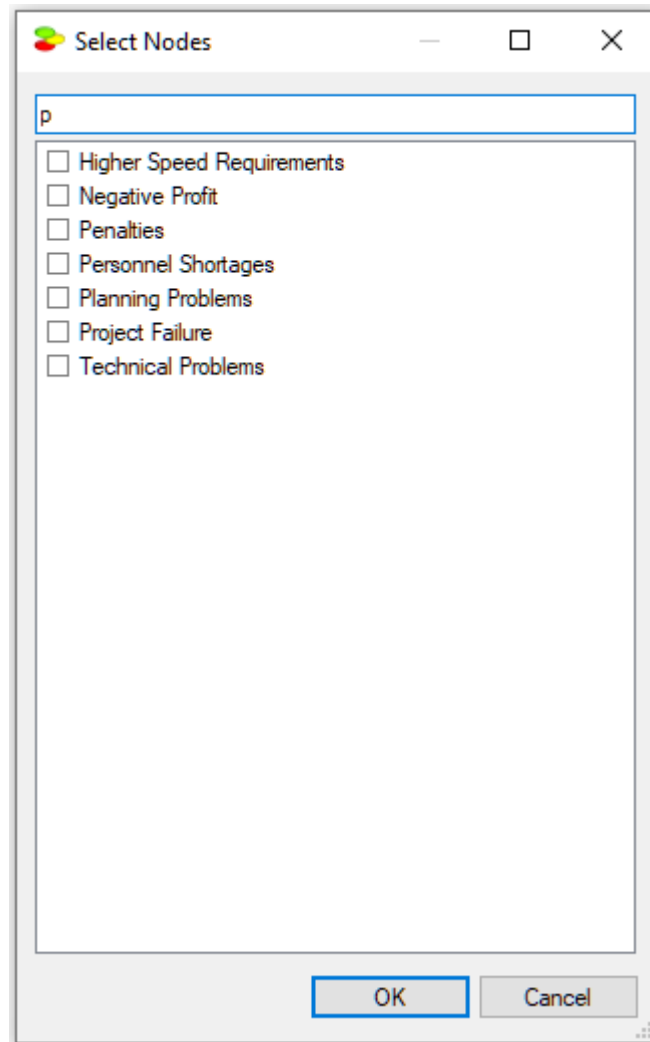
Select Evidence Nodes selects those nodes, for which the evidence has been set by the user. The function is dimmed out if no evidence is present in the model.

Select Nodes... dialog allows for sophisticated selection of nodes based on their names.

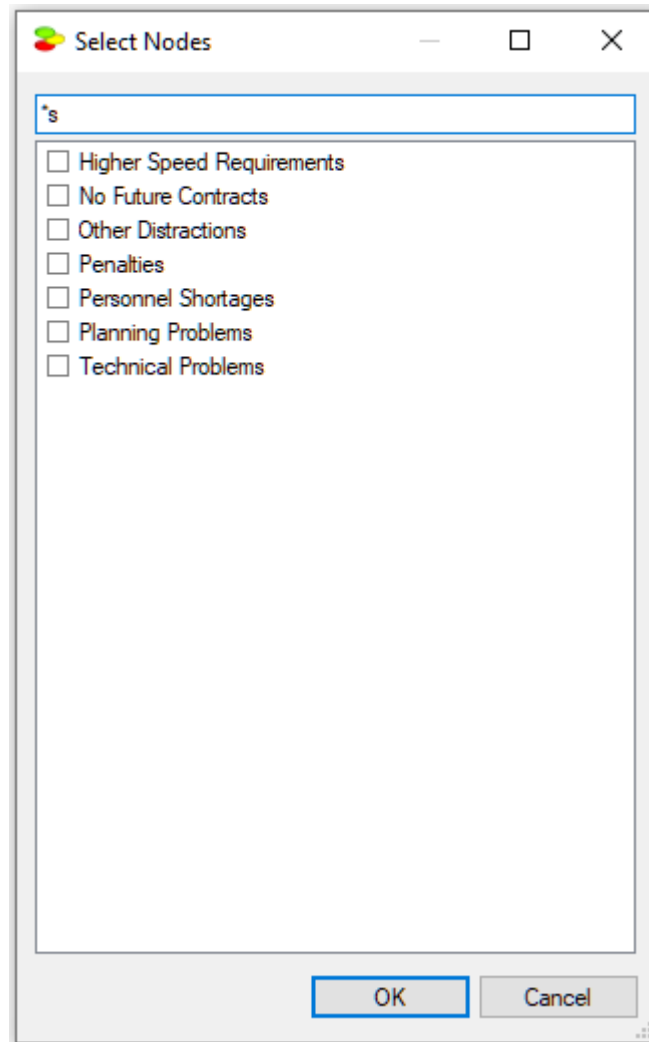


Check boxes next to variable names on the left-hand side allow us for selecting variables. All variables selected in the *Graph View* when invoking the dialog will have the check-box checked. Conversely, all variables with the check-box checked, will become selected in the *Graph View* when we exit the dialog.

Given that a practical model may contains many variables (we have seen in our work with clients qualitative models consisting of hundreds of variables), locating and selecting them may be daunting. The filter field above the list of data columns serves for selecting columns. Typing anything in the filter field causes the dialog to limit the names of data columns to those that match the filter. For example, typing "p" into the above dialog will lead to the following selection:



Letter case is ignored, i.e., "p" is equivalent to "P". In addition to regular characters, the filter field interprets wildcard characters, such as an asterisk (*) or a question mark (?) similarly to Windows. Typing "*s" will select only those variable names that end with an "s":



Any of the variables visible in the list can be selected or de-selected. Selection can be made by checking the small check-box to the left of the variable name or by pressing the space key. All highlighted variables will be selected or deselected by this. Right-clicking anywhere in the window brings up the variable list context menu:

Check item	Space
Invert Checks	Ctrl+I
Select All	Ctrl+A
Copy Checked Items	Ctrl+C
Paste Checked Items	Ctrl+V

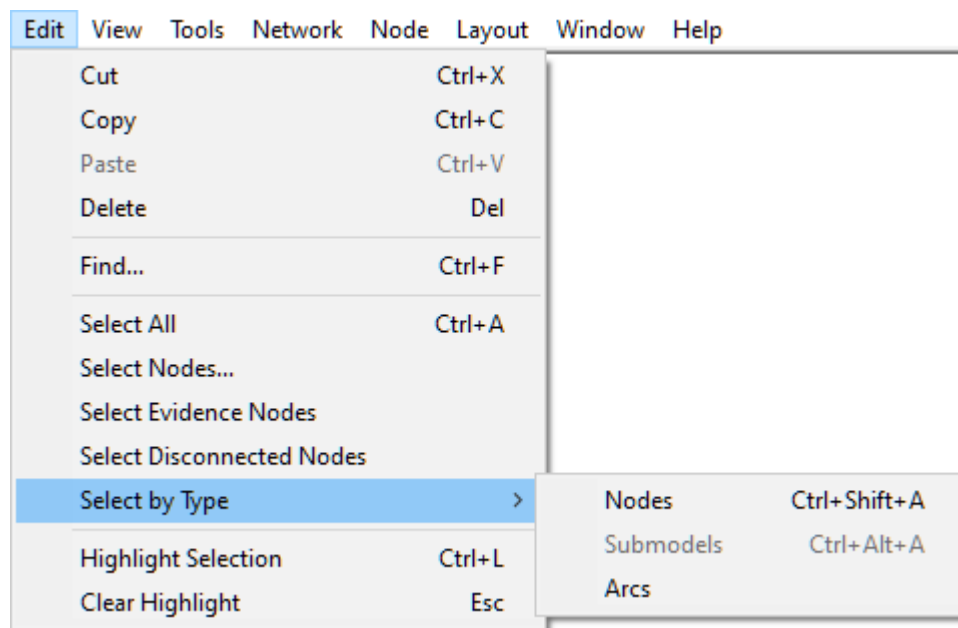
Check item (or pressing the space key) will change the value of the check-box: When it is check, it will be unchecked and when unchecked, it will be checked.

Invert Checks will invert all checks on the list.

Select All will highlight all nodes currently visible in the window. Subsequent pressing of the space key will select or deselect all of them but will have no effect on the remaining (currently invisible) model variables. There is no *Select None* choice in the menu but deselecting all nodes can be accomplished by selecting all nodes and pressing the space key once or twice (if the first pressing leads to selection).

Finally, the interface allows for copying and pasting the list of selected variables. These operations disregard the filter and copy/paste nodes from/to the entire model. *Copy Checked Items* places the list of the selected nodes (text format) on the clipboard. The clipboard format is multi-line text containing the names of items in the list. If, at some later stage, we select and copy the text list outside of QGeNIe and choose *Paste Checked Items*, QGeNIe will select only the variables on the list. *Paste Checked Items* command will remove the current check marks, unless the *SHIFT* key is pressed when command is invoked. This functionality is very convenient in case the list of variables is large and repetitive variable selection process is laborious.

Select by Type submenu offers additional possibilities for node selection:



Nodes, with a shortcut *CTRL-SHIFT-A*, selects all nodes in the current *Graph View*. Please note that *Select All* selects all objects and that includes all submodels, arcs, text boxes, etc., while *Nodes* selects only nodes.

Submodels (shortcut *CTRL+ALT+A*) selects all submodels in the current *Graph View*.

Arcs selects all arcs in the current *Graph View*.

5.5.7 Model navigation tools

QGeNIe includes several simple tools that facilitate model navigation. Each of them is described in detail in other parts of this document. This section lists them in order to expose them and their purpose.

Submodels

Very often, when a model is large, it becomes impossible to navigate through its graph - it may look like a spaghetti of nodes and arcs. Luckily, real world systems and their models tend to exhibit a hierarchical structure (Simon, 1996). There may be several variables that are strongly connected with each other and only weakly connected with the rest of the model. Such may be the case in a business model - purchasing, production, and sales may be three almost autonomous subsystems that can be connected with each other through a small number of links, their inputs and outputs. A decision maker may want to examine each of these subsystems in detail, but may also want to have a global view of the entire business without unnecessary detail. We advise to use submodels whenever a model becomes sufficiently complex.

One problem that a user will experience is navigation between submodels. To aid navigation, QGeNIe allows to traverse the model by right-clicking on small triangles in those nodes that have parents or children in other submodels and locating these.

Tree View and Graph View

Models are typically developed, edited, and viewed in [Graph View](#), which is the program's primary view. Some operations, however, may be more convenient in the [Tree View](#), which offers an alternative to the *Graph View*. Nodes in the *Tree View* are listed alphabetically, so finding them may be sometimes easier than locating them on the screen. The *Tree View* shows the submodel hierarchy and allows for moving nodes between various submodels. The two views work side by side, similarly to a tree view and directory view in Windows.

Model as a document

Following the idea that one of the main goals of a model is documenting the decision making process, QGeNIe supports two constructs that aid documenting the model: text boxes and annotations.

[Text boxes](#) allow for adding an arbitrary text to the background of the *Graph View* window. This text may be useful as a comment explaining the details of the model.

[Annotations](#), which are small yellow stick-it notes, which can be added to nodes and arcs, are useful for explaining function of nodes and arcs between them, or to note down just about anything the user feels is important regarding various model elements.

Text search

Find button and *Find* choice in the *Edit Menu* (described in the [Graph View](#) section) allows for searching through the model for a text. It searches through all text elements of the model, such as IDs, names, descriptions, annotations, text boxes, and displays a list of elements found. These elements can then be located within the model.

Visualization of strength of relationships

Intuitively, interactions between pairs of variables, denoted by directed arcs, may have different strength. It is often of interest to the modeler to visualize the strength of these interactions. QGeNIe offers a functionality (described in the [Strength of influences](#) section) that pictures the strength of

interactions by means of arc thickness. This is especially useful in the model building and testing phase. Model builders or experts can verify whether the thickness of arrows corresponds to their intuition. If not, this offers an opportunity to modify the parameters accordingly.

Status Bar and Output windows

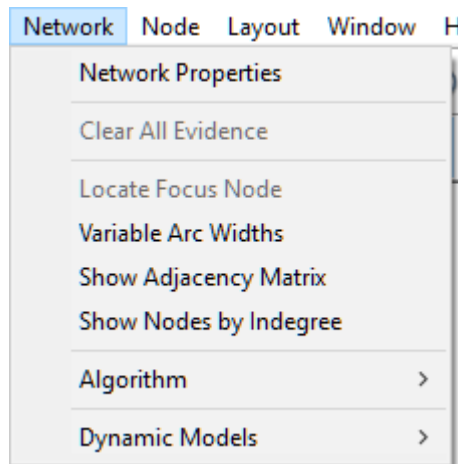
[Status bar](#) tells about problems: The *Status Bar* command displays and hides the *Status bar*. The *Status bar* is a horizontal bar located at the very bottom of the main QGeNIe window. For more information on the *Status Bar*, see the [Status bar](#) section of QGeNIe workspace.

The *Output* command displays and hides the [Output Window](#). For more information on the Output window, see the Output window section of QGeNIe workspace.

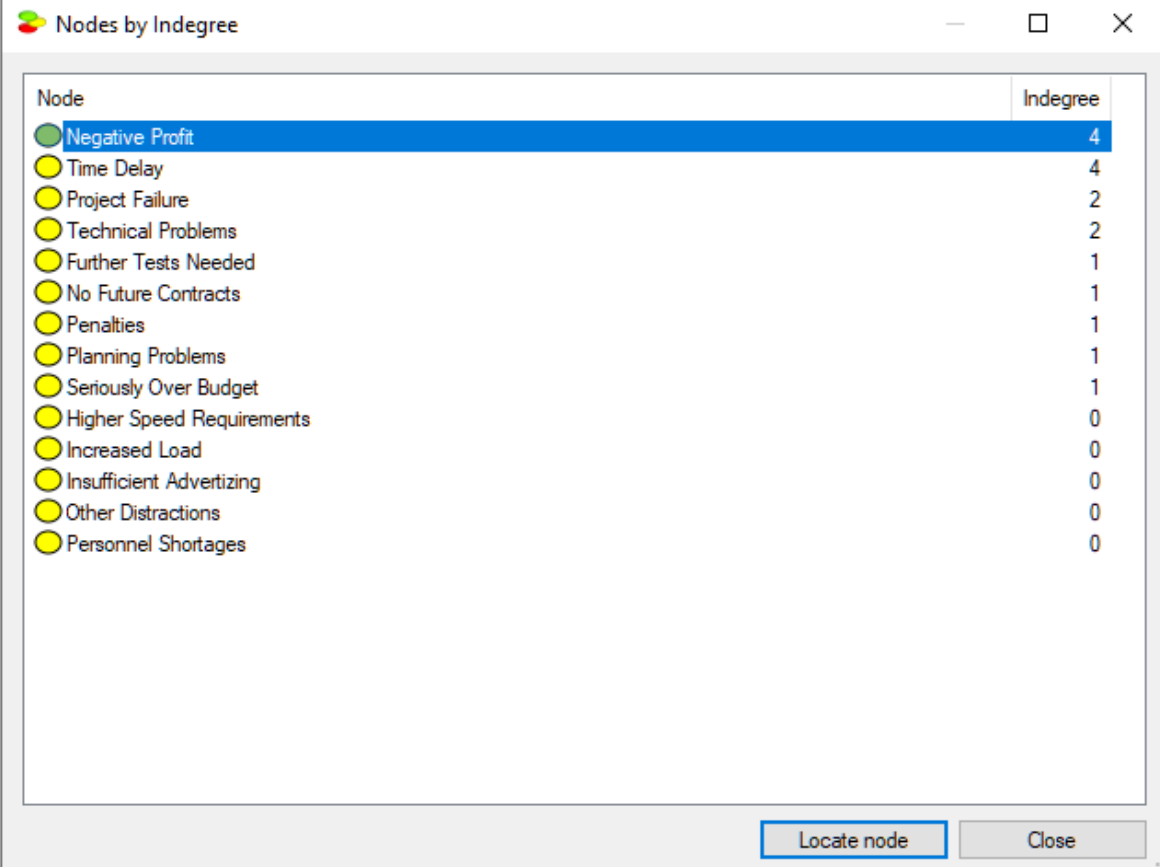
5.5.8 In-degree analysis

Probabilistic inference is worst-case NP-hard, which is a Computer Science term for problems that are possibly unsolvable with limited resources. When constructing models, you may encounter a situation in which the model becomes too complex for inference (QGeNIe will generally warn you about this in the [Output Window](#)).

Problems with inference are typically caused by the complexity of connections in the graph. While it is hard to control complexity of connections, there is one way that it can be kept reasonably low - making sure that nodes do not have too many parents. QGeNIe will warn you when you add more than 20 parents to a node but will not forbid you doing so. Nodes with fewer than 20 parents can also lead to difficulties with inference. There is also a dialog that allows you to view nodes by their number of parents (this is called in the literature *indegree*). Choosing *Show Nodes by Indegree* from the *Network Menu*:



shows the following dialog:



Node	Indegree
Negative Profit	4
Time Delay	4
Project Failure	2
Technical Problems	2
Further Tests Needed	1
No Future Contracts	1
Penalties	1
Planning Problems	1
Seriously Over Budget	1
Higher Speed Requirements	0
Increased Load	0
Insufficient Advertizing	0
Other Distractions	0
Personnel Shortages	0

The user can get an idea which families (nodes and their parents) can give possible problems with inference. Reducing the number of parents of those nodes that have the highest indegree can help with making inference more efficient. To visit a node from the list in the [Graph View](#), please double click on the node or select on the list and press the *Locate node* button.

5.6 Saving and loading models in QGeNIe

5.6.1 Introduction

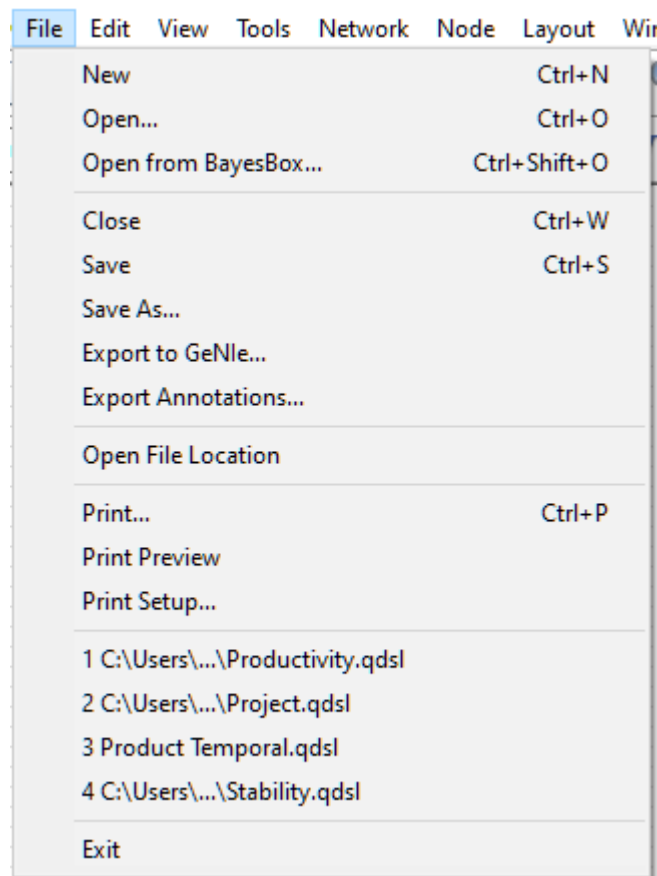
While QGeNIe is a general purpose decision modeling environment, it has been originally written with research and teaching environments in mind.

Opening a model in QGeNIe

There are three ways in which you can open a model in QGeNIe,

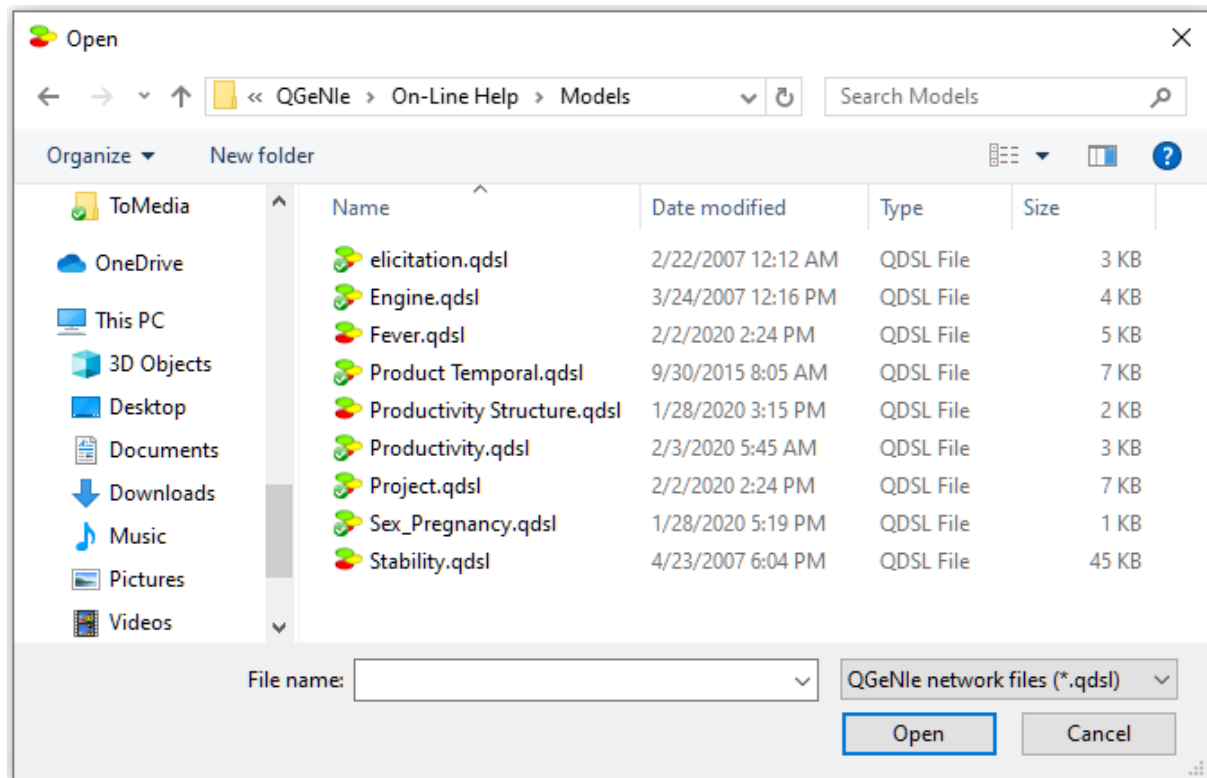
- Choose *Open...* dialog from the *File* menu
- Click on *Open* (📁) button from the *Standard Toolbar*
- Use the *CTRL+O* shortcut

Shown below is the *Open...* option in [File Menu](#).



The numbered list of files at the end of the [File Menu](#) are the *Most Recently Used (MRU)* file list. You can click on any of those names to re-open the file.

Each of the three ways of invoking *File Open* dialog leads to the following:



The dialog that appears allows you to choose a file to load.

QGeNIe uses the QDSL file format, which is an XML-based format.

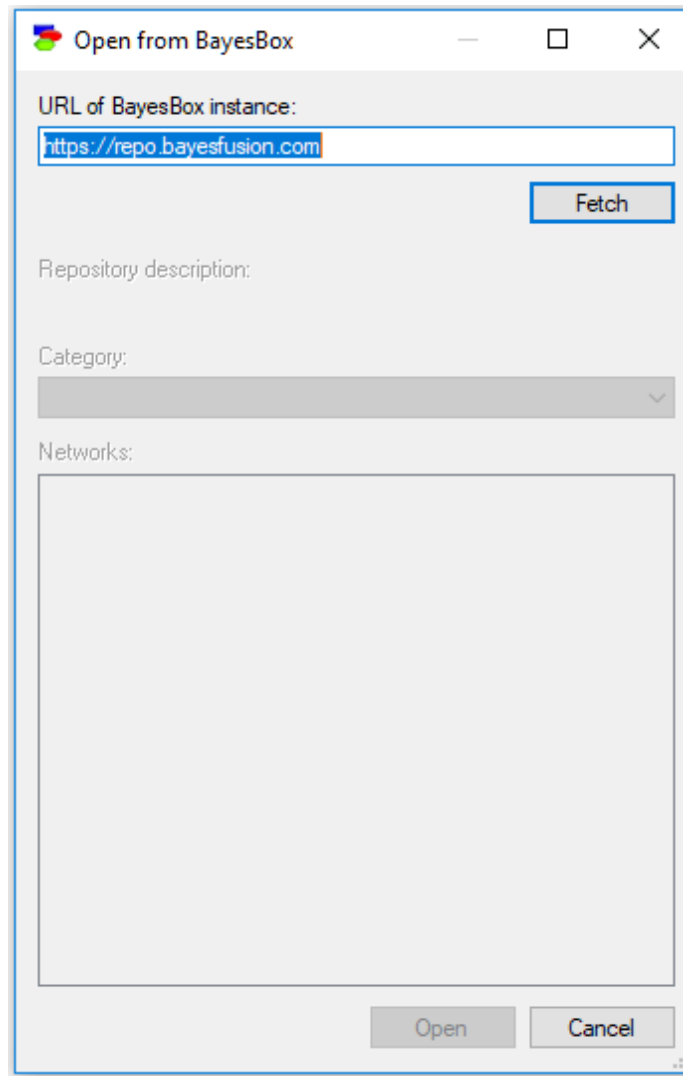
Note : You can have multiple files open at the same time.

Opening a QGeNIe model from a BayesBox-powered model repository

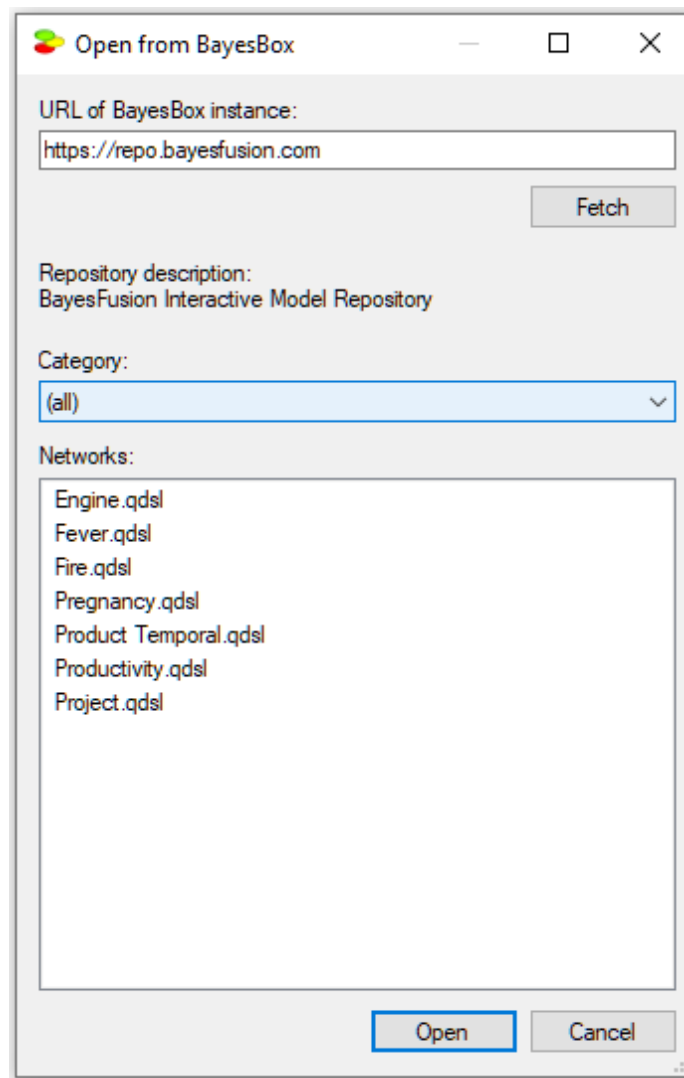
QGeNIe supports loading models from BayesBox-powered repositories, which are web-based repositories, possibly local and internal to user's organization. There are three ways in which you can open a repository model in QGeNIe,

- Choose *Open from BayesBox...* from the *File Menu*
- Use the CTRL+SHIFT+O shortcut
- Press the *Open network from BayesBox* (📦) button

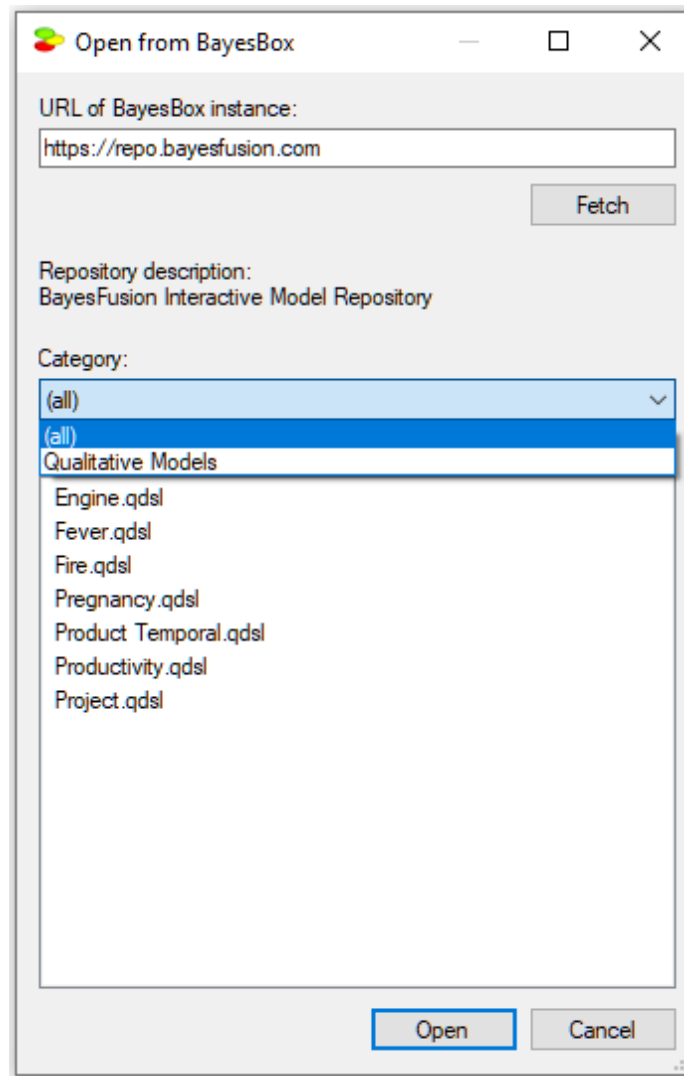
Each of the three actions opens the *Open from BayesBox* dialog:



The default BayesBox instance is BayesFusion's model repository but any web address can be entered here. GeNIe remembers the last successful connection to a BayesBox instance, which is convenient for those users who rely on their internal BayesBoxes. Fetch allows to see models and categories present in the chosen BayesBox instance:




It is possible to traverse BayesBox's directory structure by clicking on the Category pop-up menu:



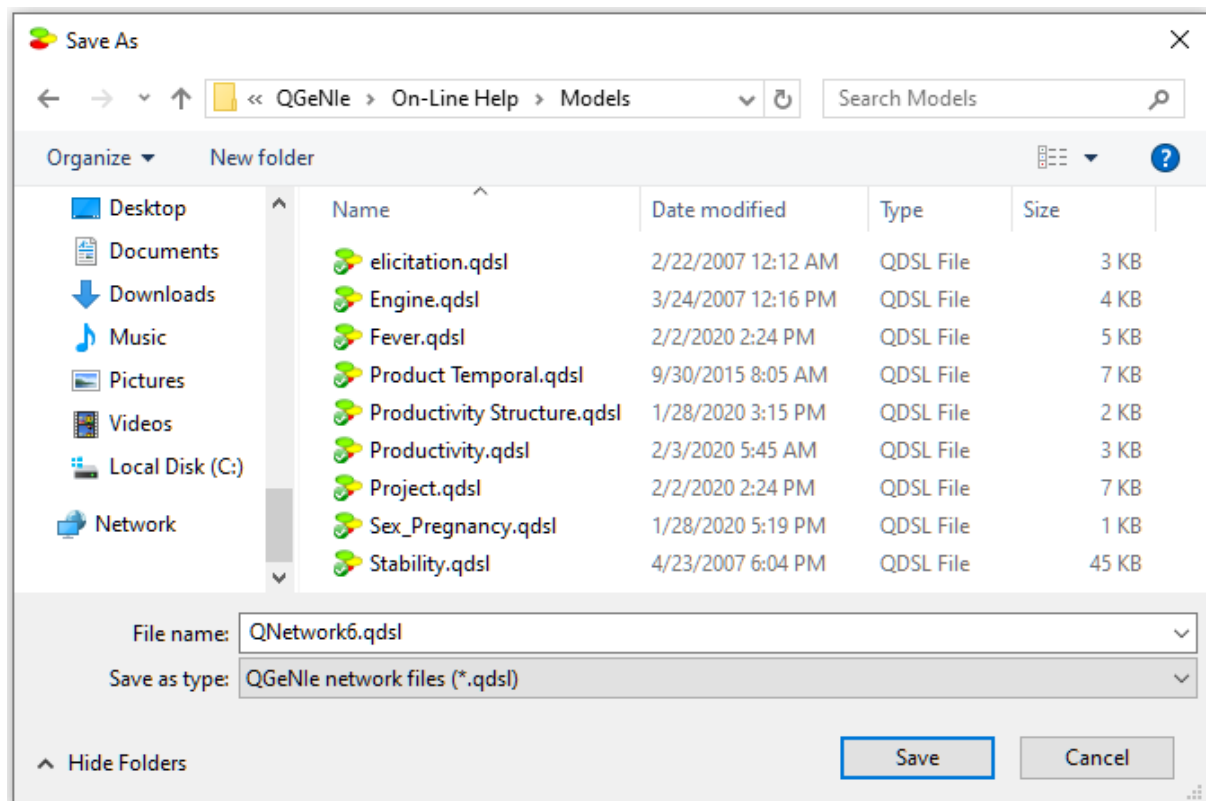
Selecting a model from the list fetches the model from the current BayesBox instance and opens it in QGeNIe.

Saving a model in QGeNIe

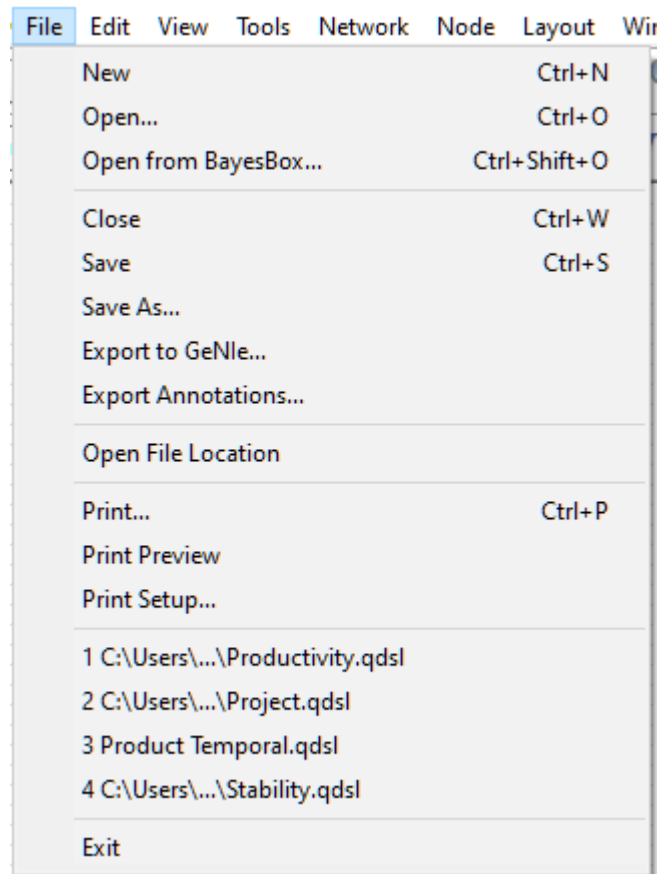
There are three ways in which you can save a file in QGeNIe:

- Choose *Save* or *Save As* from the [File menu](#)
- Click on Save () button from the [Standard Toolbar](#)
- Use the *CTRL+S* (*Save*) shortcut

The difference between *Save* and *Save As* is that *Save As* lets you store the file under a different name, so that you can keep the original model file intact. *Save* will store the changes in the original file. However, if you are working on a new file, then *Save As* is the only option and *Save* converts to *Save As*.



5.6.2 File menu



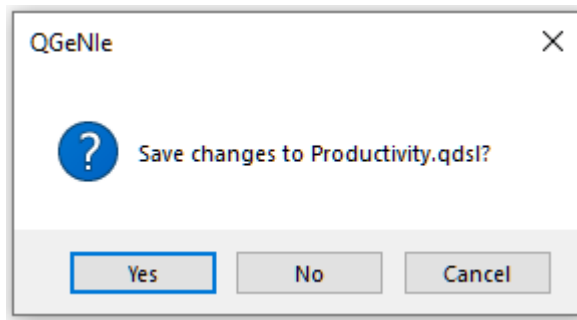
The *File* menu offers the following commands:

New starts a new QGeNIe model and opens it in a new *Graph View* window. This command can be also invoked by pressing the New (📄) tool on the [Standard Toolbar](#) or using the *CTRL+N* shortcut.

Open... starts the Open dialog that allows you to open an existing model, been saved previously on the disk. This command can be also invoked by pressing the Open (📁) tool on the [Standard Toolbar](#) or using the *CTRL+O* shortcut.

Open from BayesBox opens an existing model from a BayesBox-powered model repository. This command can be also invoked by using the *CTRL+SHIFT+O* shortcut.

Close closes all windows of the current model. If there are any changes to the currently opened model that have not been saved, QGeNIe will warn you using the following dialog box



If you want to save the changes that you have made since you have last saved the model, click the *Yes* button or press *Enter*. If you want to discard them, click the *No* button. If you have second thoughts about exiting QGeNIe, click *Cancel*.

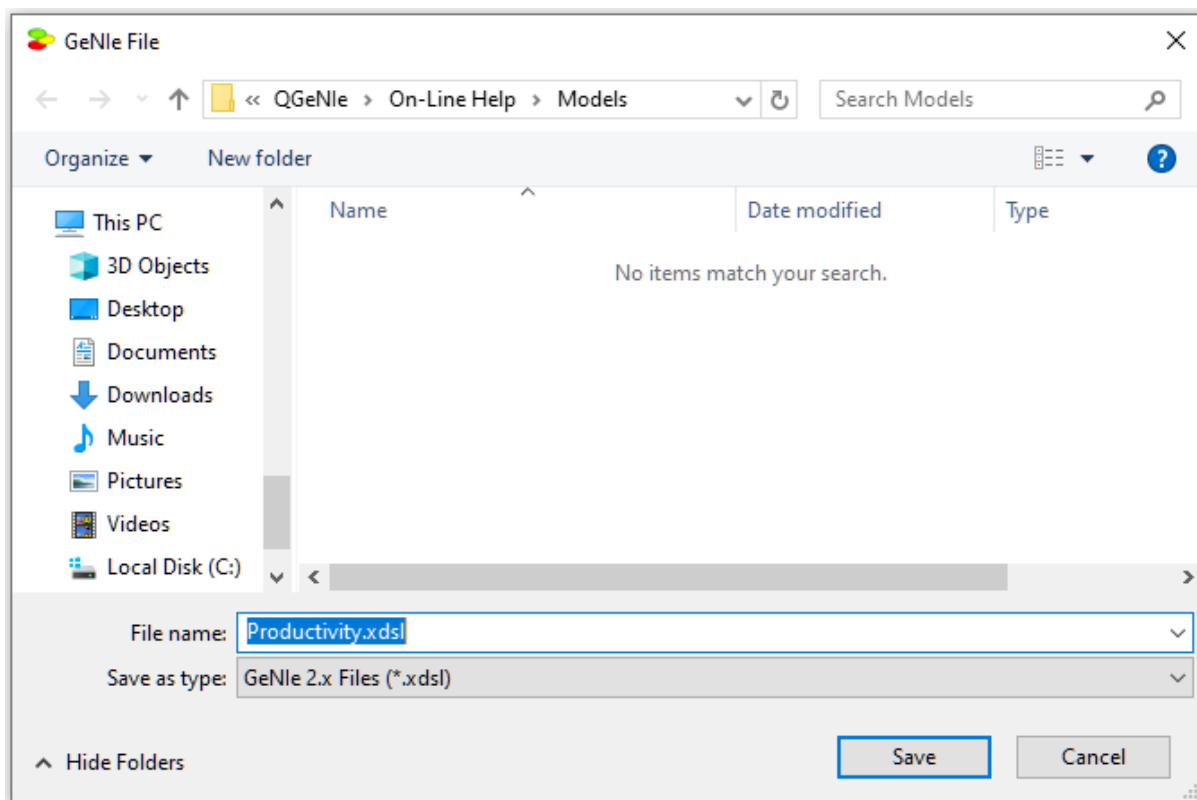
Save saves the currently opened model using the current file name and file format. If the document is new (i.e., if it has never before been saved), this command is equivalent to the *Save As...* command.

This command can be also invoked by pressing the Save (💾) tool on the [Standard Toolbar](#) or using the *CTRL+S* shortcut.

Save As... starts a *Save As* dialog that allows you to save the currently opened document to a newly specified file.

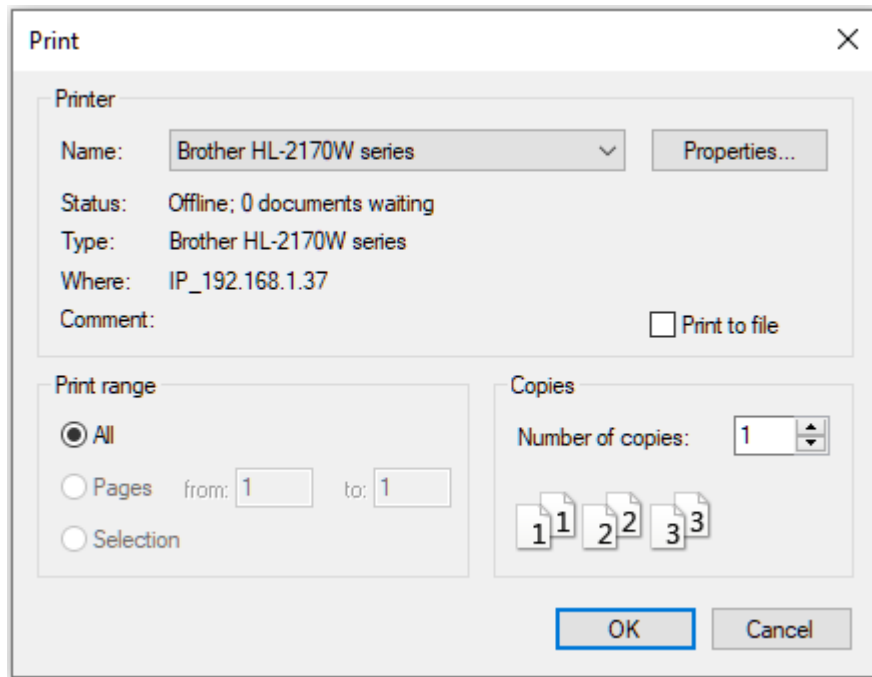
The *Open Network* and *Save/Save As* commands are discussed in more detail in the [introduction](#) to this section.

Export to GeNIe... opens a dialog that allows to save the current model as a GeNIe file. The main reason why you might want to do this is a further refinement or examination of the model in a fully quantitative setting. Many users start with a qualitative model, which is very easy to build, and subsequently export the model to GeNIe to refine it further.

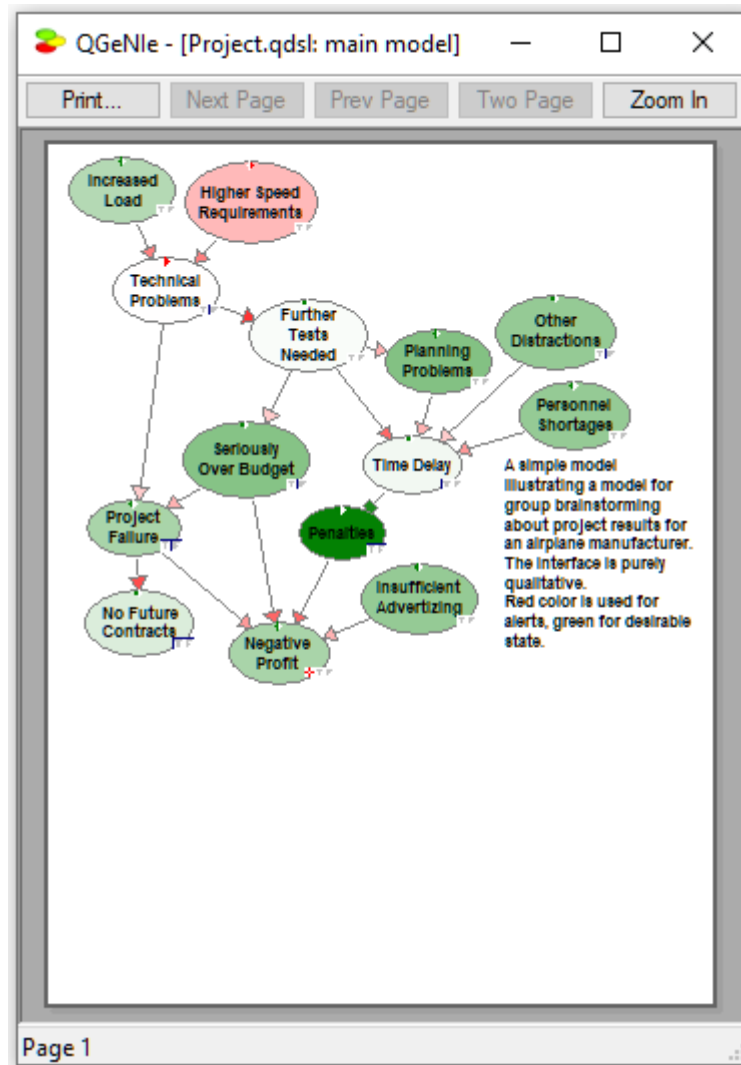


Print... prints the current *Graph View* window. This command can be also invoked using the *CTRL+P* shortcut.

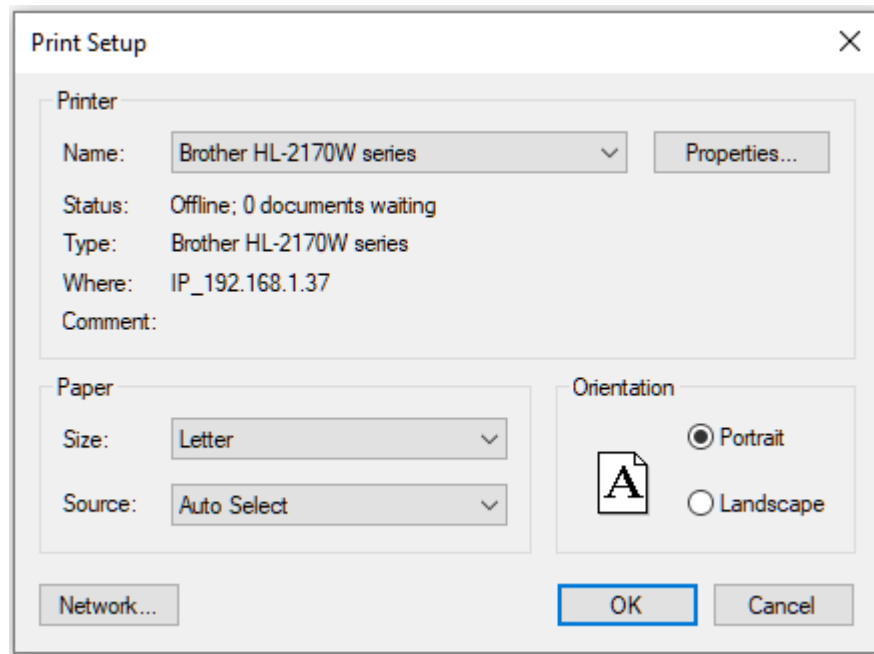
The following dialog box allows you to modify some of the printing options, such as choose the printer, the range of pages to be printed, the number of copies to be printed, and other printer properties (through *Properties...* button).



Print Preview displays the content of the current *Graph view* window on the screen as it would appear when printed. When you choose this command, the main window is replaced with a print preview window in which one or two pages will be displayed in their printing format. The print preview toolbar offers you options to view either one or two pages at a time; move back and forth through the document; zoom in and out of pages, and initiate the print job (bypassing the *Print* command).



Print Setup... opens a dialog that allows for selecting the printer and printer and its properties, including the paper size, source, and orientation. This command presents a *Print Setup* dialog box, where you specify the printer and its connection. The appearance of the dialog box below may vary depending upon your system configuration.



QGeNIe displays the names and disk locations of eight most recently used models. You can load them by choosing their names from the menu, bypassing the *Open* command.

Exit ends your QGeNIe session and exits the program.

5.6.3 QDSL file format

The XML Schema for QGeNIe's native QDSL file format can be found at the following location: <https://support.bayesfusion.com/docs/>.

5.7 Inference algorithms

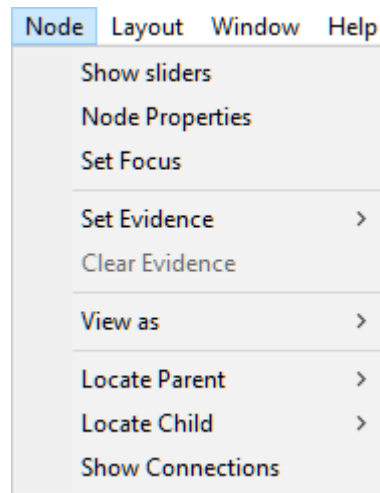
5.7.1 Introduction

The default QGeNIe algorithm for model updating, the clustering algorithm is exact, very efficient, and fast.

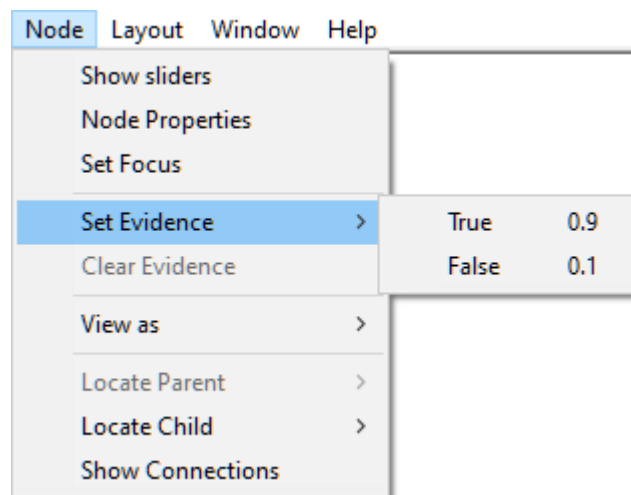
Because probabilistic reasoning is worst-case NP-hard, a QGeNIe user may encounter networks for which the memory requirements or the updating time may be not acceptable. In such a case, it may be necessary to choose a different algorithm. Even though we have never experienced the need in practice to change the default algorithm, it is conceivable that the clustering algorithm will run out of memory or will take too much time. In such a case, we recommend trying the Relevance-based decomposition algorithm, which is also an exact algorithm making use of the structural properties of the model at hand and with some overhead can make the model tractable. If this does not work, we recommend the EPIS sampling algorithm, which is quite likely the most efficient stochastic sampling algorithm for discrete Bayesian networks. While it is an approximate algorithm, it shows excellent convergence rates and should deliver precision that will be satisfactory. Tradeoff between precision and computation time can be controlled by selecting the number of samples.

5.7.2 Node menu

We have deferred the description of the commands *Set Evidence*, *Clear Evidence*, *Control Value*, *Set Focus* and *Clear Focus* to the current section, which offers background information to Bayesian network algorithms. We reproduce the *Node* menu, which is to a large degree duplicated by the node pop-up menu, below:



Set Evidence submenu allows for setting the node state, which amounts to observation of that node state. The submenu is active only if there is one (and only one) node selected in the *Graph View*.



To select a state of the currently selected node, select this state on the submenu listing the states and release the mouse button. The state will have a check mark next to its name. The node will from that point on be marked with a corresponding icon, indicating that one of the states of this node (*True* or *False*) has been observed. To change the node back to the unobserved state, choose *Clear Evidence* from the *Node* menu. Setting evidence for a node can be accomplished in three other ways: (1) By choosing *Set Evidence* in the node context menu, which is similar to the *Node* menu, (2) double clicking on one of the state icons (*T* or *F*), or (3) pressing the letter *T* or *F* (for *True* and *False*) once the node has been selected.

Control Value submenu works precisely like the *Set Evidence* submenu but it stands for controlling rather than observing the value. Controlling means that the value has been set from outside. QGeNIe's implementation of controlling the value follows so called arc-cutting semantics, which means that the incoming arcs of the controlled node become inactive (nothing inside the model influences the node, as its value is set from outside). QGeNIe shows these inactive arcs as inactive by dimming them. See [Controlling values](#) for more information about this functionality.

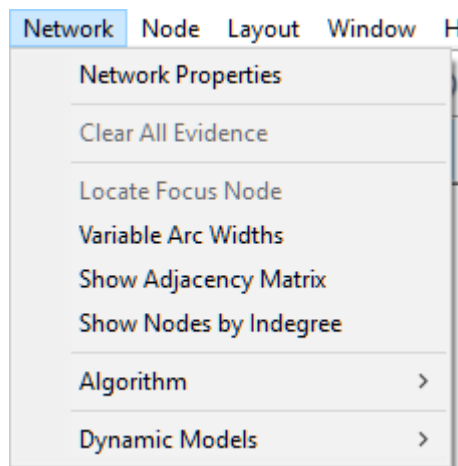
Clear Evidence command (for controlled nodes, this command is called *Release Value*) is active only if there are nodes selected in the *Graph View* and at least one of these node has been previously observed (or controlled). *Clear Evidence* un-observes a state, i.e., it reverses the effect of the *Set Evidence* command. *Release Value* un-controls a state, i.e., it reverses the effect of the *Control Value* command. The check mark next to the state name will disappear. The status icon will be adjusted accordingly.

The *Set Focus* command is active only if there is exactly one node selected in the *Graph View*. It allows you to set the status of the selected node(s) to be the *Focus*. The node will from that point on be equipped with the *Focus* (🔍) status icon, indicating that the node is the focus of reasoning. This is important in the value of information and value of control calculations.

The *Clear Focus* command is active only if the node selected is the focus. *Clear Focus* reverses the effect of the *Set Focus* command. The *Focus* (🔍) status icon will disappear.

5.7.3 Network menu

The *Network* menu allows for performing operations that relate to the entire network. It offers the following commands:



Network Properties invokes the *Network Properties* sheet for the current model. The network property sheet can also be invoked by double-clicking in any clear area on the main model *Graph View* window. See [Network properties](#) section for more information.

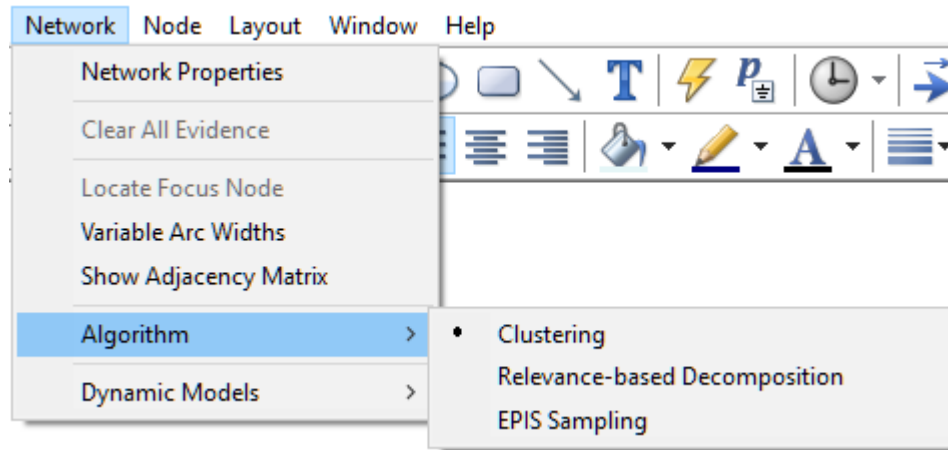
Clear All Evidence allows for retracting all evidence in one simple step rather than doing it for each individual node.

Locate Focus Node finds the node designated as focus.

Variable Arc Widths helps with analyzing the model structure by displaying links between nodes (the arcs) in variable width, corresponding to the magnitude of their influence.

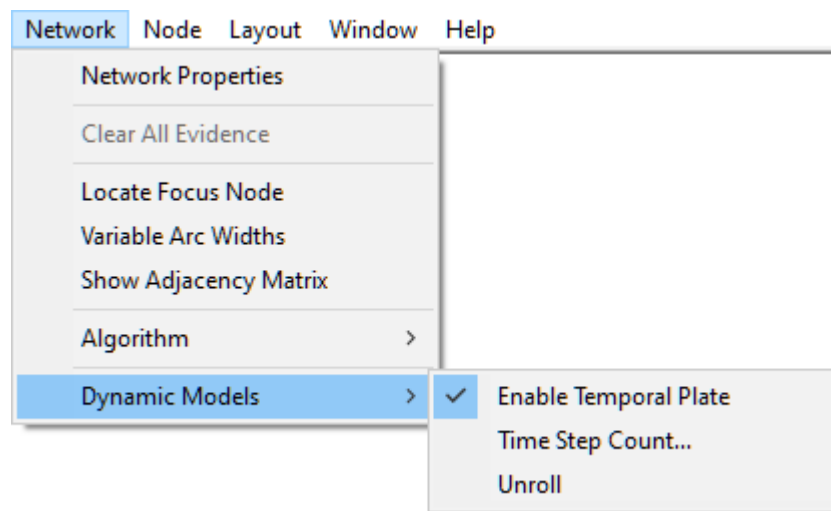
Export Annotations collects all annotations, descriptions, and comments from the entire model and places them in a single text file that can be analyzed outside of QGeNIe.

Algorithm submenu allows for choosing the default algorithm for Bayesian networks:



The menu allows for choice of the default algorithm (marked with a bullet). Whenever updating takes place, the current default algorithm will be executed.

Dynamic Models submenu groups all operations on dynamic Bayesian networks



This submenu is discussed in detail in the section of *Dynamic Bayesian networks*.

5.7.4 Clustering algorithm

Clustering algorithm is the fastest known exact algorithm for belief updating in [Bayesian networks](#). It was originally proposed by Lauritzen and Spiegelhalter (1988) and improved by several researchers, e.g., Jensen et al. (1990) or Dawid (1992). Our implementation of the clustering algorithm is very efficient and lightning fast. In fact, it is quite possibly the fastest implementation of this algorithm in existence.

The clustering algorithm is the default algorithm used by QGeNIe. The clustering algorithm is QGeNIe's default algorithm and should be sufficient for most applications. Only when networks become very large and complex, the clustering algorithm may not be fast enough. In that case, it is suggested that the user choose an approximate algorithm offered by the program, the EPIS-BN algorithm (Yuan & Druzdzel, 2003).

5.7.5 Relevance-based decomposition

Relevance-based decomposition is an exact algorithm based on the clustering algorithm that performs a decomposition of the network when the network is very large. The algorithm was described in (Lin & Druzdzel, 1997). Relevance-based decomposition extends the boundary of what is computable, while gracefully changing into the clustering algorithm for small networks. Because there is some overhead related to decomposition, we suggest that this algorithm be used only when the clustering algorithm cannot handle your networks.

5.7.6 EPIS Sampling

The *Estimated Posterior Importance Sampling (EPIS)* algorithm is described in (Yuan & Druzdzel 2003). This is quite likely the best stochastic sampling algorithm for discrete Bayesian networks available. It produces results that are even more precise than those produced by the AIS-BN algorithm and in case of some networks produces results that are an order of magnitude more precise. The EPIS-BN algorithm uses loopy belief propagation to compute an estimate of the posterior probability over all nodes of the network and then uses importance sampling to refine this estimate. In addition to being more precise, it is also faster than the AIS-BN algorithm, as it avoids the costly learning stage of the latter.

5.8 Keyboard shortcuts

File operations

CTRL+N: Create a new network

CTRL+O: Open an existing network

CTRL+P: Print the current network

CTRL+S: Save the currently active file to disk

CTRL+W: Close the current network

Layout of elements in the *Graph View*

CTRL+G: Toggle display of grid lines

CTRL+L: Highlight selected model elements

CTRL+SHIFT+G: Toggle auto alignment of elements to grid

F8: View nodes as bar charts

SHIFT+F8: View nodes as icons

Finding / selecting nodes / spreadsheets

CTRL+F: Find a node

CTRL+A: Select all elements

CTRL+SHIFT+A: Select all nodes

CTRL+ALT+A: Select all submodels

Show / hide windows

CTRL+T: Toggle display of the *Tree View* window

CTRL+U: Toggle display of the *Output* window

CTRL+ALT+C:

F11: View network full-screen (hides all views, menus, and toolbars)

F12: Zoom to fit window

CTRL+F12 / *CTRL + **: Zoom to 100%

CTRL+PLUS: Zoom in

CTRL+ MINUS: Zoom out

Renaming objects

F2: Rename object

Editing

CTRL+B: Bold font

CTRL+I: Italic font

CTRL+C: Copy

CTRL+V: Paste

CTRL+X: Cut

Using QGeNle

6 Using QGeNIe

6.1 Introduction

This section presents various modules of QGeNIe from the point of view of their function. It is an alternative view to the one presented in the previous section, which focused on QGeNIe's building blocks.

6.2 Applications of qualitative probabilistic modeling

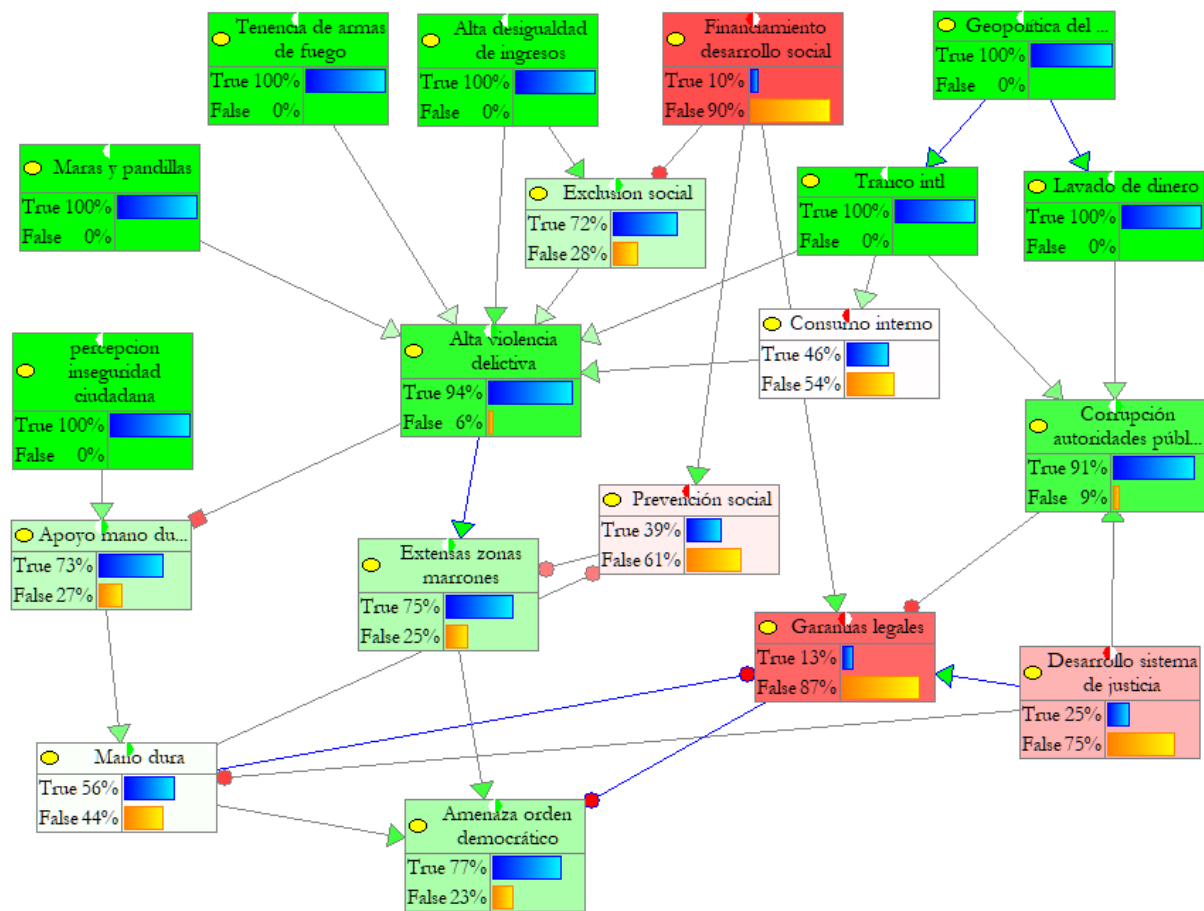
A qualitative modeling environment like QGeNIe has many applications. In this section, we give three example of applications that we have encountered.

A. Rapid Prototyping

Even if the goal is to build a fully quantified Bayesian network, QGeNIe is extremely valuable as an environment for rapid prototyping. Models developed quickly within QGeNIe can be exported to GeNIe for further elaboration and parameter refinement.

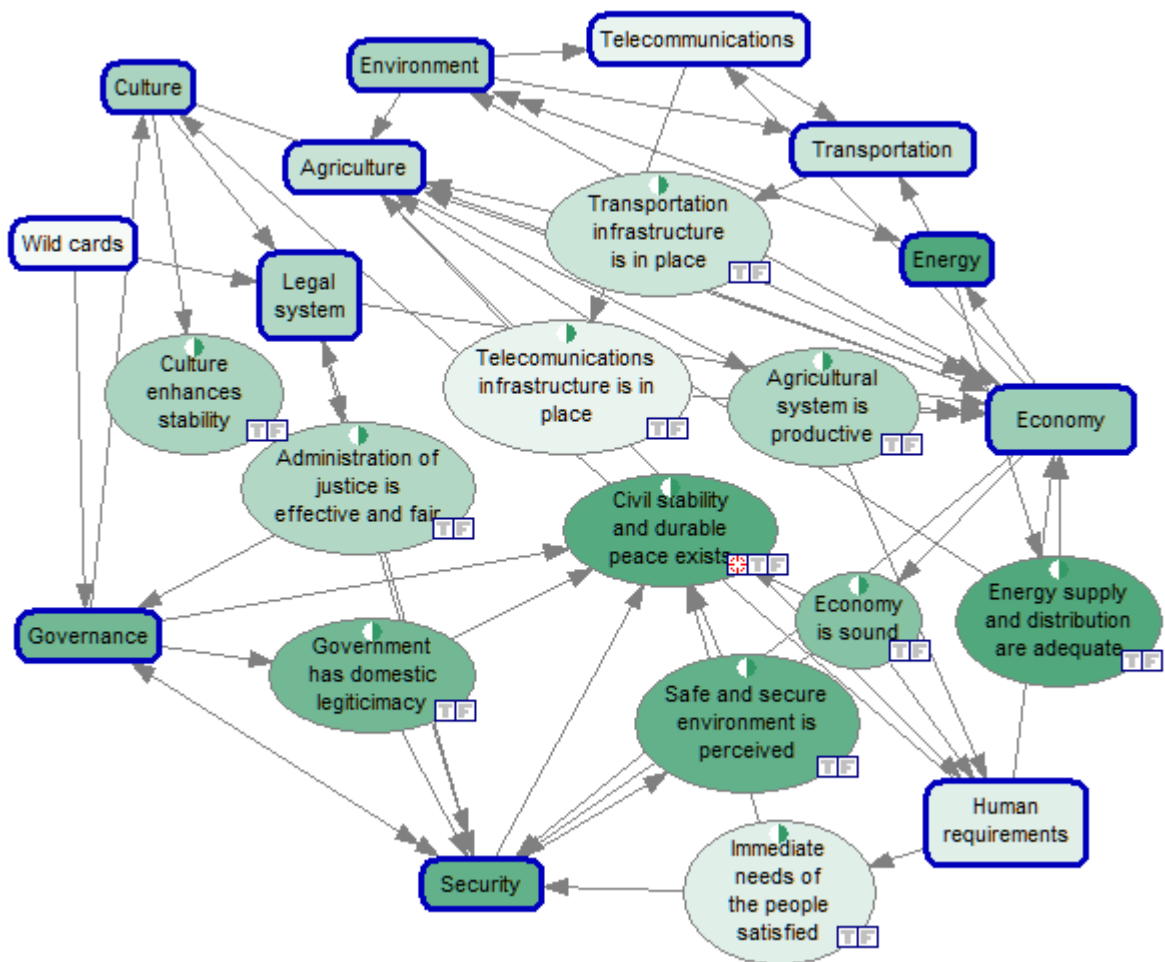
B. Modeling in “Soft Sciences”

As late Herb Simon convincingly argued (Simon 1996), “soft sciences” are really hard sciences. Models of social sciences systems are really hard to build, because very often little is known about them. Still, there is ample evidence from behavioral decision theory (e.g., Dawes, 1988) that even simplest mathematical models typically perform better than unaided human intuition. QGeNIe has been applied to projects that are truly hard to tackle with formal methods. One example that we are aware of is an application in modeling security in Costa Rican cities (Pérez-Liñán, 2008). The screen shot below shows a simple QGeNIe model used in the study.



C. Group Decision Making

The original inspiration for QGeNIe was provided by our collaboration with policy analysts at the United States Naval War College, Bradd Hayes and Theo Gemelas. A model developed there (see the screen shot below, the foundations for this model were developed in (Hayes & Sands, 1997), Figure 5-1, page 101) consisted of 99 variables organized into 12 submodels. The goal of this model was to bring together experts from a variety of areas relevant to stability of a region (in this case, the Black Sea region). The experts know some aspect of the problem (e.g., economy, culture, or energy) but not everything. They may help with building various parts of the model. Asking “what if” questions of the complete model allows each of the individual experts to verify their intuitions but also see how manipulations propagate through those submodels that they did not know much about. A model of this complexity cannot typically be understood completely by a single expert. Presence of multiple experts in a room, each of whom is familiar with a parts of the model, usually helps with verifying model assumptions and obtaining insight into the problem and consequences of the resulting decisions.



6.3 Bayesian networks

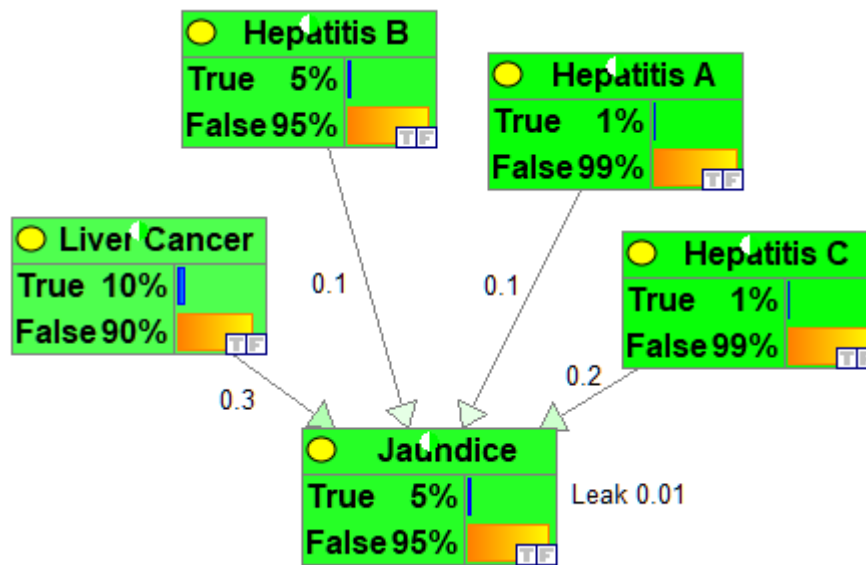
6.3.1 Building a qualitative Bayesian network

Building a qualitative [Bayesian network](#) in QGeNIe is demonstrated step for step in section [Hello QGeNIe!](#)

6.3.2 Simple interaction patterns

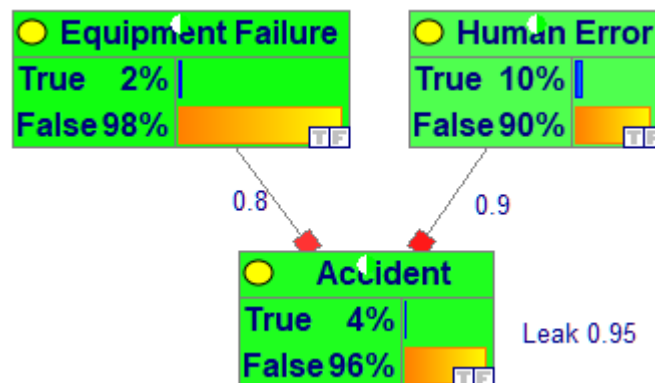
The [DeMorgan](#) model of interactions between a variable and its parents is flexible and allows for modeling a wide variety of possible interaction patterns. In this section, we show two simple, popular interaction patterns: Noisy-OR and Noisy-AND, familiar to those modelers who have worked with canonical models in the context of Bayesian networks.

The Noisy-OR interaction takes place when all links from parents to the child node are *Causes*. Consider the following example:



The values of prior probabilities are shown in the parents' prior marginal distributions, the value of strengths of influences are marked on the arcs, the value of *Leak* in the node *Jaundice* is 0.01. The model reproduces precisely the NoisyOR.xdsl model in [BayesFusion's interactive model repository](#).

The Noisy-AND interaction takes place when all links from parents to the child node are *Requirements*. Consider the following example:



The values of prior probabilities are shown in the parents' prior marginal distributions, the value of strengths of influences are marked on the arcs, the value of *Leak* in the node *Accident* is 0.95. The model reproduces precisely the NoisyAND.xdsl model in [BayesFusion's interactive model repository](#).

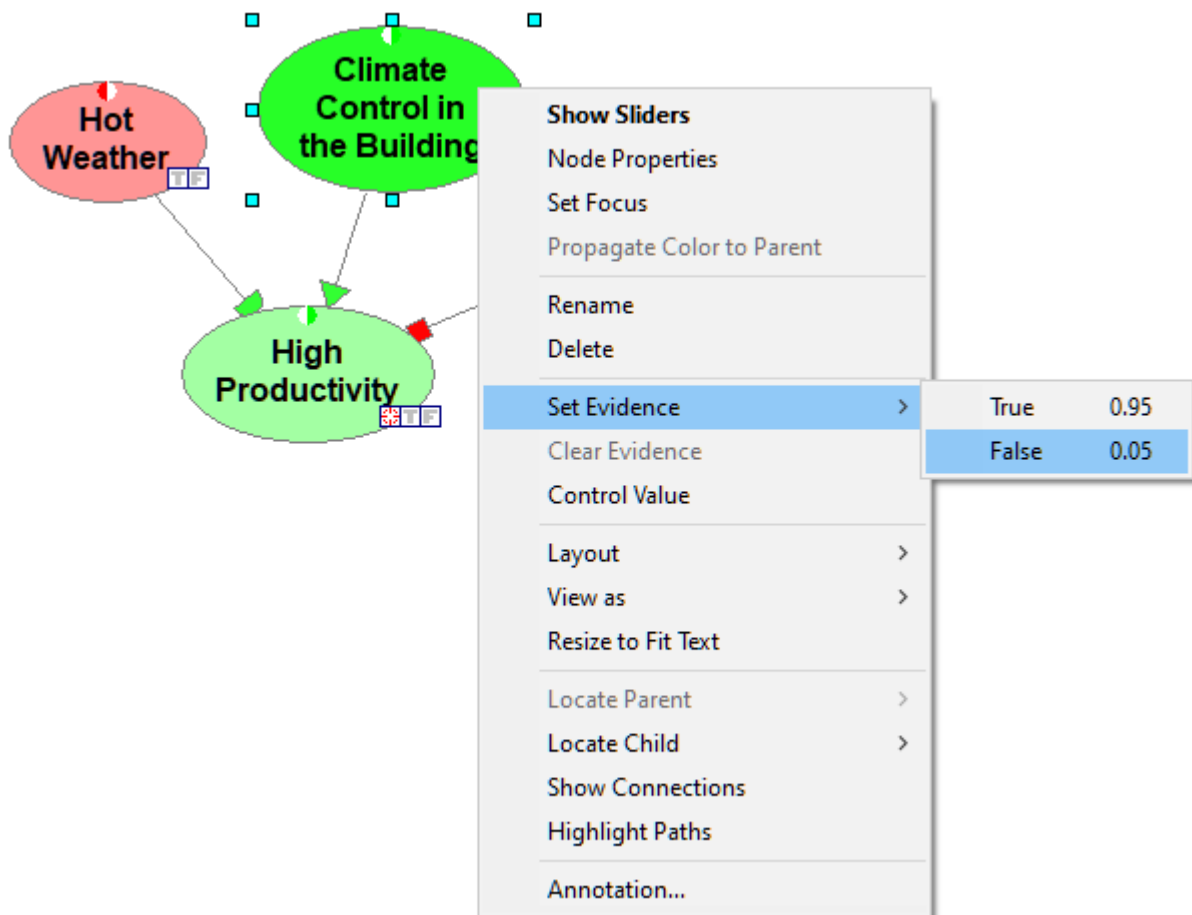
Introduction of *Barriers* in Noisy-OR models is straightforward: The resulting model will be still a Noisy-OR but the parents' (*Barriers*) states can be viewed as reversed. The same happens when we introduce *Inhibitors* in the Noisy-AND model: The resulting model will be still a Noisy-AND but the parents' (*Inhibitors*) states can be viewed as reversed. Mixing all four types of causal influences in one model makes the model no longer a pure Noisy-OR or a Noisy-AND but a formally correct combination of noisy OR and AND functions that allows for modeling a complex interaction of parents.

6.3.3 Entering and retracting evidence

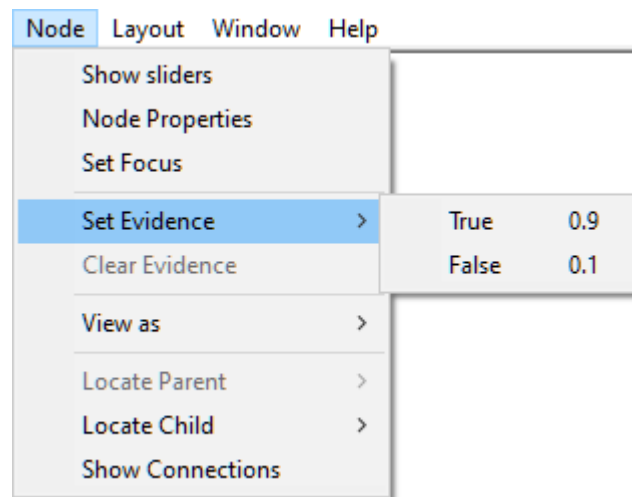
Entering observations (evidence) is one of the basic operations on a probabilistic model. It amounts to adjusting the model to a new situation, one in which more information is available. It allows to query the system subsequently about the new, posterior probability distributions. If you have gone through [Building a Qualitative Bayesian network](#), you might remember that you have already entered evidence for the *Hot Weather* and *Climate Control in the Building* nodes. Let us go through this again.

You may load the model *Productivity.qdsl* created in [Building a Qualitative Bayesian network](#) from the *Example Networks* folder.

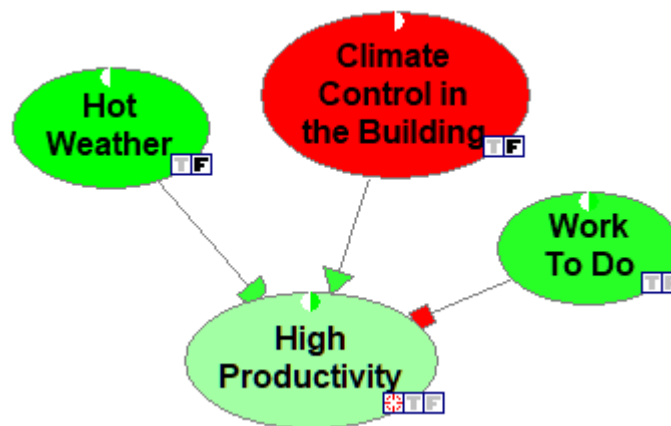
To enter evidence into your model, right-click on the node in question (in the picture below, node *Climate Control in the Building*) and choose *Set Evidence*.



Alternatively, select the node in question and choose *Set Evidence* submenu from the [Node Menu](#).

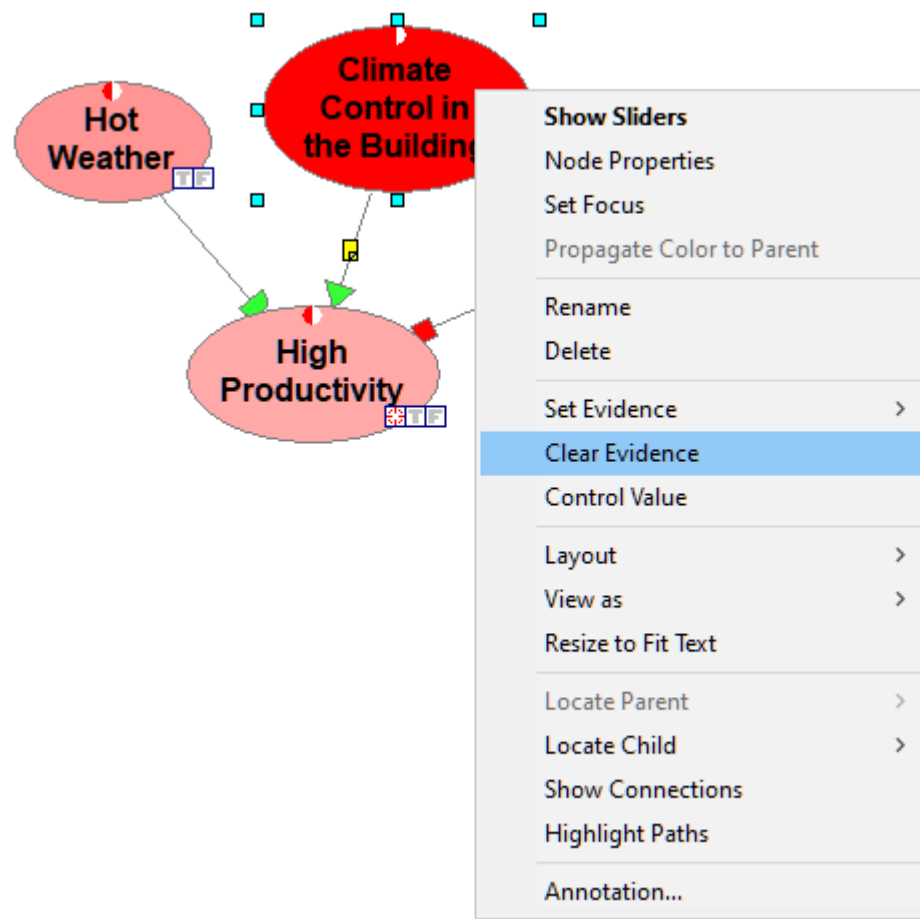


Yet another way of entering evidence is double-clicking on one of the small square icons (TF) in the lower-right corner of a node. In the image below, we double-clicked on the icon *False* (F) in both *Climate Control in the Building* and *Hot Weather* nodes.

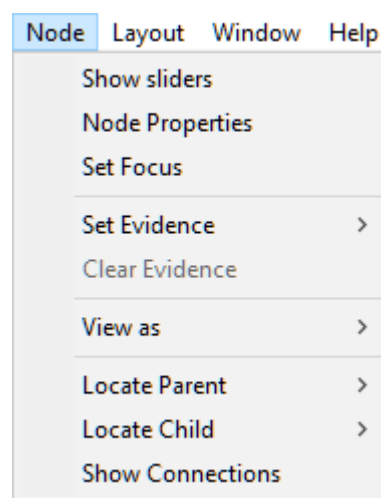


Finally, perhaps the simplest way of entering evidence is pressing the letter *T* or *F* (for *True* and *False* respectively) once the node has been selected.

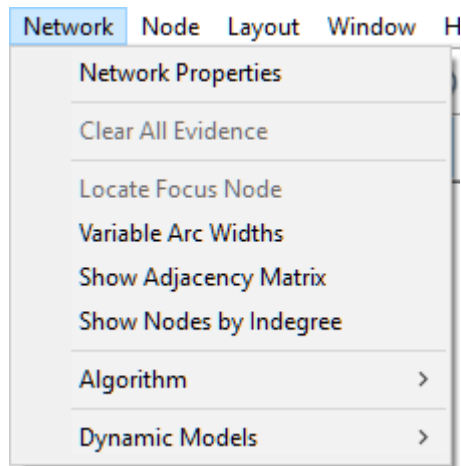
To retract evidence, double-click on the bold icon in a node with previously entered evidence (F) shown in the image above, press the letter *C* (for *Clear*) once the node has been selected, or right-click on the node and choose *Clear evidence*.




Alternatively, select the node in question and choose *Clear evidence* from the [Node Menu](#).



You can also retract all evidence by choosing *Clear all evidence* from the *Network* menu:

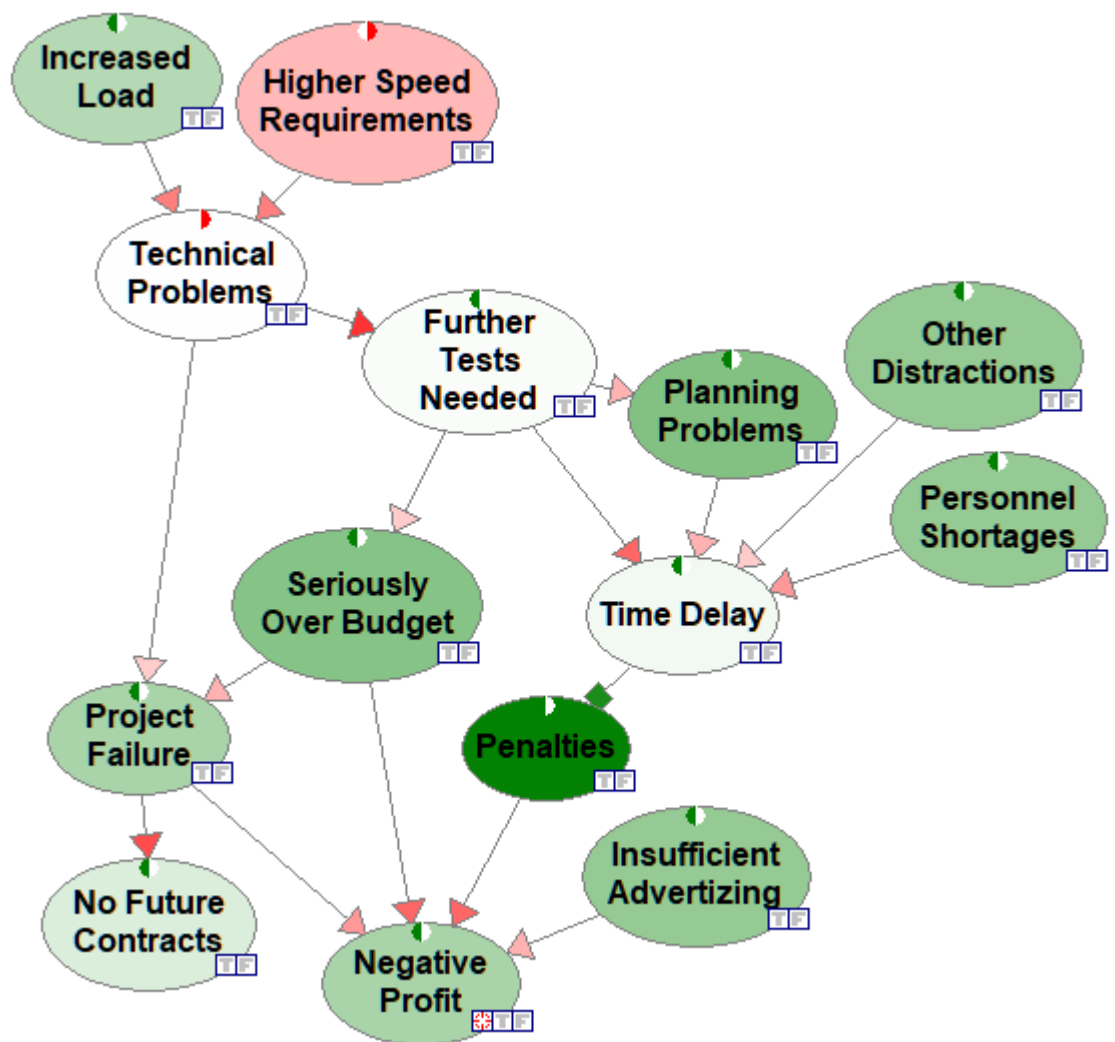


To enter different evidence instead of retracting evidence altogether, just double click on a different state icon () or set different evidence from the *Set Evidence* sub-menu.

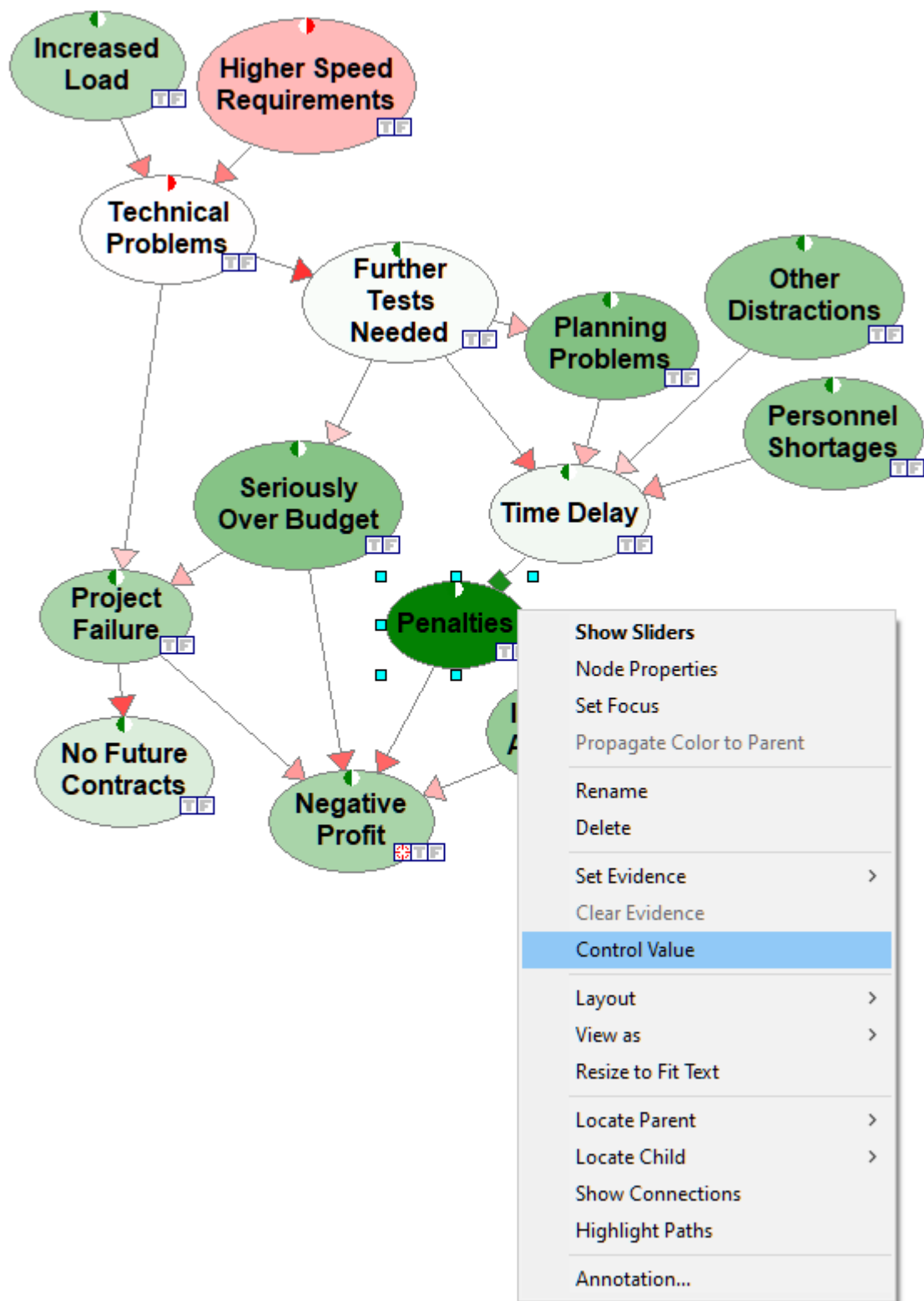
6.3.4 Controlling values

In causal probabilistic models, there is an additional class of inference problems: Predicting the effects of external intervention. In the context of [Bayesian networks](#), computing the effects of observations amounts to belief updating after setting evidence for the observed network variables. The effect of intervention, on the other hand, is a change in the network structure, related to external manipulation of the system modeled by the network, followed by setting the values of the manipulated nodes and updating beliefs.

We will explain control values with the help of the following example (the model, `Project.qdsl`, is included among the example models) describing a set of variables that may influence profits of an airplane manufacturer.



Imagine that we are conducting this analysis at the very initial stage of negotiations with the client and would like to know the impact of negotiating in the contract that no penalties can be imposed on the manufacturer. This is a typical question involving controlling the values of variables. No matter what happens, we impose zero penalties on the system, which means that we impose the probability of zero on penalties. To predict the effect of this manipulation, we right-click on the node *Penalties* and choose *Control Value*.



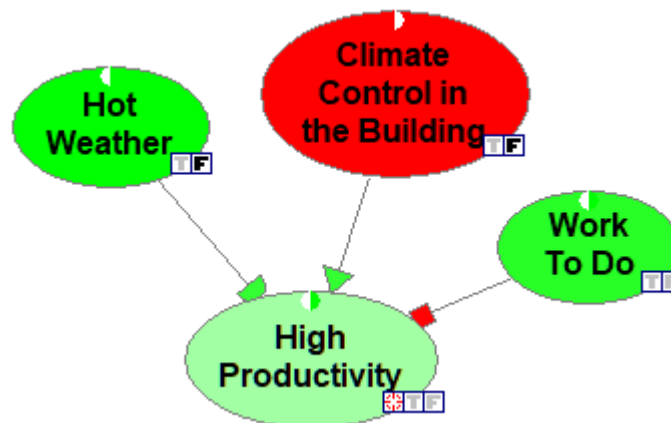
Notice that the intervention only changes the posterior probabilities of the descendants of the controlled node. The control value operation is not available for those nodes that have observed or manipulated descendants. Controlling the value of a descendant of an observed node would lead to a theoretical problem, which one could summarize briefly as a desire to modify the past.

6.3.5 Viewing results

QGeNIe is all about exploring models and viewing results of calculations after observing evidence and performing manipulations. There are several ways of viewing interesting information from a model.

Probabilities of various propositions

Probabilities of propositions can be viewed in several ways. The easiest, one that does not require any action, is observing the colors of nodes, which represent probabilities of truth (of falsity -- this can be set in node properties) of the proposition represented by the node. The *Productivity* model, used in the *Building a qualitative Bayesian network* section, shown below, shows the probabilities of each of the four propositions represented by the four model variables

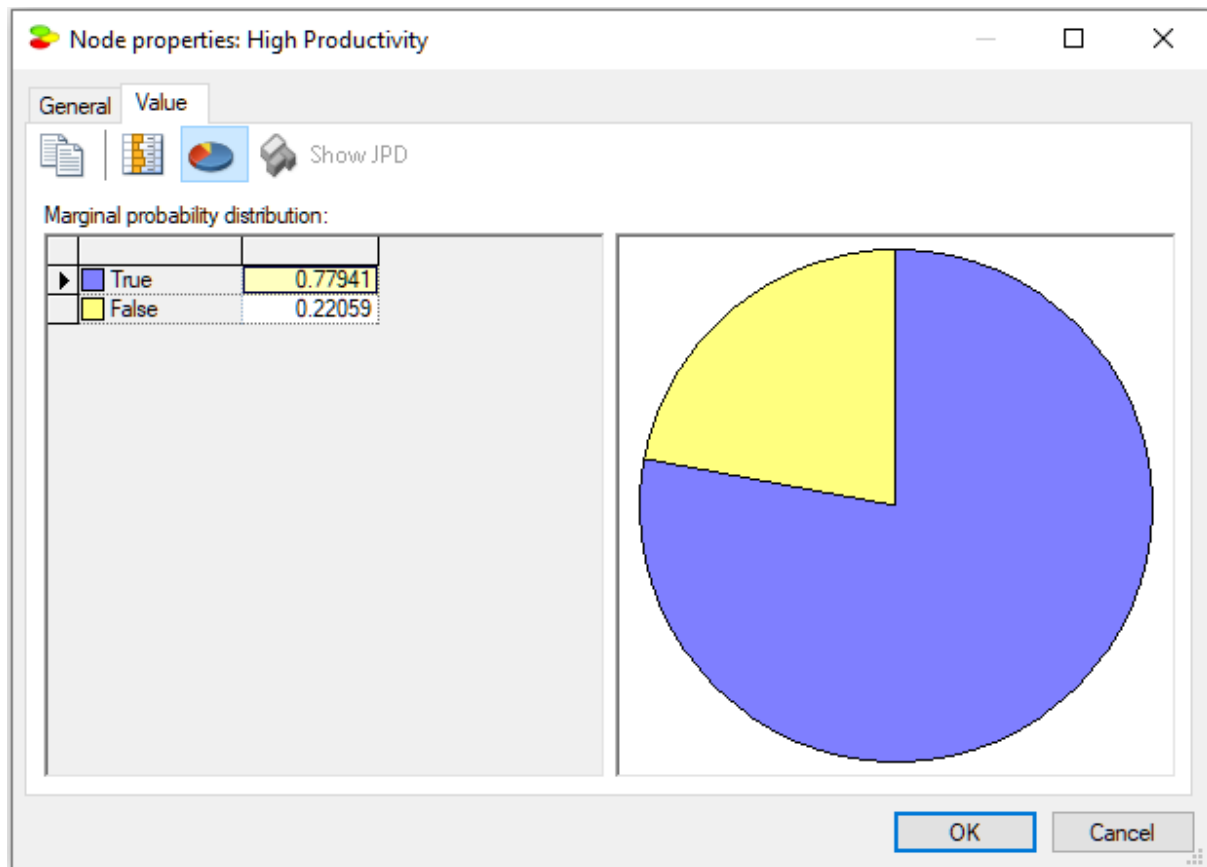


Climate Control in the Building is marked as observed to be false and it is also bright red, showing that the probability of the climate control in the building is zero (either because of absence of an AC unit or because it is broken). *Hot Weather* is marked as observed to be false and it is also bright green (please note that the colors for the node *Hot Weather* are reversed, so green denotes falsity of the proposition), showing that the probability of the weather being hot is zero. *High Productivity*, the focus variable in the model, has a probability between 0.5 and 1.0 judging by the node color. The small circle in the upper part of the model shows probability 0.5 and 1.0 by its white and intensive green halves. Clearly, the node color is closer to green than it is to white.

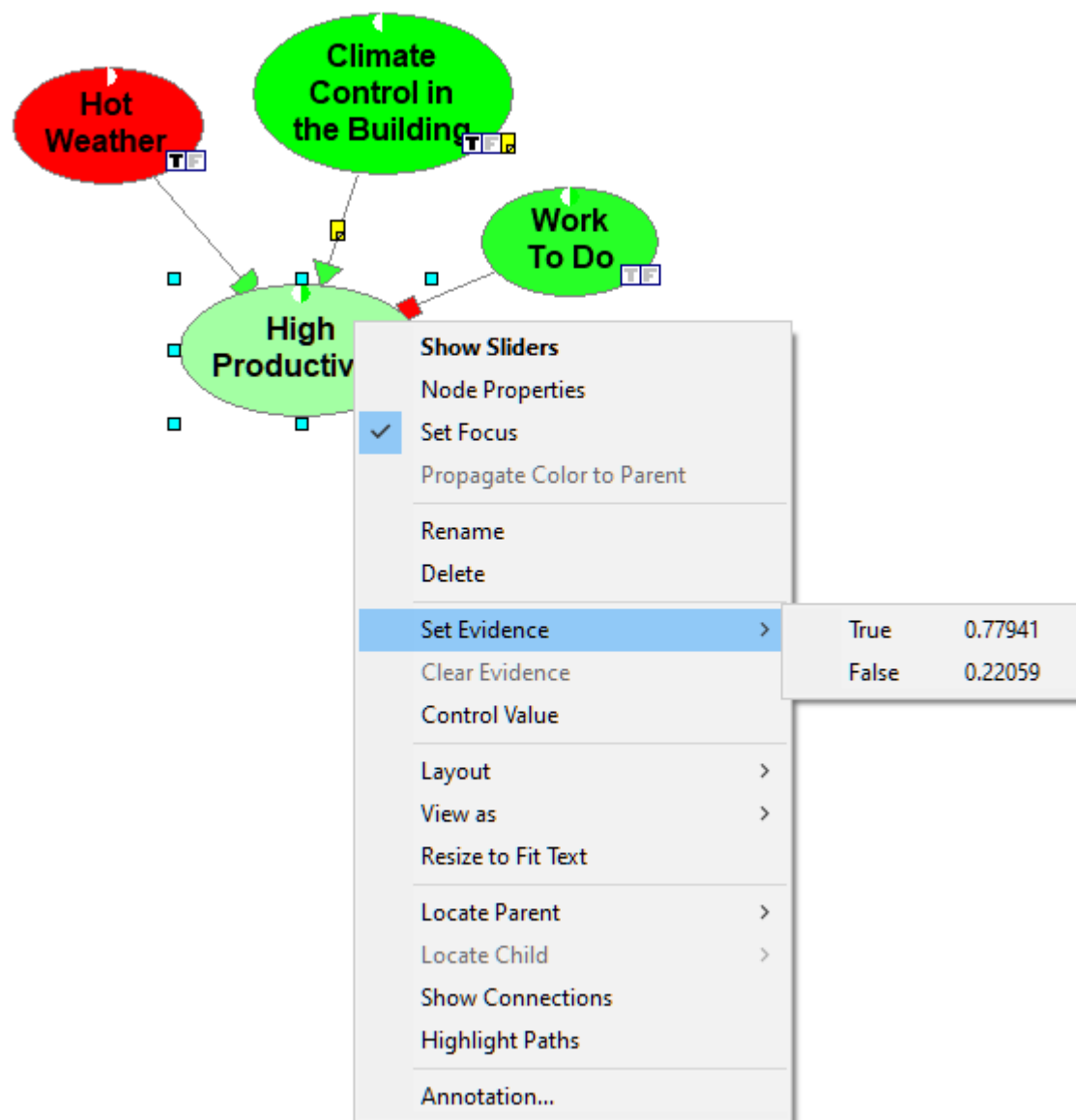
It is possible to see the exact numerical values of the probabilities by opening the properties of any node. Let us open the properties of the node *High Productivity*:



The *Value* tab shows the probability of truth of *High Productivity* is 0.74929816.



You can also right-click on the node and choosing *Set Evidence*. GeNIe shows a list of states of the node along with their probabilities, when these are available.



There are other model elements that belong to broadly-understood results that you can view through QGeNIe interface. We will discuss these in the following sections of the manual: [Structural analysis](#), [Value of information/manipulation](#), and [Inference in DBNs](#).

6.3.6 Structural analysis

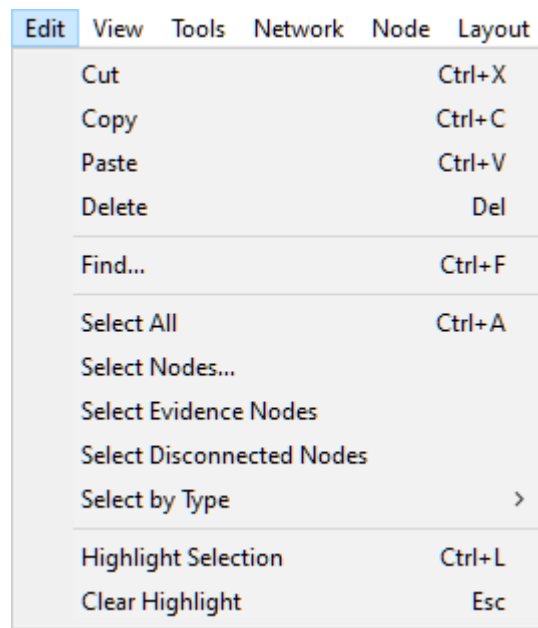
One of the important elements of probabilistic modeling is the ability of directed probabilistic graphs to represent the causal structure of the modeled domain. The structure itself is very valuable and is an important source of insight. Models built by means of QGeNIe can be examined structurally. An important element of this analysis is the structure itself, viewing the strengths of influences and pathways through the graph. This section describes tools for the analysis of the graph structure.

Dimming unnecessary arcs

Dimming unnecessary arcs, discussed earlier in this manual, is a simple tool that allows for finding one class of modeling errors, errors of omission. One such error is having an influence arc that has zero numerical effect. QGeNIe shows such arc as dimmed, as they are unnecessary and are quite likely the result of an error. Whenever you see a dimmed arc, please have a look at the child node and its parents - chances are that it is not that the arc is unnecessary but rather that you have forgotten to set the numerical value of influence of the parent node on the child node.


Locating disconnected nodes

When modeling, many a user create a collection of nodes and then connect them by means of arcs. When the model under construction is sufficiently large, it is not uncommon to forget about some of the nodes and leave them unconnected to the rest of the network. Locating disconnected nodes is a functionality that helps to find such nodes so that we can give them more attention. To find disconnected nodes, please select *Select Disconnected Nodes* from the *Edit Menu*:

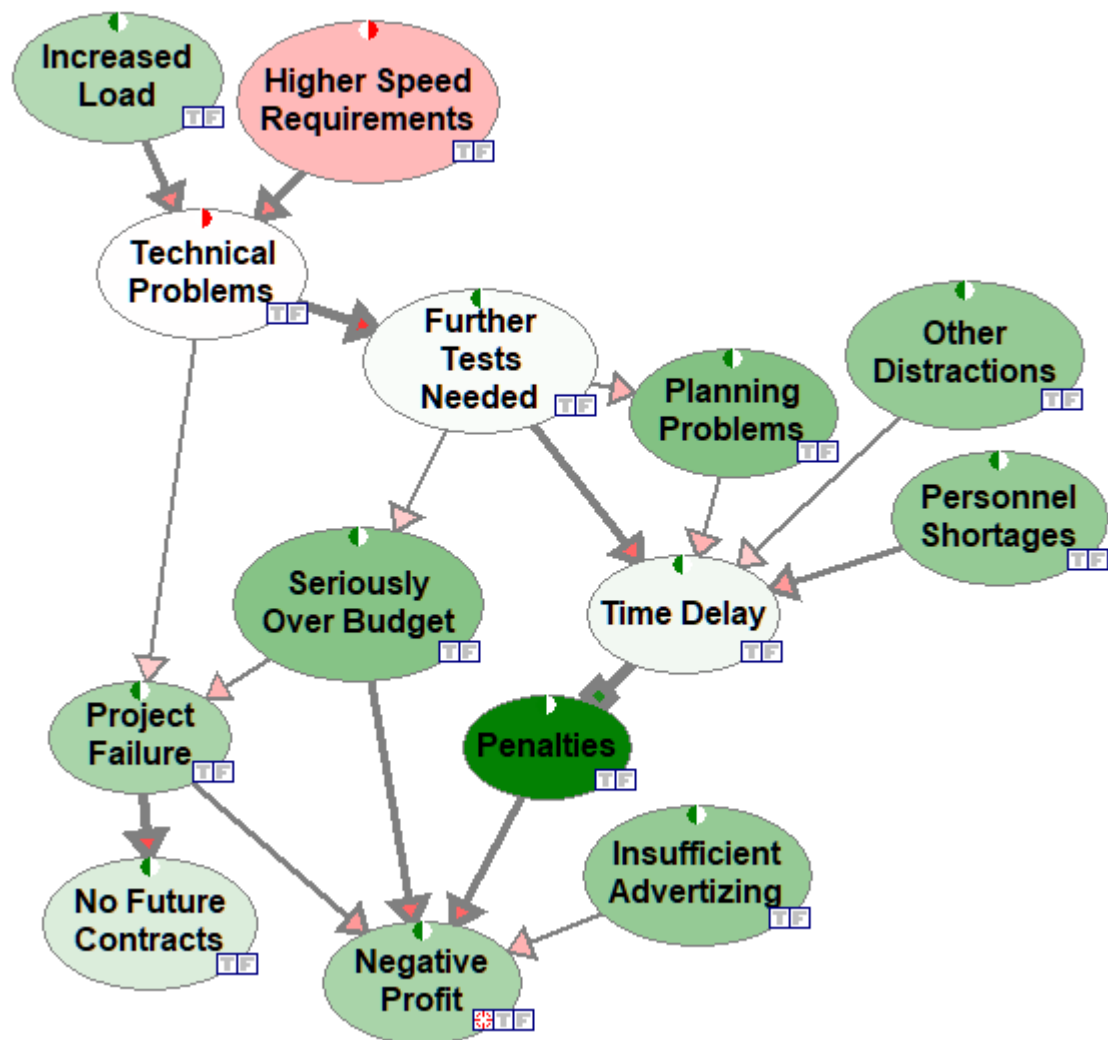


If there are no disconnected nodes in the model, this choice is going to be grayed out. Nodes selected can be subsequently highlighted (choice *Highlight Selection* or *Ctrl-L* in the *Edit Menu*) and easily located in the *Graph View* visually.

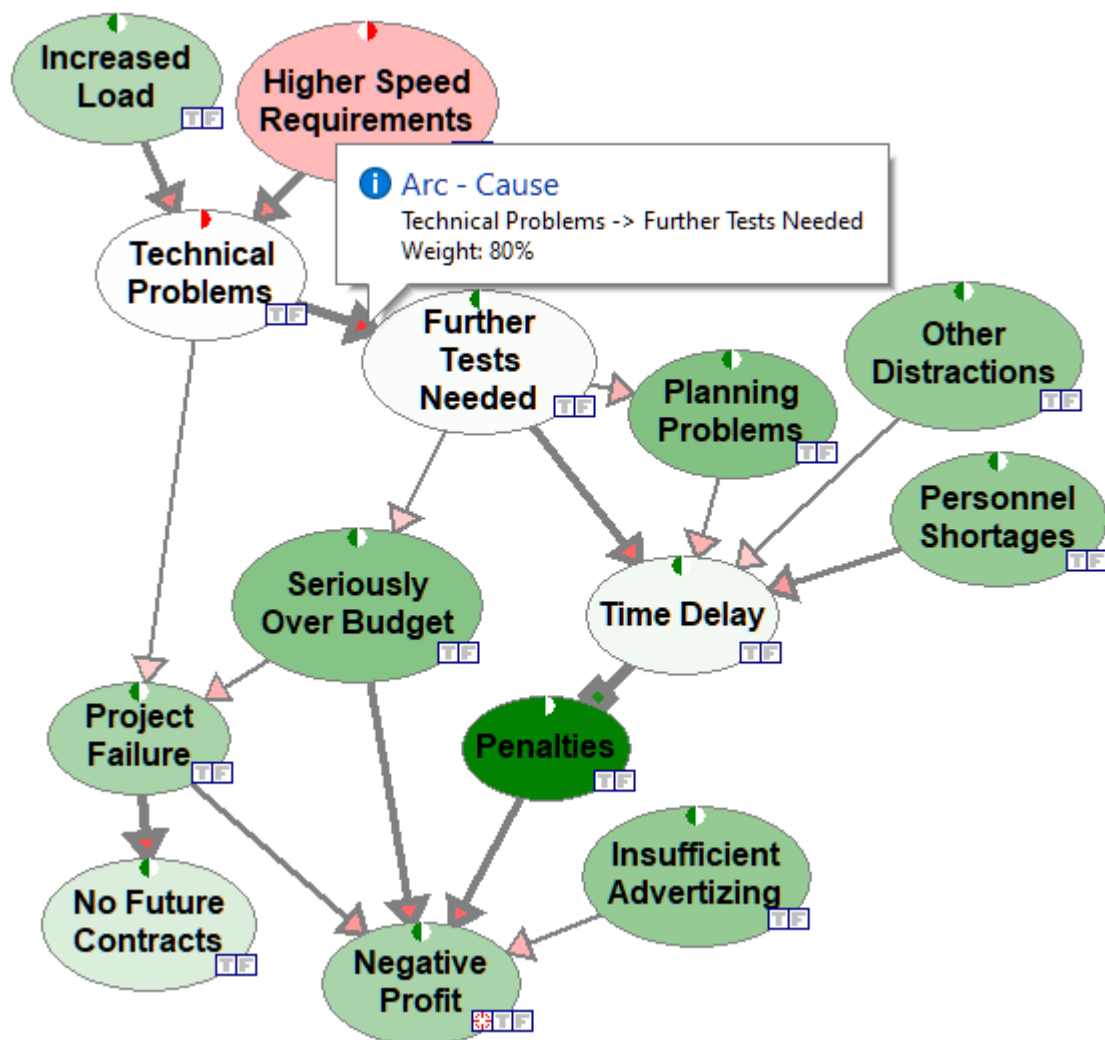
Strength of influence

Clicking on the *Enable variable arc widths* () tool from the [Standard Toolbar](#) changes the appearance of the arcs in the network. The arcs have different thickness, dependent on the strength of influence between the nodes that they connect. Strength of influence is based on the arc weight parameters, entered during the construction phase of the model.

Here is a fragment of the *Project* network with the *Enable variable arc widths* tool pressed.



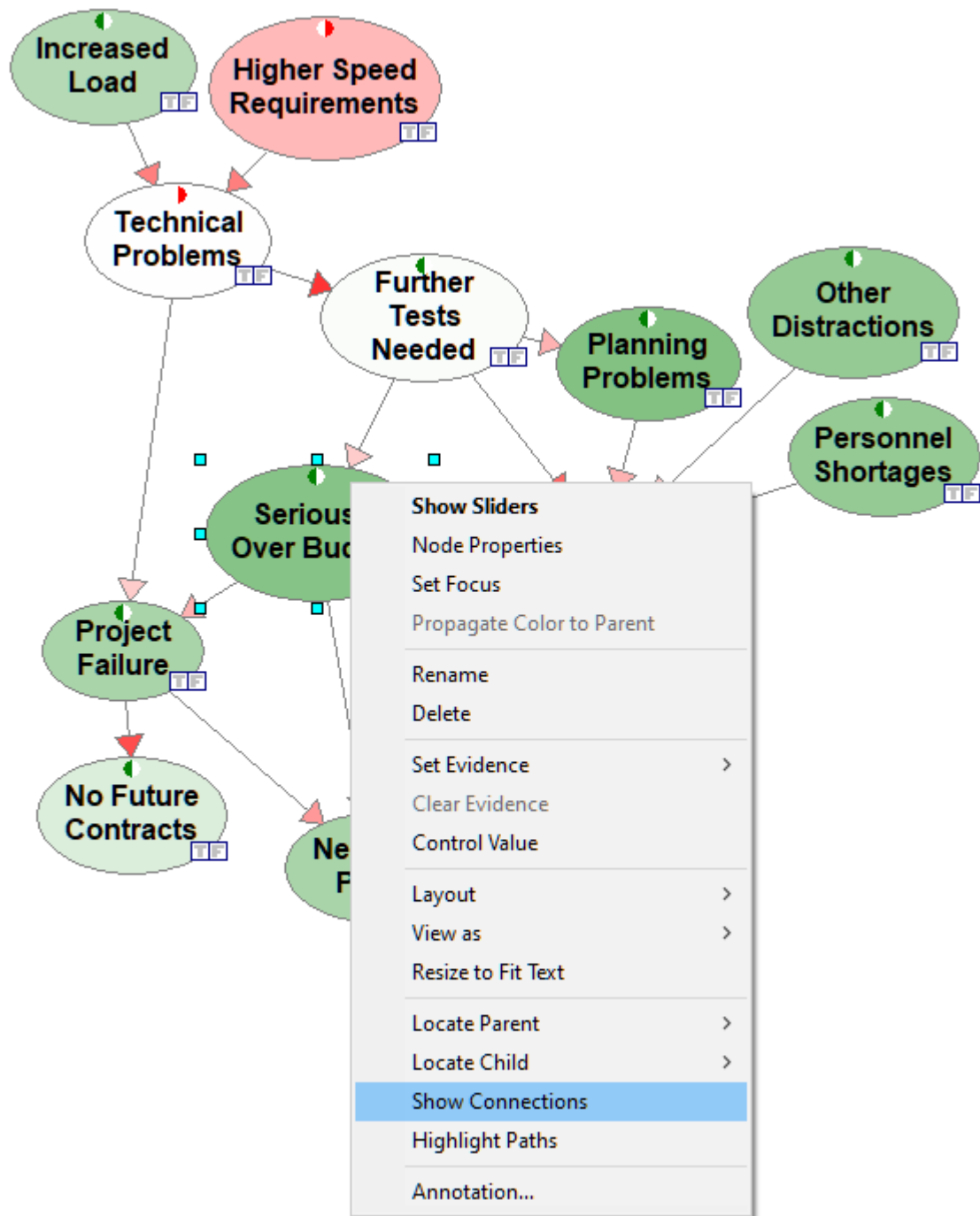
If the mouse is placed on the head of the arrow, information relating the strength of influence is shown in a comment box. The image below shows the strength of influence of the arc between *Technical Problems* and *Further Tests Needed*.



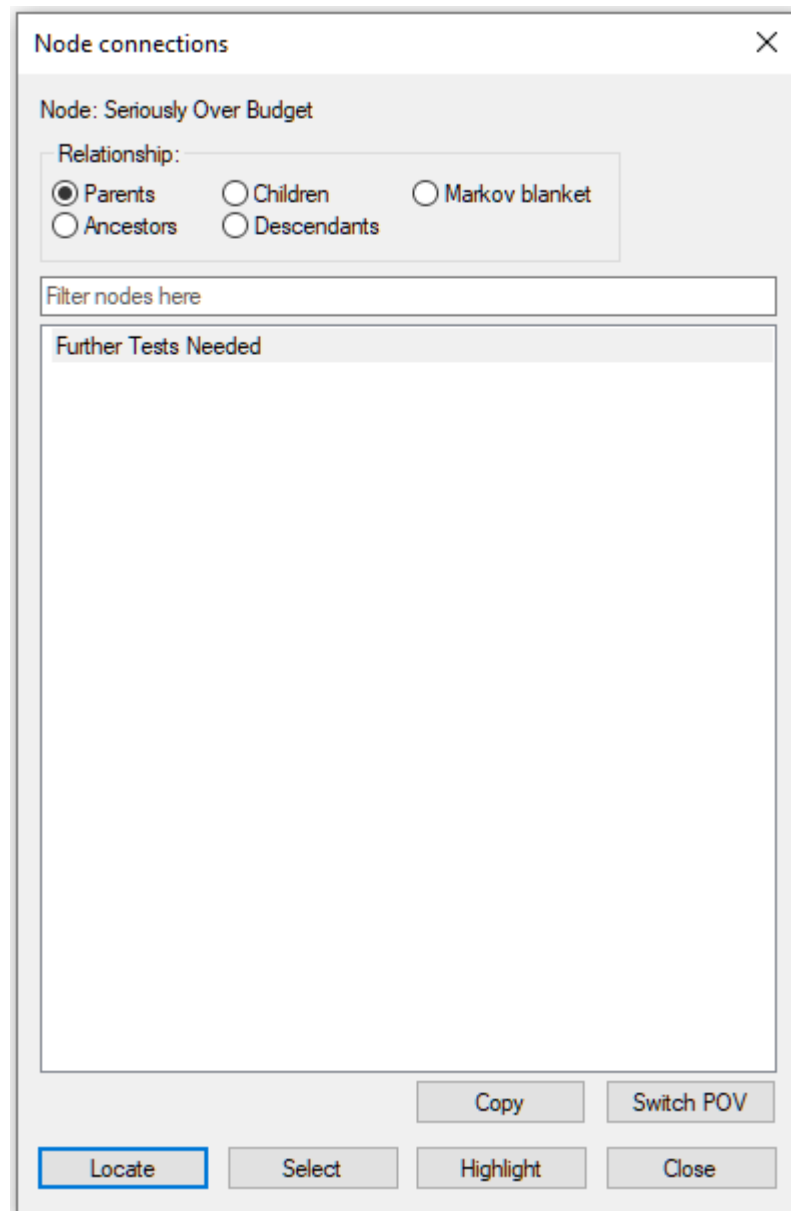
Variable arc widths allows for structural analysis of the model, as they show the strength of influences and, by this, the most critical paths of information flow. They allow for a visual verification of the intuitions that are captured in the model. The above screen shot of the Project model shows, for example, that the most critical pathway influencing profits is through customer demands for increased load and higher speed, which may result in technical problems, need for further tests, budgetary problems, time delay, and penalties, resulting in decreased profits.

Neighborhoods

There are several useful functions that help with analysis of connections and pathways through the qualitative graph. One group of such functions is showing connections of a selected node. To open the connections dialog, please select *Show Connections* from the context menu of the node in question. The image below shows invoking the dialog for the node *Seriously Over Budget*.



The dialog allows for selecting parents (direct predecessors in the graph), children (direct successors in the graph), ancestors (all predecessors in the graph), descendants (all successors in the graph), and the node's Markov blanket (the set of nodes that make the node in question independent of the other nodes in the graph). The only parent of the node *Seriously Over Budget* in the graph is *Further Tests Needed*.



Further Tests Needed is also a part of the ancestors of the node *Seriously Over Budget*.

Node connections

Node: Seriously Over Budget

Relationship:

☐ Parents ☐ Children ☐ Markov blanket

☒ Ancestors ☐ Descendants

Filter nodes here

Further Tests Needed

Higher Speed Requirements

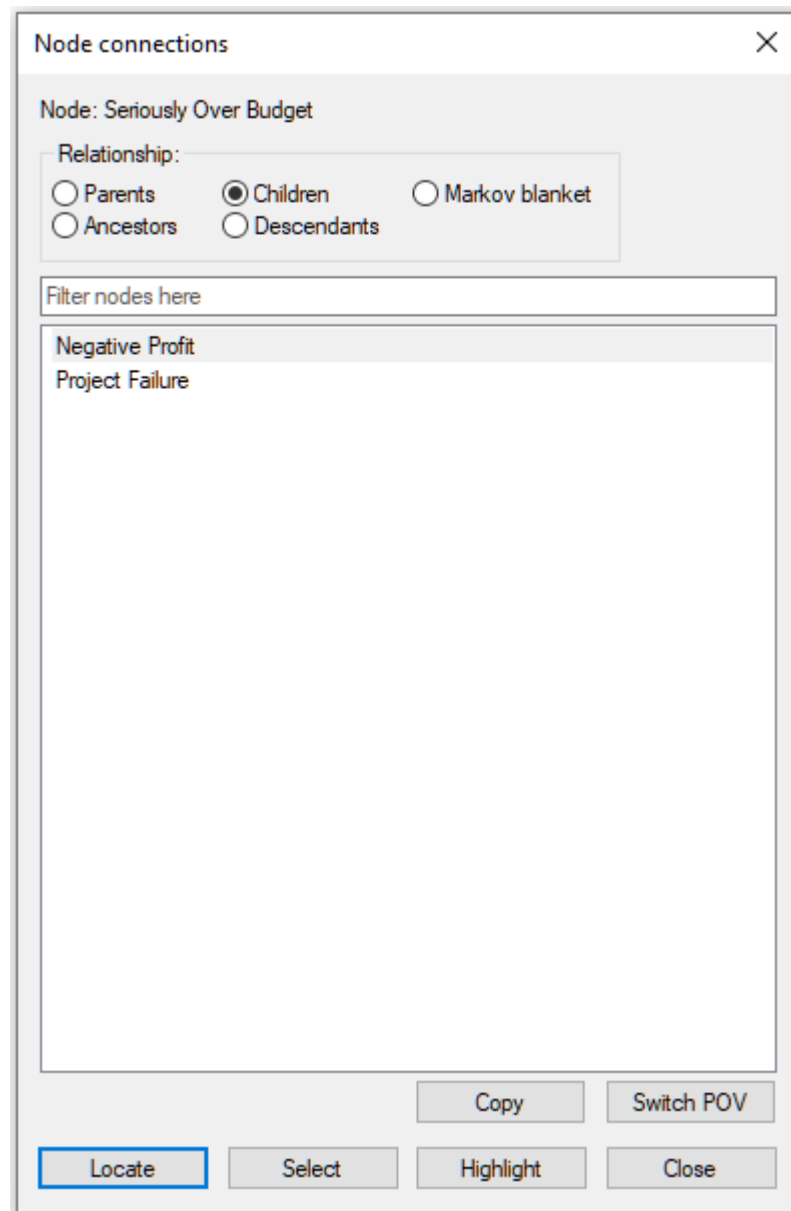
Increased Load

Technical Problems

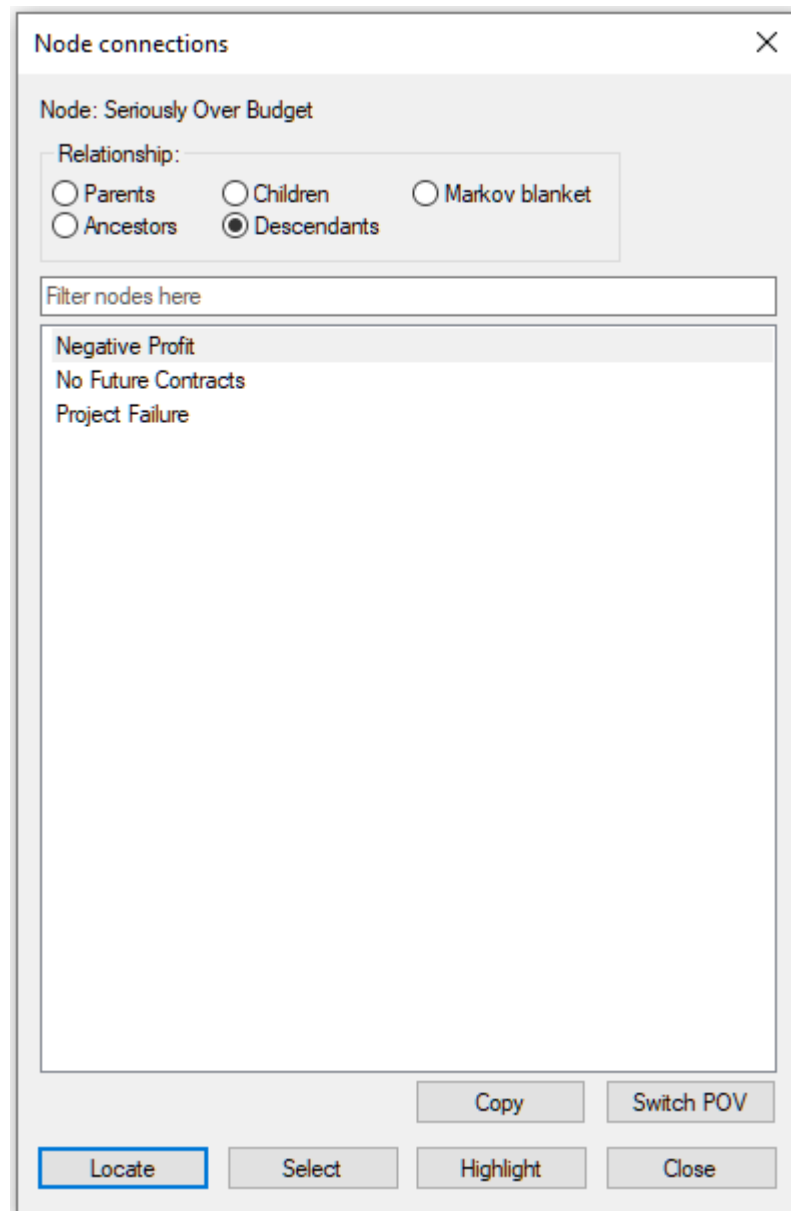
Copy Switch POV

Locate Select Highlight Close

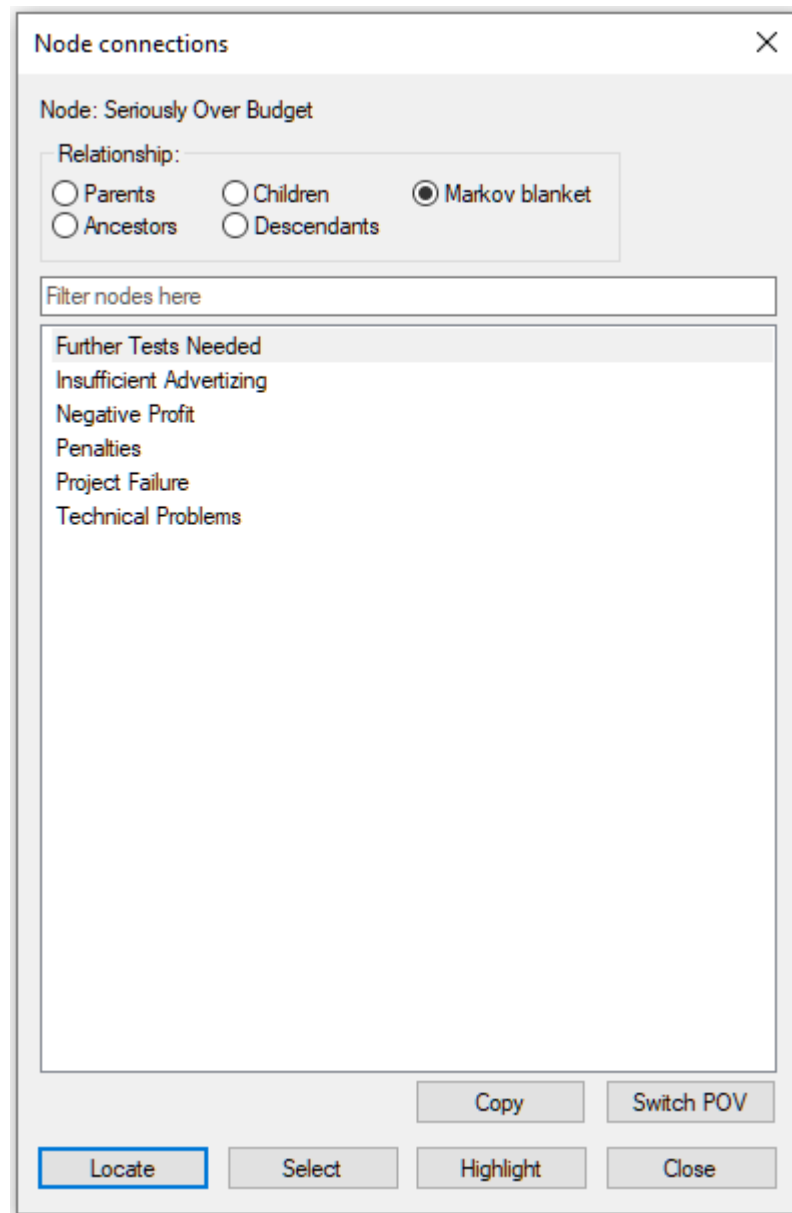
Seriously Over Budget has two children: *Negative Profit* and *Project Failure*.



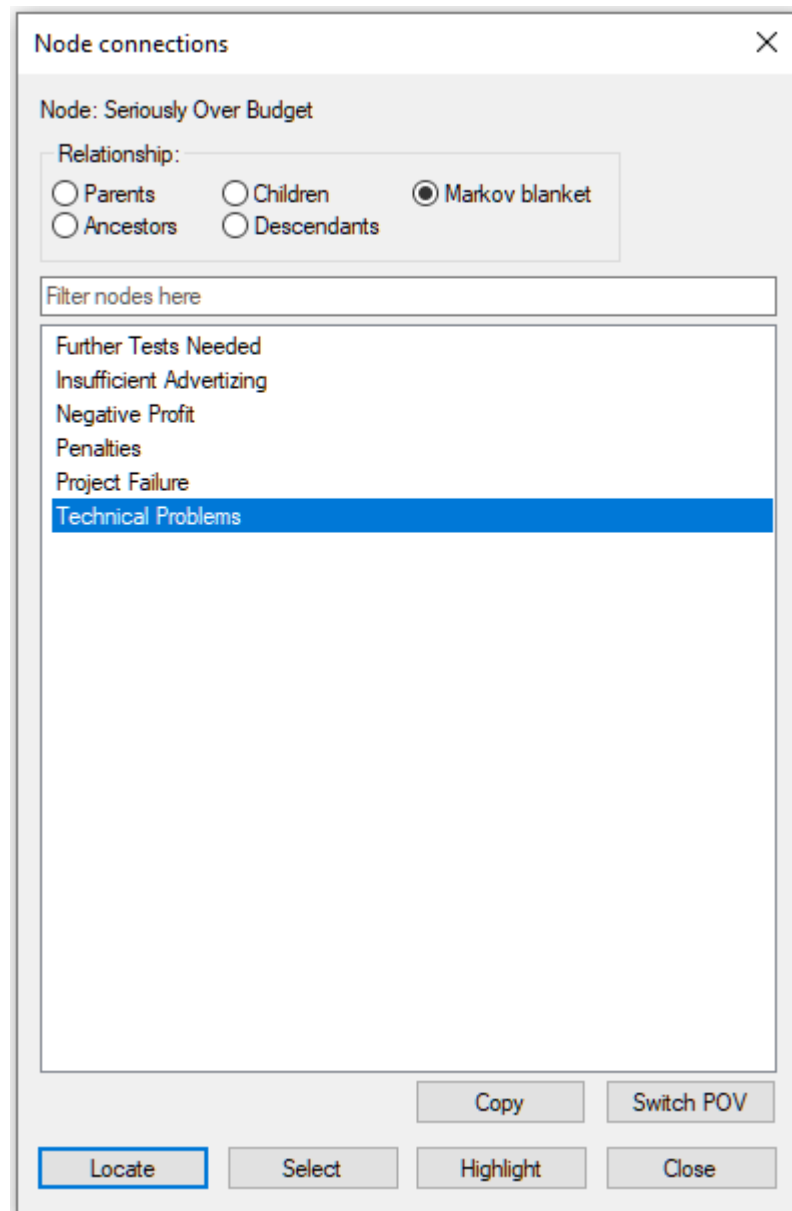
They both belong to the descendants of *Seriously Over Budget*.



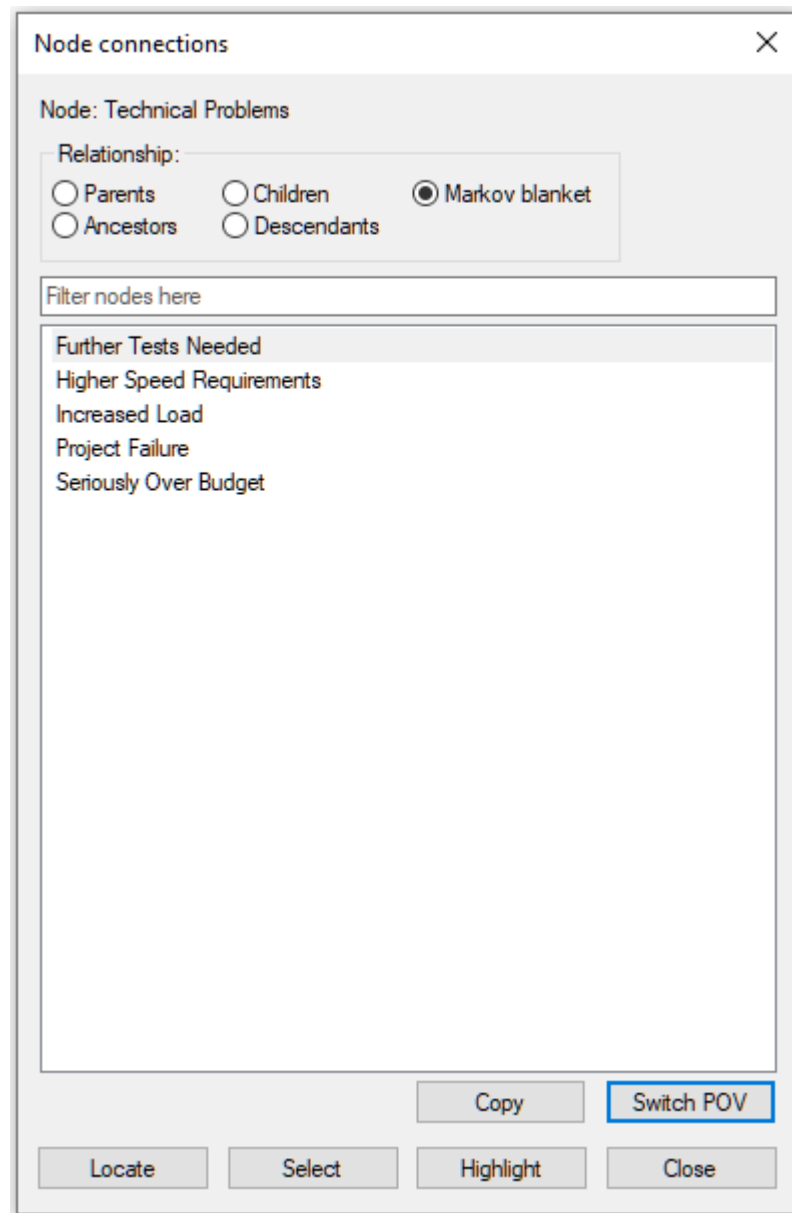
Parents, children, and the parents of those children belong to the Markov blanket of the node *Seriously Over Budget*.



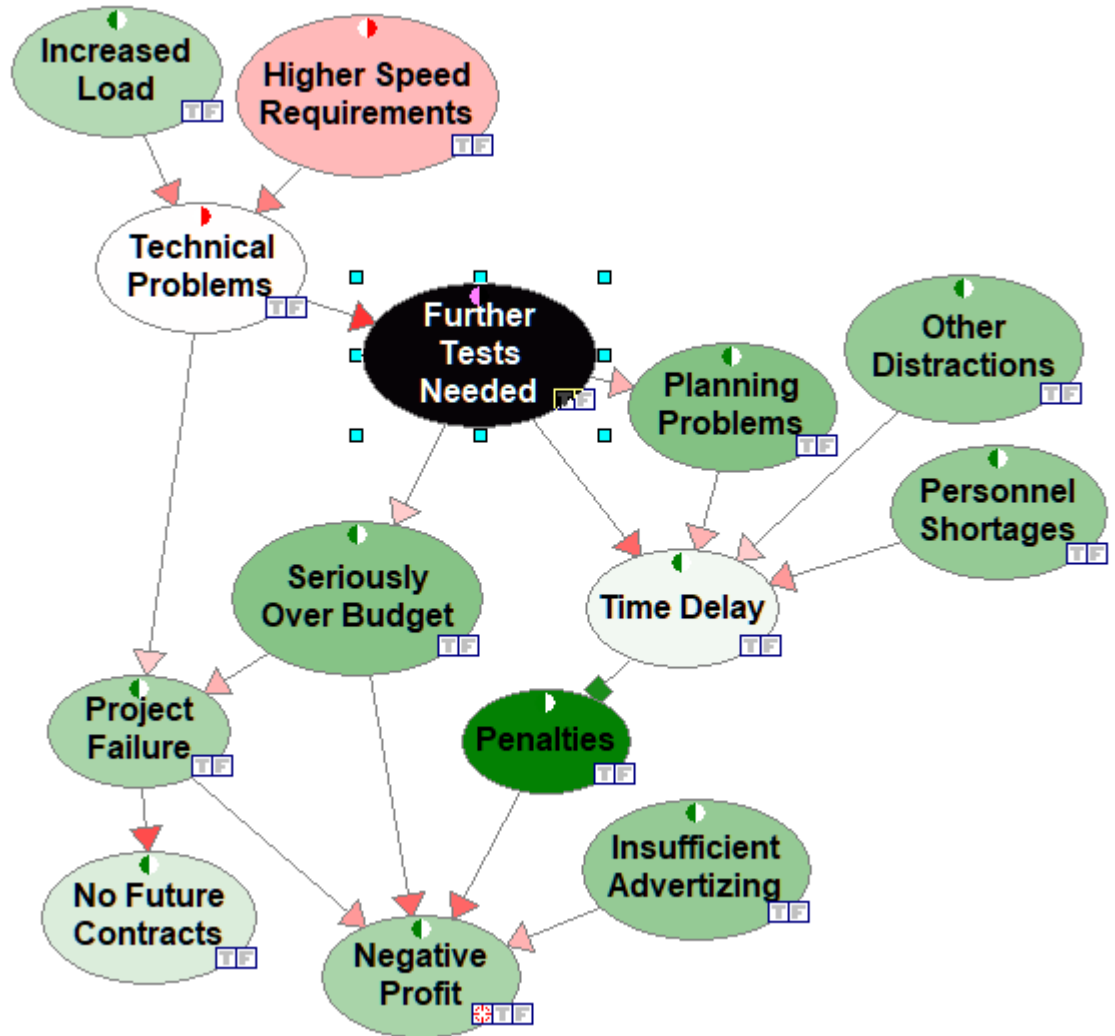
The dialog allows for traversing the graph through changing the focus of the analysis. A new focus can be chosen by selecting it from the list of nodes and pressing the *Switch POV (Point of View)* button in the lower-right corner of the dialog. Let us select *Technical Problems* and press the *Switch POV* button.



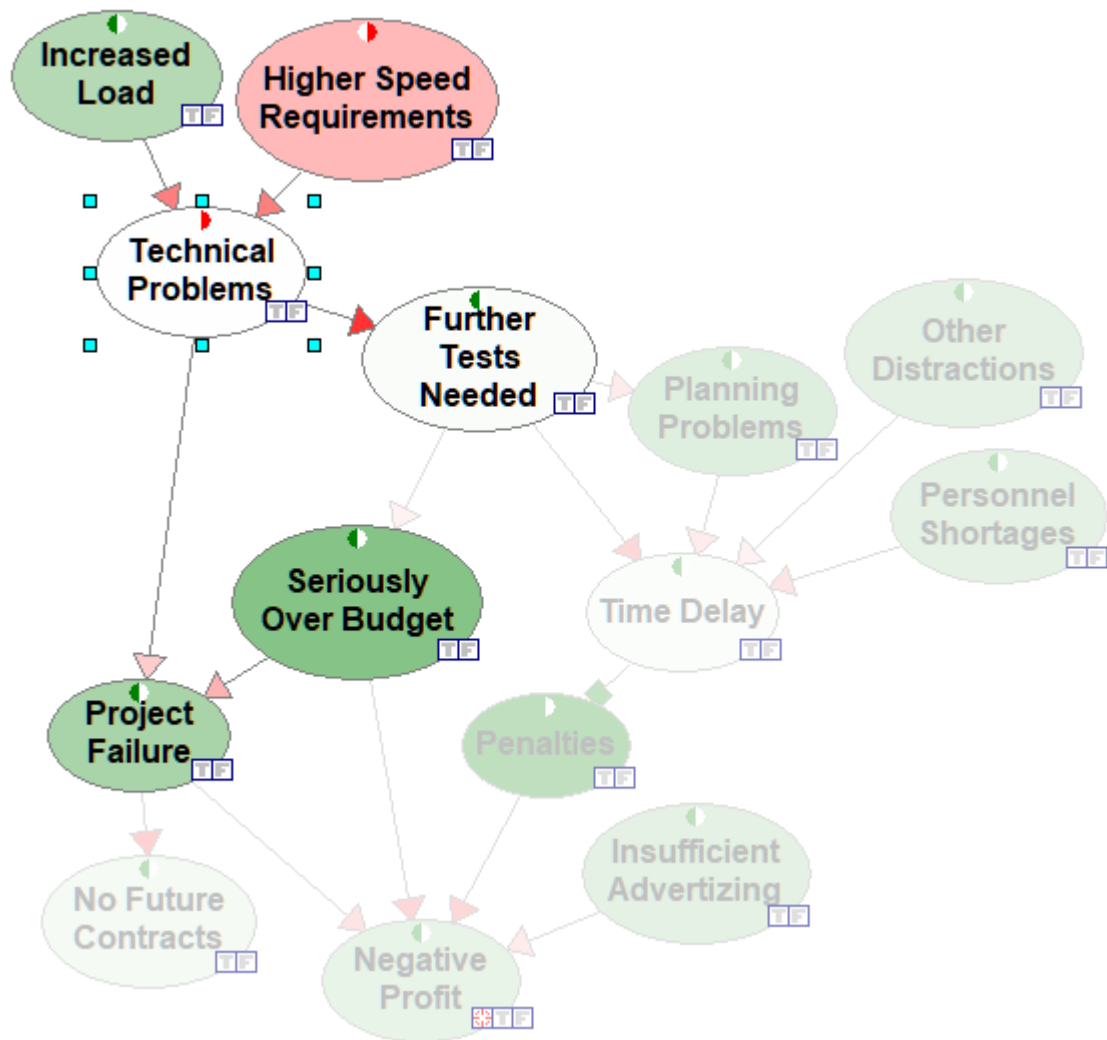
Pressing the *Switch* button leads to refocusing the neighborhood to the node *Technical Problems*.



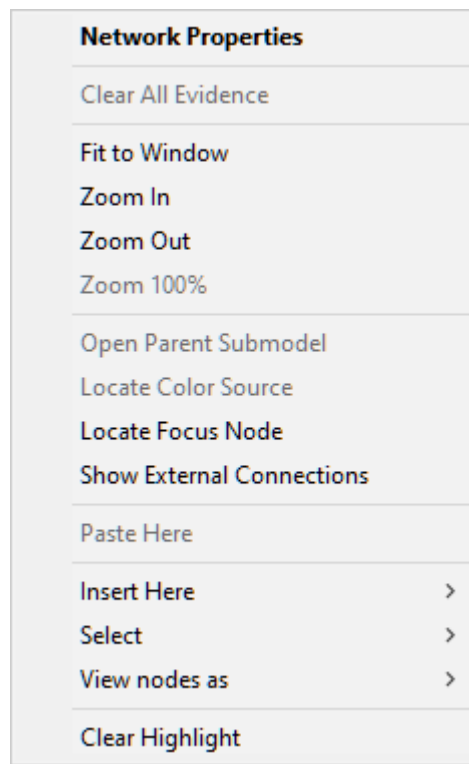
Selecting any of the nodes on the list and pressing *Locate node* button locates the node in the *Graph View* and flashes the node three times. Double-clicking on the selected node has the same effect. Selecting *Further Tests Needed* and pressing the *Locate node* button yields:



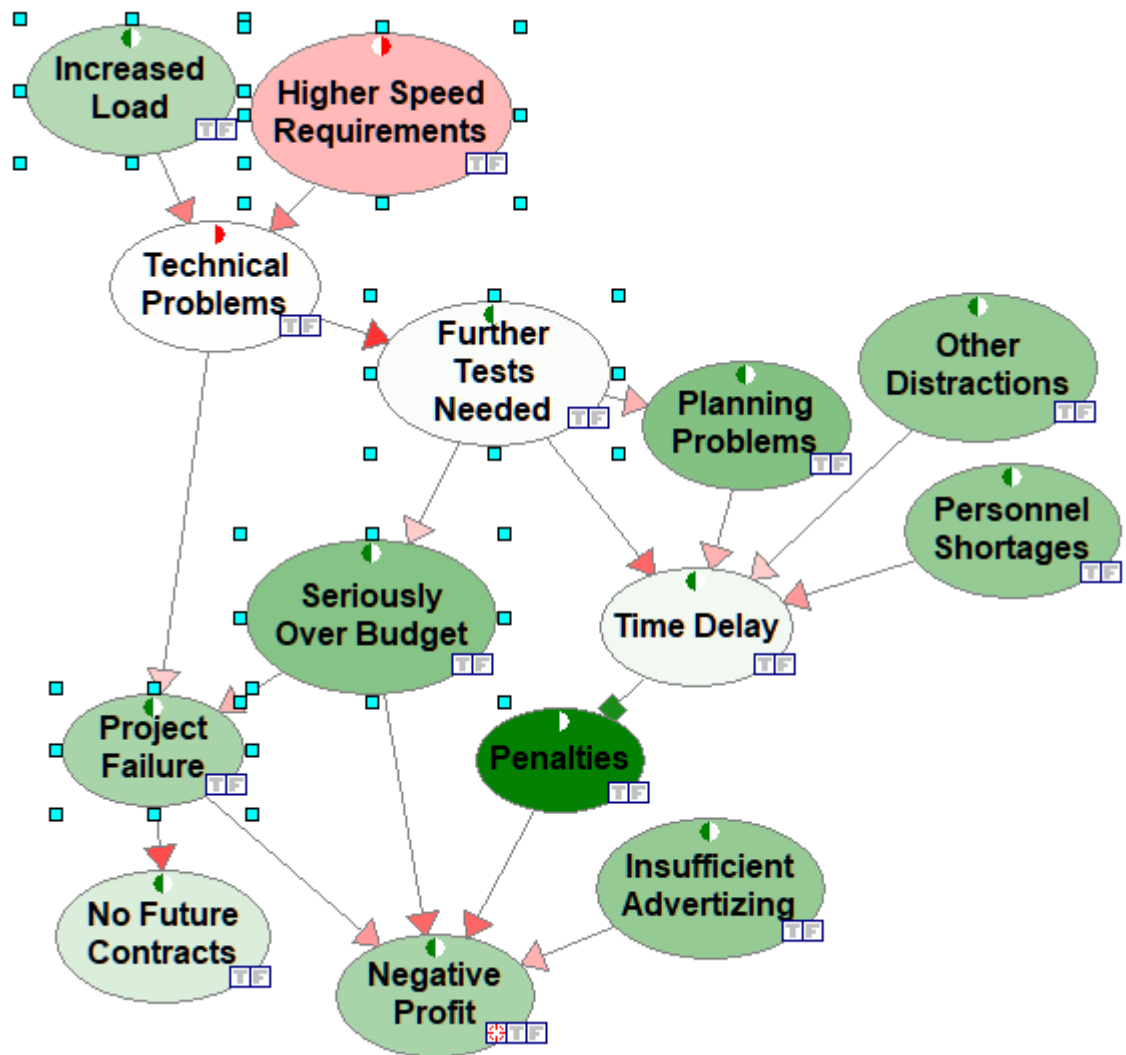
Pressing the *Highlight all* button exits the dialog and highlights all nodes on the list. In the image below, *Highlight all* button was pressed when the *Node connections* dialog showed the Markov blanket of the node *Technical Problems*.



Pressing ESC or choosing *Clear Highlight* from the network pop-up menu will clear the selection.



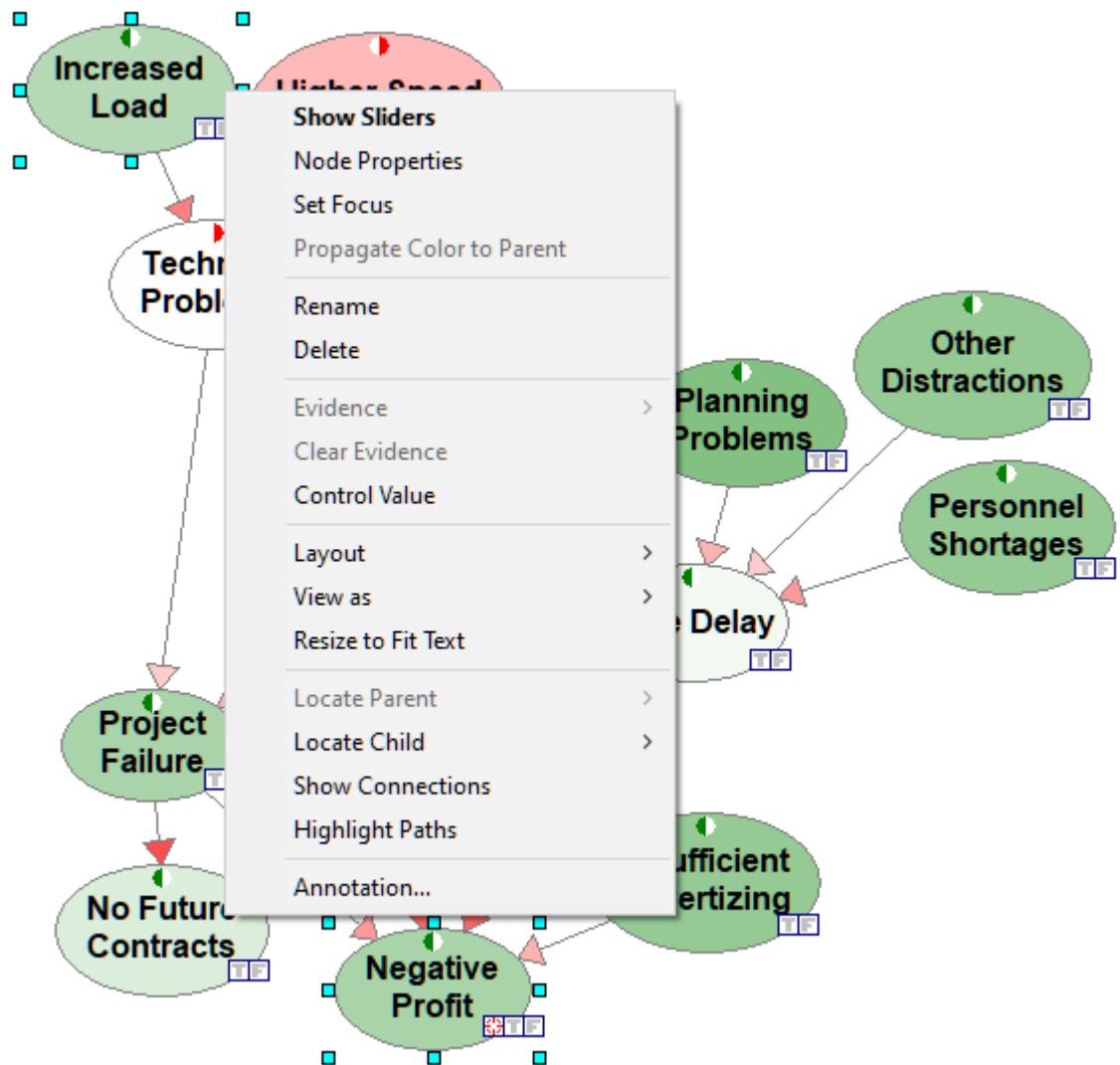
Pressing *Select* button in the *Show Connections* dialog selects all nodes on the list (selection does not include the POV node!) in the [Graph View](#). In the image below, *Select* button was pressed when the *Node connections* dialog showed the *Markov blanket* of the node *Technical Problems*.



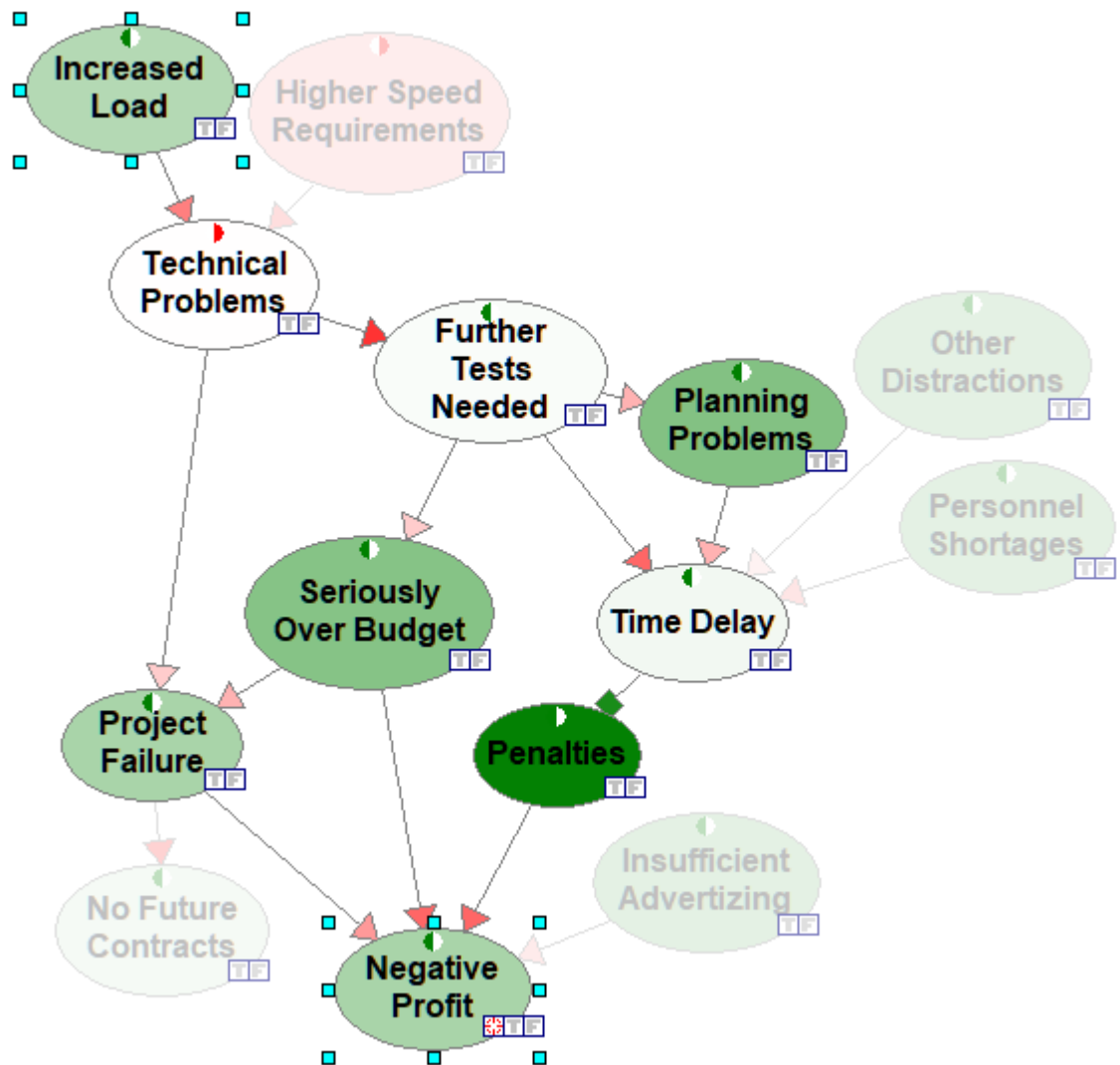
This selection can be further enhanced through other ways of selecting nodes and arcs. Please see the *Select Nodes...* dialog, described in the [Selection of model elements](#) section.

Pathways of information flow

QGeNIe allows for showing active paths through which information flows between a pair of nodes. To show all paths between two nodes A and B, please select these two nodes and choose *Highlight Paths* from the node context menu of one of the two nodes. The following image shows how to invoke highlighting paths between the nodes *Increased Load* and *Negative Profit*.



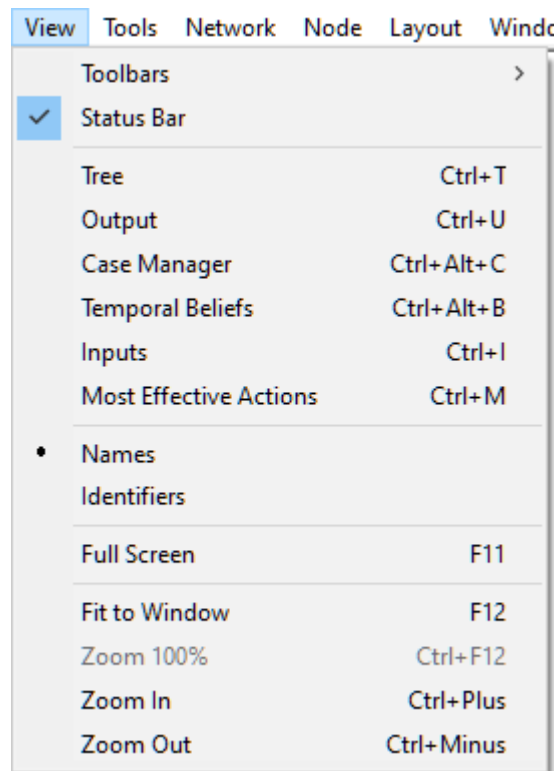
The result of choosing *Highlight Paths* above is



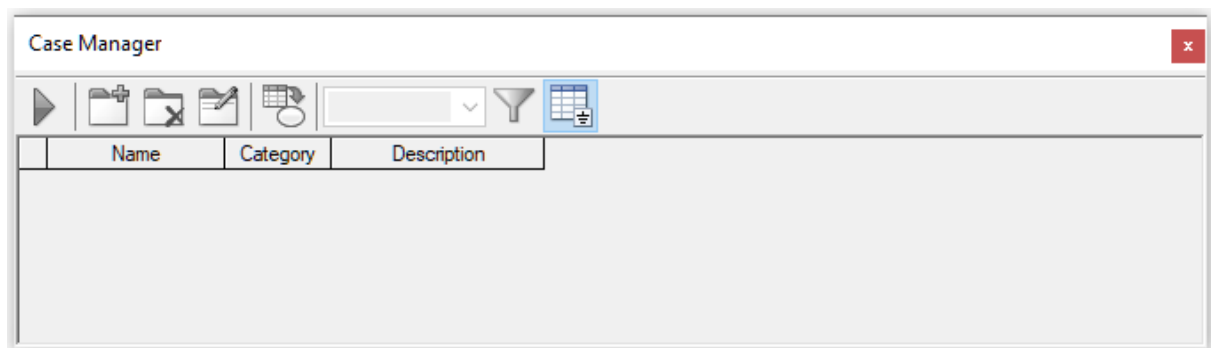
Pressing ESC or choosing *Clear Highlight* from the network pop-up menu will clear the selection.

6.3.7 Case Manager

QGeNIe includes a *Case Manager* window that allows users to save a partial or a complete session as a case and retrieve this case at a later time. Cases are saved alongside the model, so when the model is loaded at a later time, all cases are going to be available. *Case Manager* window can be opened through the *View Menu*:

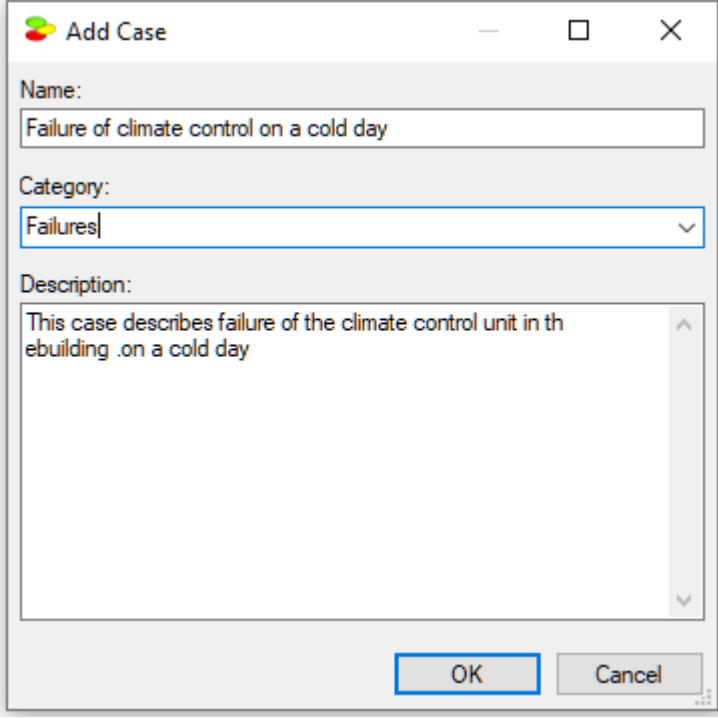


Case Manager window looks initially as follows:



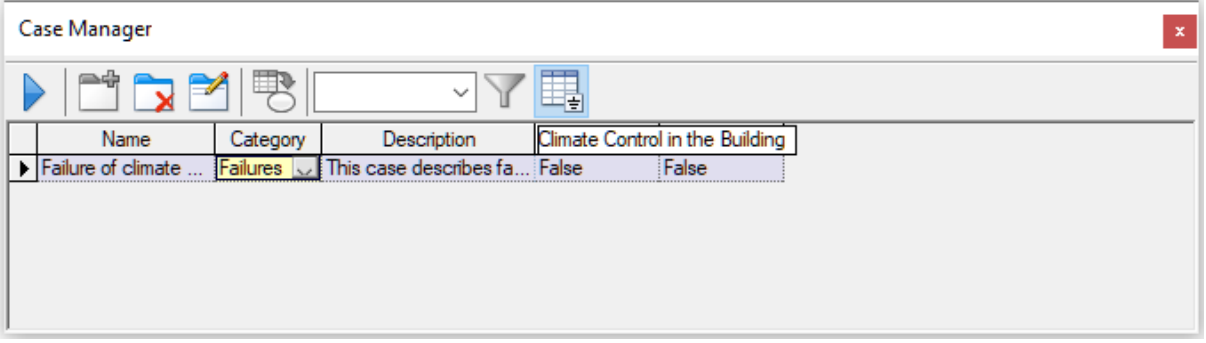
Adding cases to the Case Manager

Let us add to the Case Manager the three cases analyzed in the [Building a qualitative Bayesian network](#) section of this manual. The first case involved failure of climate control on a cold day. Once the evidence in the model has been set, we click on the *Add new case* button (📄). This results in the following dialog, which allows for entering case details.



The "Add Case" dialog box is shown. It has a title bar with a QGeNle logo and standard window controls. The "Name:" field contains "Failure of climate control on a cold day". The "Category:" dropdown menu is set to "Failures". The "Description:" text area contains "This case describes failure of the climate control unit in the building on a cold day". At the bottom are "OK" and "Cancel" buttons.

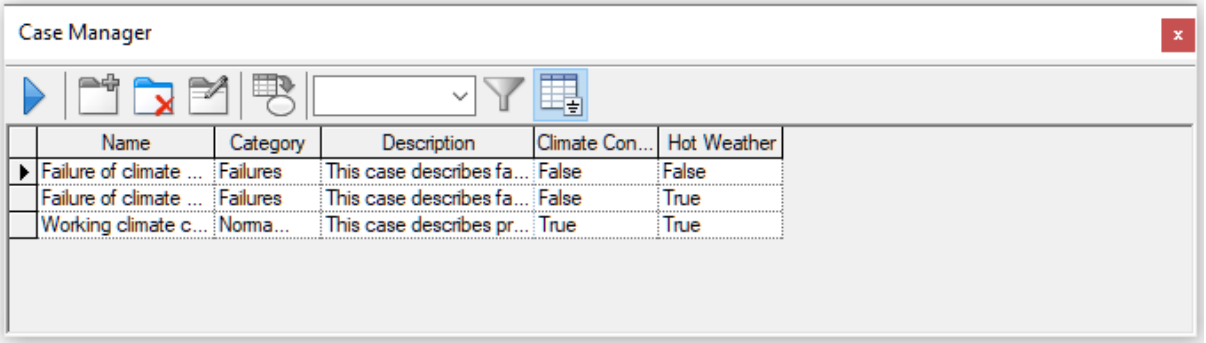
Once we click OK, the case is visible in the *Case Manager*:



The "Case Manager" window displays a table of cases. The table has columns: Name, Category, Description, Climate Control in the Building, and Hot Weather. The first case is "Failure of climate control on a cold day" under the "Failures" category.

Name	Category	Description	Climate Control in the Building	Hot Weather
Failure of climate control on a cold day	Failures	This case describes failure of the climate control unit in the building on a cold day	False	False





Adding two remaining cases to the *Case Manager* will result in the following window.



The "Case Manager" window now displays three cases in the table.

Name	Category	Description	Climate Control in the Building	Hot Weather
Failure of climate control on a cold day	Failures	This case describes failure of the climate control unit in the building on a cold day	False	False
Failure of climate control on a hot day	Failures	This case describes failure of the climate control unit in the building on a hot day	False	True
Working climate control on a cold day	Normal	This case describes proper operation of the climate control unit in the building on a cold day	True	True

The *Case Manager* window shows the currently applied case with a grayed background.

The *Show only evidence nodes* button () reduces the number of columns displayed to those only that have any observations at all. The *Apply case* () button allows for transferring a case to the *Graph View* window. The *Delete case* button () removes the current case from the *Case Manager*. The *Update applied case with network evidence* () button transfers the current set of evidence from the current model to the current case.

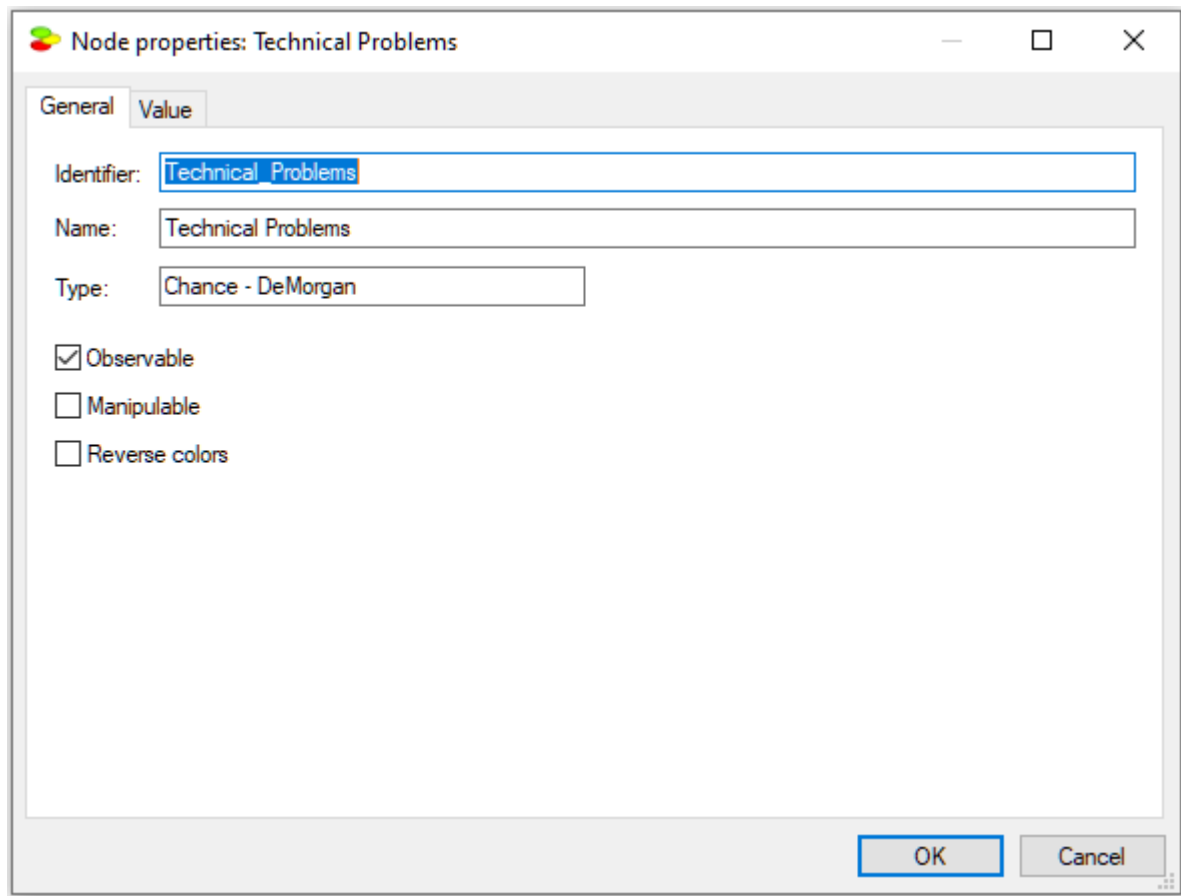
Exporting case records to a data file

Currently, QGeNIe does not allow to export cases from a model to a data file. However, it is possible to retrieve cases using any text editor from the model file (.qdsl). To locate the case descriptions, please search for the tag <cases> in the XML model file.

6.4 Most Effective Actions calculation

6.4.1 Introduction

It is often of interest to a user which variables can be best manipulated in order to affect a specified node most. To this effect, QGeNIe allows for specifying a *Focus* node, denoted by a small target icon on the graph. Given a focus node, QGeNIe supports calculating the value of information flowing from observations and the value of intervention (also known as manipulation of control). For that calculation to be performed, the model builder should specify for every node in the model whether it is possible to observe it and whether it is possible to manipulate it. *Observable* and *Manipulable* properties are part of the *General* property sheet.

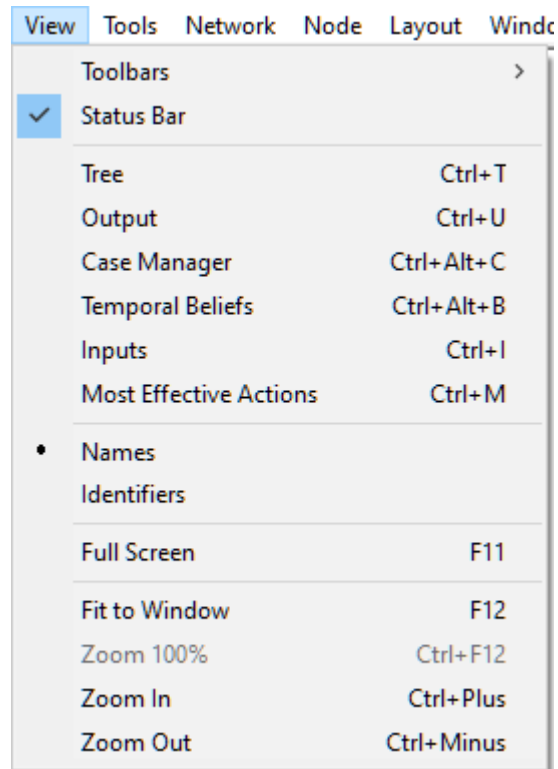



The *Technical Problems* node pictured above is *Observable* but not *Manipulable*.

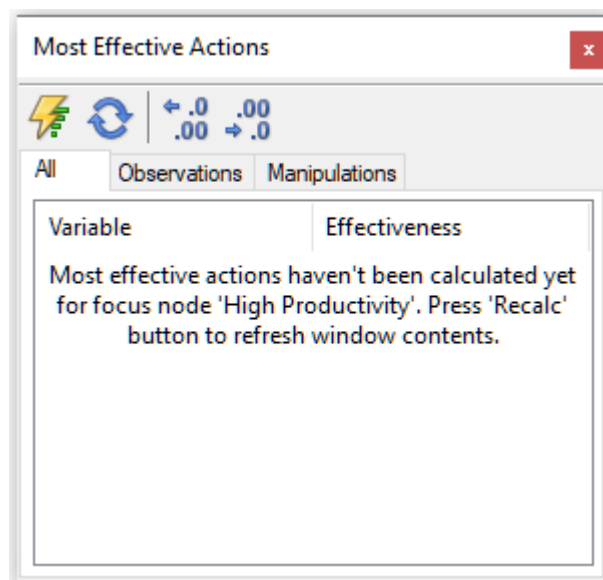
The following two sections show how to enable value of information and value of intervention calculation and how to perform and interpret its results.

6.4.2 Enabling Most Effective Actions calculation

To enable the most effective actions dialog, choose *Most Effective Actions* from the *View* menu

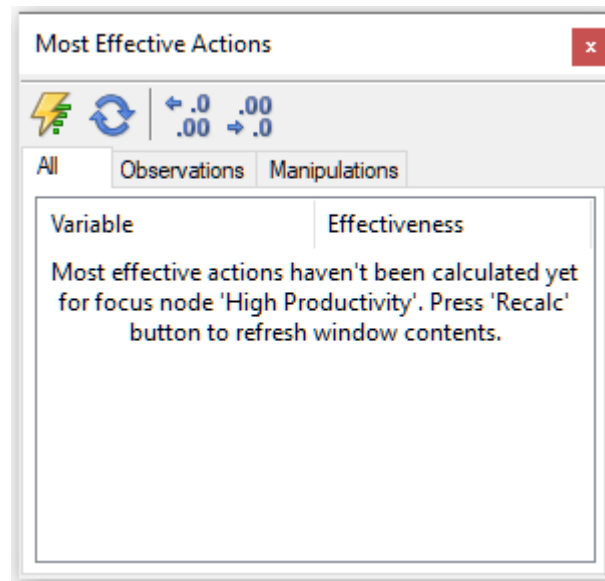




Alternatively, use the shortcut CTRL-M or press the *Toggle Most Effective Actions window* () tool, which will all open the *Most Effective Actions* window

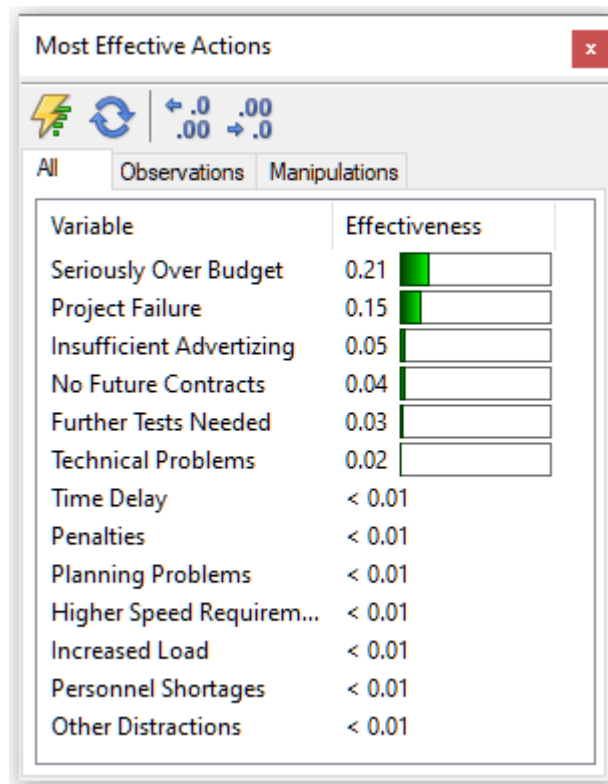


6.4.3 Most Effective Actions calculation

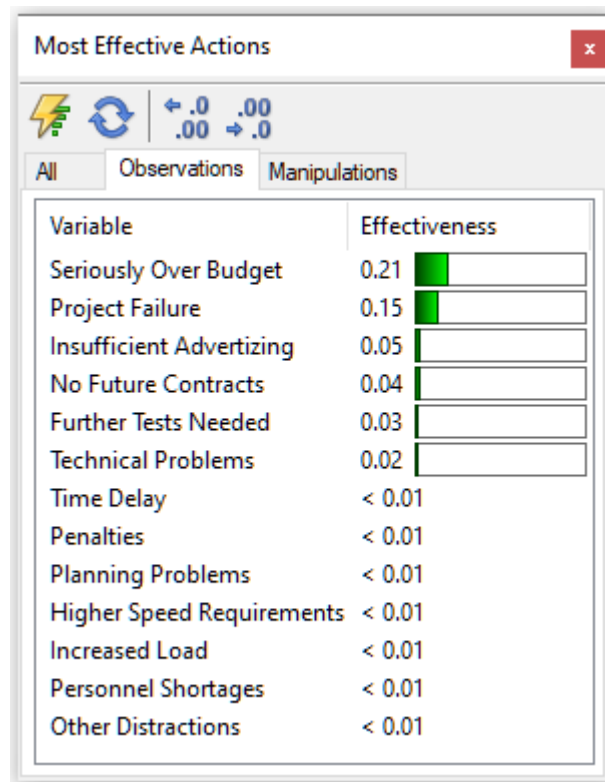
Specifying for every node whether it is observable and manipulable and enabling the *Most Effective Actions* calculation will all open the *Most Effective Actions* window



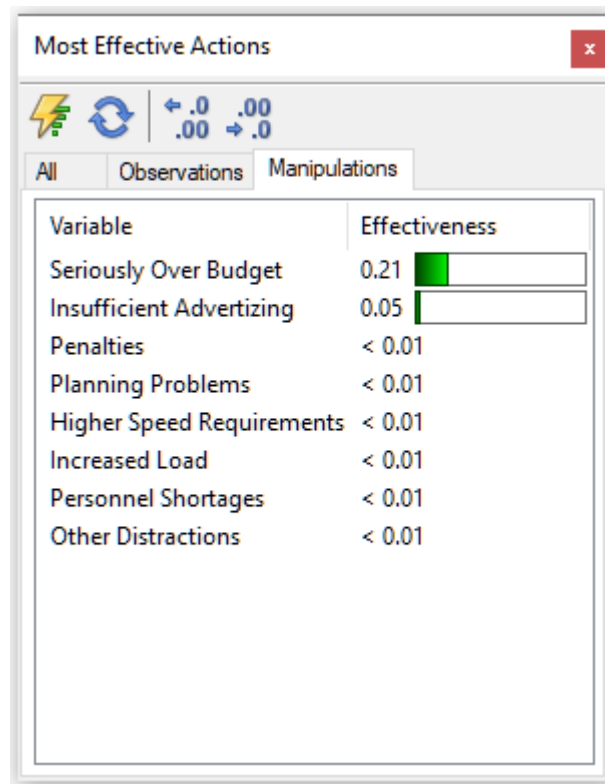
Pressing the *Recalc* () button or, better yet, pressing on the *Autoupdate* () button, will calculate the value of observations and manipulations of those variables in the model that can be observed or manipulated.



We can look only at those variables that are observable



or manipulable



It is important to remember that the value of information/manipulation does not have units. It is based on calculation of cross-entropy and just expresses the expected change in entropy of the model *Focus* variable. The most valuable in this calculation is the relative magnitude of the calculated values and their order.

6.5 Dynamic Bayesian networks

6.5.1 Introduction

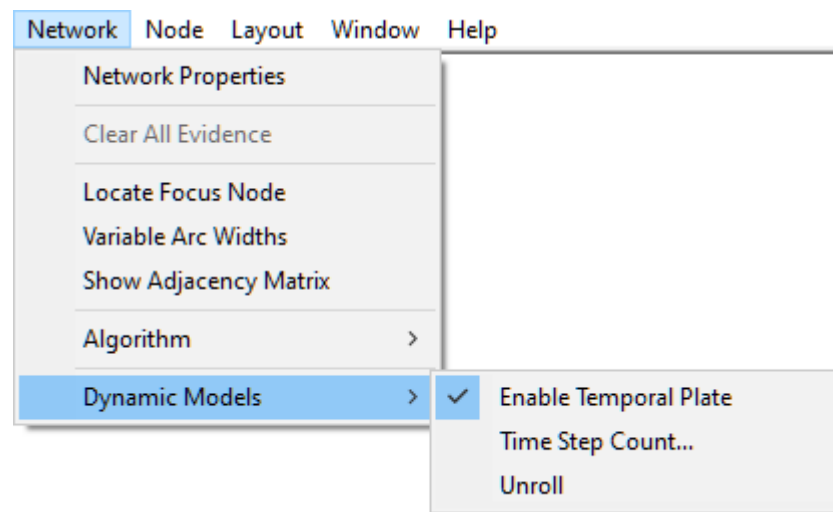
A Bayesian network is a snap shot of the system at a given time and is used to model systems that are in some kind of equilibrium state. Unfortunately, most systems in the world change over time and sometimes we are interested in how these systems evolve over time more than we are interested in their equilibrium states. Whenever the focus of our reasoning is change of a system over time, we need a tool that is capable of modeling dynamic systems.

A *dynamic Bayesian network (DBN)* is a Bayesian network extended with additional mechanisms that are capable of modeling influences over time (Murphy, 2002). We assume that the user is familiar with DBNs, Bayesian networks, and QGeNIe. The temporal extension of BNs does not mean that the network structure or parameters changes dynamically, but that a dynamic system is modeled. In other words, the underlying process, modeled by a DBN, is stationary. A DBN is a model of a stochastic process. The implementation of DBNs in QGeNIe is based on (Hulst 2006).

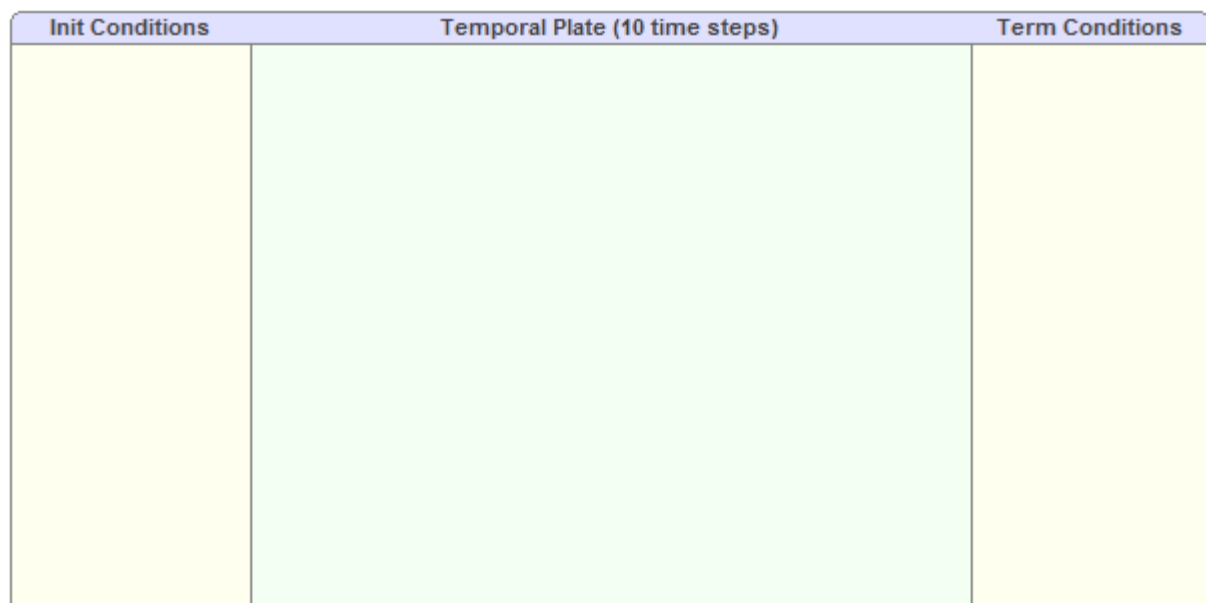
6.5.2 Creating DBN

We will use QGeNIe to create a qualitative model of a time-dependent domain focusing on a manufacturer's decision whether to produce a high quality product and whether this will have effect on profits and market share.

We start modeling with enabling the *Temporal Plate*, which is a special construct in the *Graph View* that allows for building dynamic models



The effect of enabling temporal plate in the *Graph View* is the following

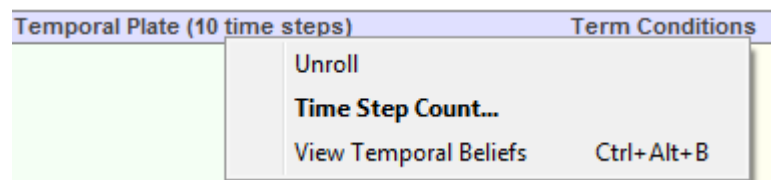


The *Temporal Plate* divides the *Graph View* into four areas:

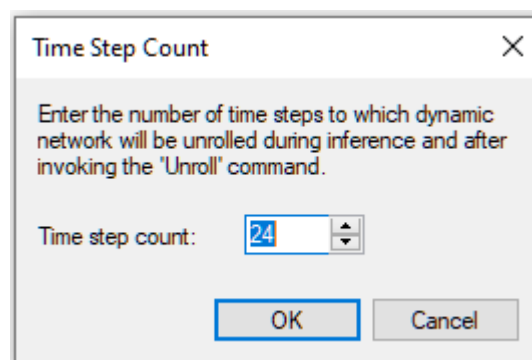
- *Contemporals*, which is the part of the *Graph View* window that is outside of the temporal plate. All nodes outside of the *Temporal Plate* are static.
- *Init Conditions*, which is the part of the network area where, so called, *anchor nodes* are stored. An *anchor node* is a node outside of the temporal plate that has one or more children inside the temporal plate. Anchor nodes are similar to static nodes outside of the temporal plate but they are only connected to their children in the first time-slice of the network.
- *Temporal Plate*, which is the main part representing the dynamic model. Nodes in the *Temporal Plate* are the only nodes that are allowed to have *Temporal Arcs*. This area also shows the number of time-slices for which inference will be performed.
- *Term Conditions*, which is the part of the network area where the *terminal nodes* are stored. A terminal node is a node outside of the temporal plate that has one or more parents inside the temporal plate. *Terminal nodes* are only connected to its parents in the last time-slice of the network.

The size of the *Temporal Plate* can be changed by clicking and dragging its edges and so can the sizes of its three areas (*Init Conditions*, *Temporal Plate*, and *Term Conditions*). There is a small subtlety in resizing the three. If you click and drag the extreme right or extreme left edge of the temporal plate, it is the middle part (the *Temporal Plate*) that gets resized and the sizes of *Init Conditions* and *Temporal Plate* remain the same. Pressing the *SHIFT* button when dragging the edges has the effect that the size of the *Temporal Plate* remains the same and the sizes of *Init Conditions* and *Temporal Plate* change.

For our example, we set the number of steps to 24. We can either double-click or right-click on the header of the *Temporal Plate*

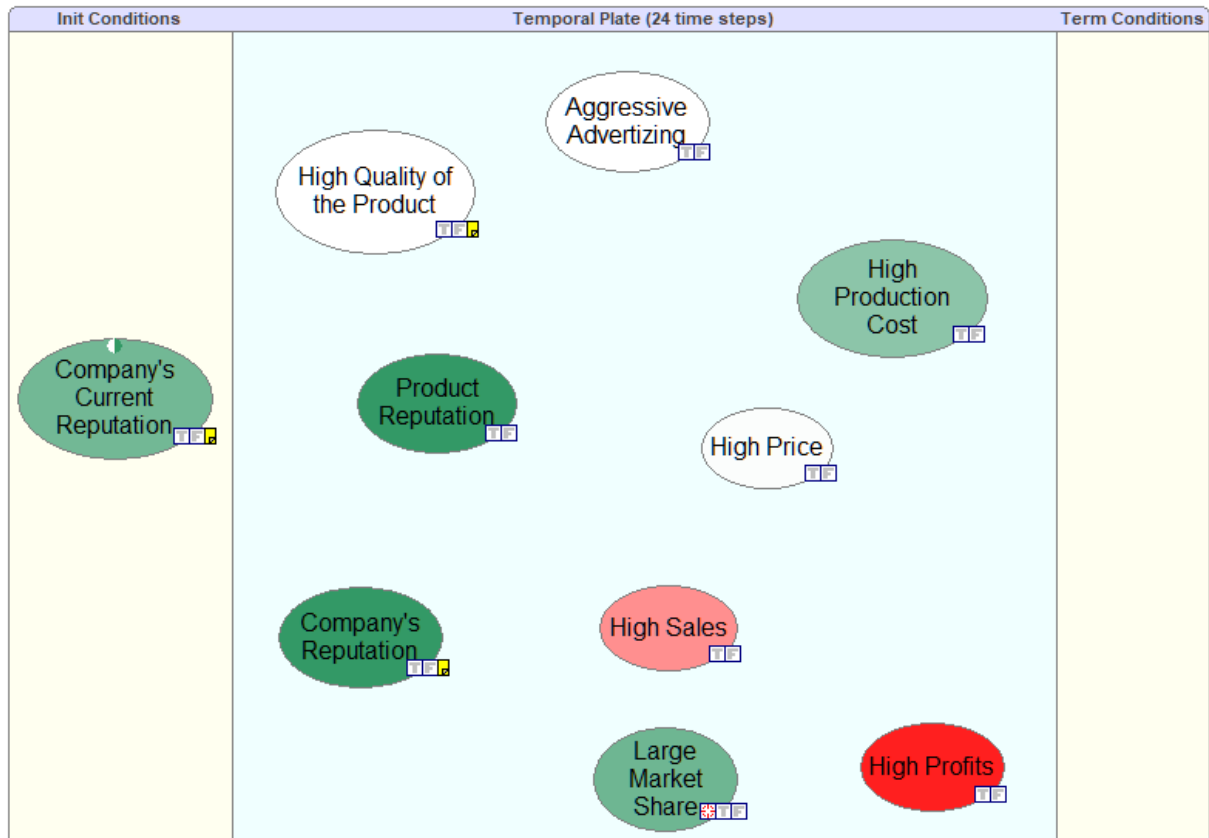


which will invoke the *Time Step Count* dialog that allows to change the *Time step count*




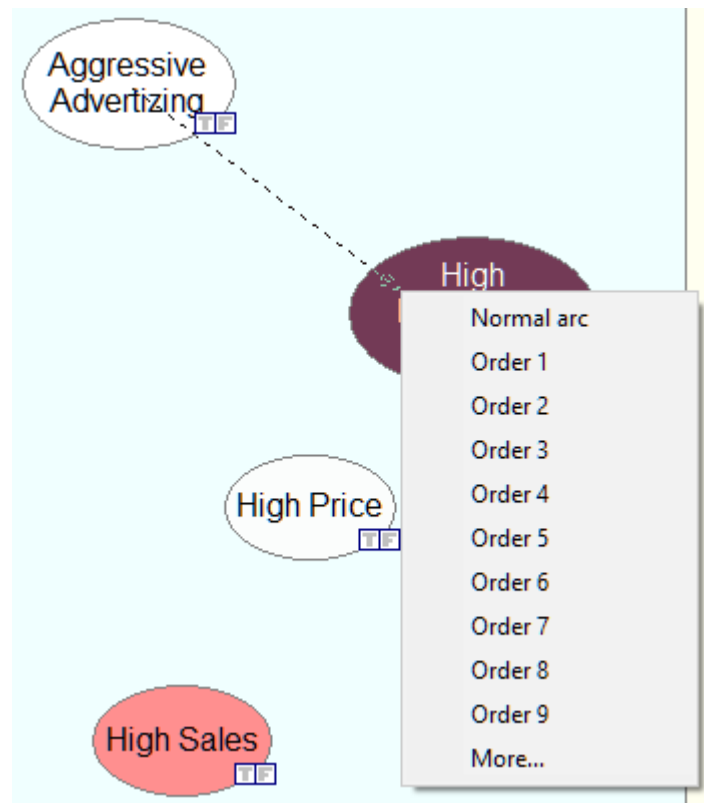
Time steps have no units and are a matter of interpretation of the modeler. In our model, we can interpret the time step as one month but in any other model it could mean a year, a day, or a millisecond.

We create the following nodes: *High Quality of the Product*, *Product Reputation*, *High Production Cost*, *Aggressive Advertizing*, *High Price*, *High Sales*, *High Profits*, *Large Market Share*, and *Company's Reputation* in the *Temporal Plate* and the *Company's Current Reputation* node in the *Init Conditions* area.

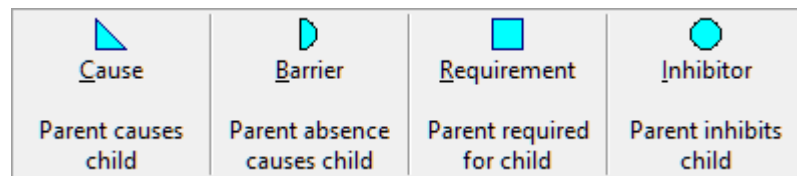


The next step is to connect these nodes. There are two types of arc between nodes: normal arcs and temporal arcs. Let us first create an arc between *Aggressive Advertizing* and *High Production Costs*.

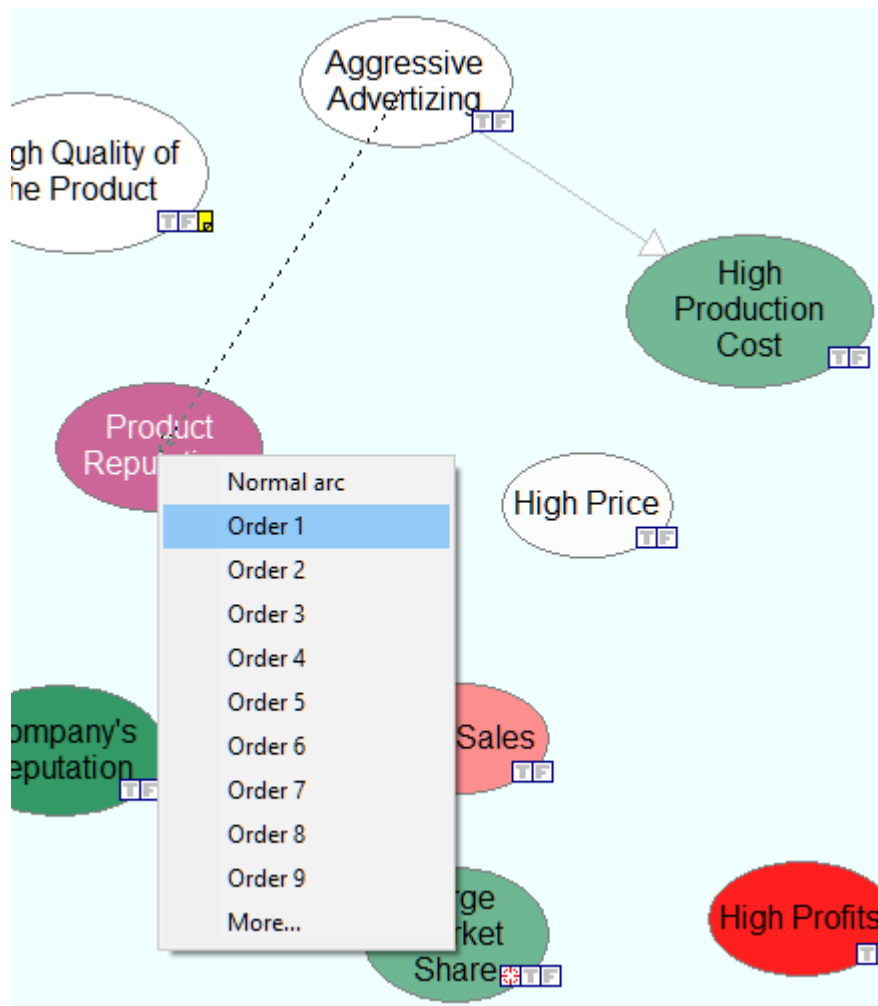
We click on the *Arc* button (), subsequently click on the node *Aggressive Advertizing* and drag and release the mouse inside the node *High Production Costs*. A menu pops up.



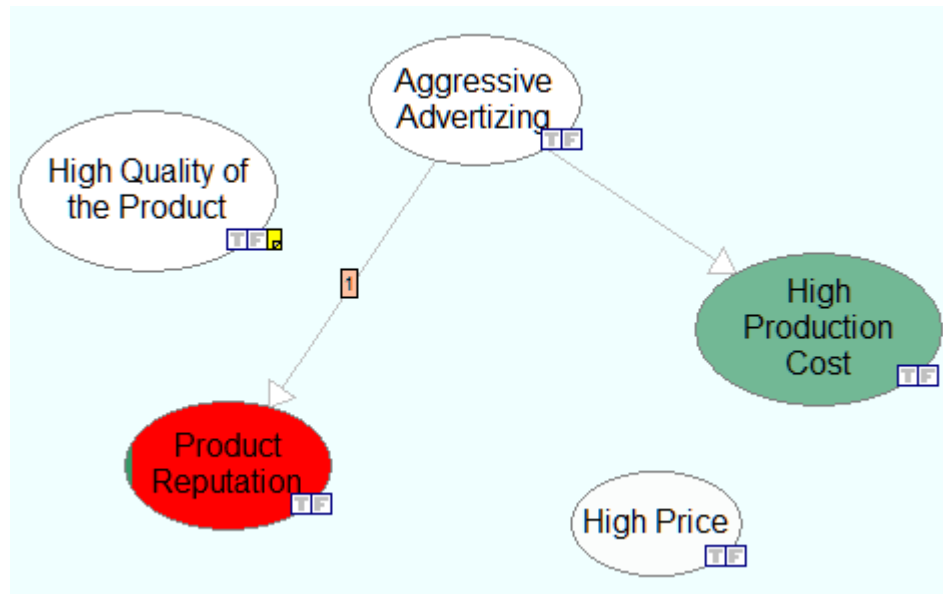
We select *Normal arc*, which creates an arc between *Aggressive Advertizing* and *High Production Costs*. and show the familiar arc type dialog



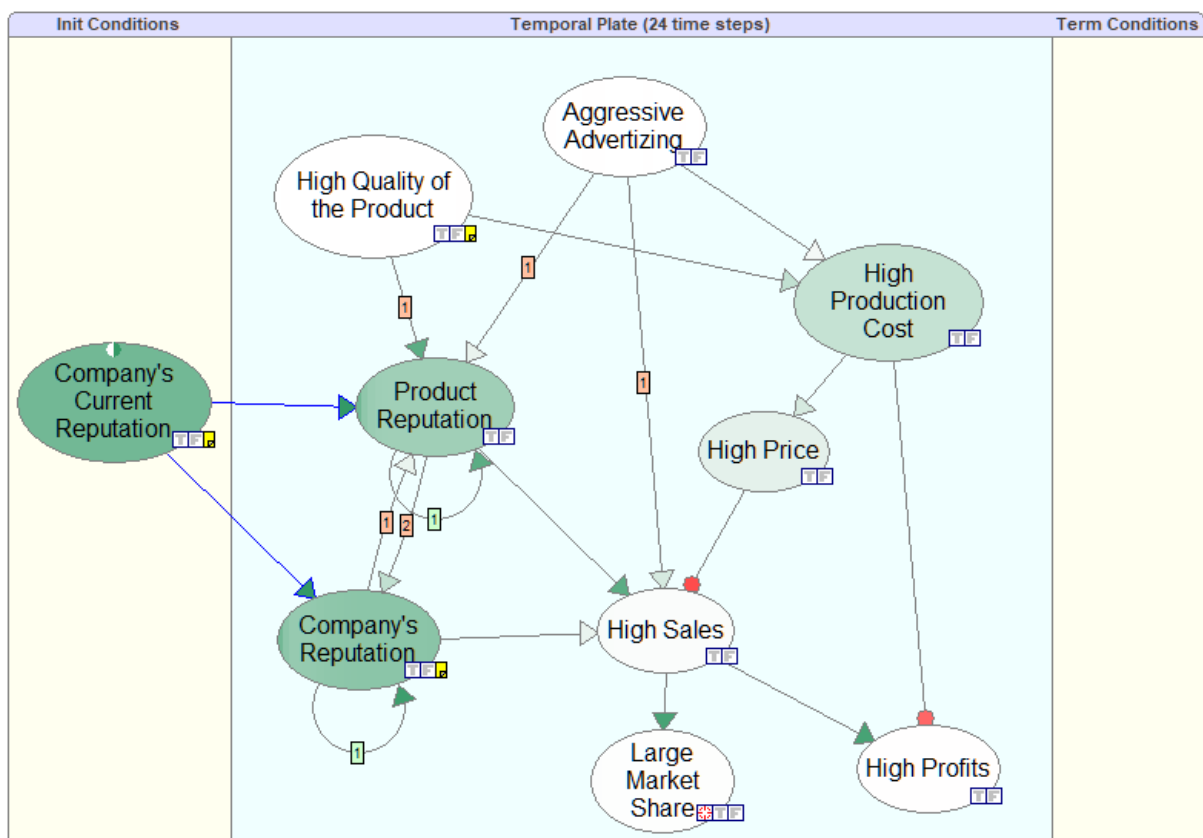
We select *Cause* for this interaction, which results in a normal (static) arc between the two variables. The next arc to add will be a temporal arc between the nodes *Product Reputation* and *Aggressive Advertizing*. We start again with clicking on the *Arc* button (), subsequently clicking on *Aggressive Advertizing* and dragging and releasing the mouse inside the node *Product Reputation*.



This time we want to indicate that the influence of *Aggressive Advertizing* on *Product Reputation* is not immediate but takes one time step. We select a (temporal) Order 1 arc that we subsequently designate as a Cause in the arc type dialog.



We continue adding arcs between nodes, quantify them (the model *Product Temporal.qdsl* can be found among the example models) and create the following structure



There is one type of temporal arcs inside the *Temporal Plate* that was not allowed in static models, namely arcs starting and ending at the same node and, by this forming cycles. Cycles represent

temporal processes and are allowed only for temporal arcs. Similarly to static networks, normal arcs are not allowed to form cycles, even inside temporal plates. Arcs in *Temporal Plate* can start and end in the same node. We create such arcs by starting and ending the drawing in the same node. These arcs essentially mean that the state of the variable in question at time t influences its state at time $t+1$. For example, reputation gained (*Product Reputation* and *Company's Reputation*) tends to stick. Dynamic models allow for cycles. For example, we have that *Product Reputation* influences *Company's Reputation* and *Company's Reputation* influences *Product Reputation*. The interpretation of such cycles is simple: Any temporal influence links variable in two different time slices and when these slices are unrolled, there are no cycles.

Please note there while most of the temporal arcs are labeled with [1], meaning that they are temporal influences of the first order, there is one arc in the model (from *Product Reputation* to *Company's Reputation*) that is labeled [2], representing a temporal influence of order 2. QGeNIe is unique among the existing Bayesian network software in that it allows for any order influences, which means that DBNs in QGeNIe can model dynamic processes of any order.

Quantification of this model proceeds in the same way as quantification of static models (please see the section on [Building a qualitative Bayesian network](#) for examples).

This concludes the creation and specification of the DBN modeling the problem. There is another way of creating a DBN. Rather than constructing it directly in the *Temporal Plate*, we can construct a BN in the *Graph View* window, drag it into the *Temporal Plate*, and then add temporal links. This has to be done cautiously, as the order of dragging can make a difference. For example, if we drag a node into the *Temporal Plate*, QGeNIe will remove all its outgoing arcs, as it is not allowed to have arcs from temporal plate enter nodes in the *Contemporals* section. To avoid that, we can best drag entire groups of nodes into the temporal plate, in which case no links will be deleted.

In the next section, we will show how to use dynamic models to obtain insight into the modeled domain.

6.5.3 Inference in DBNs

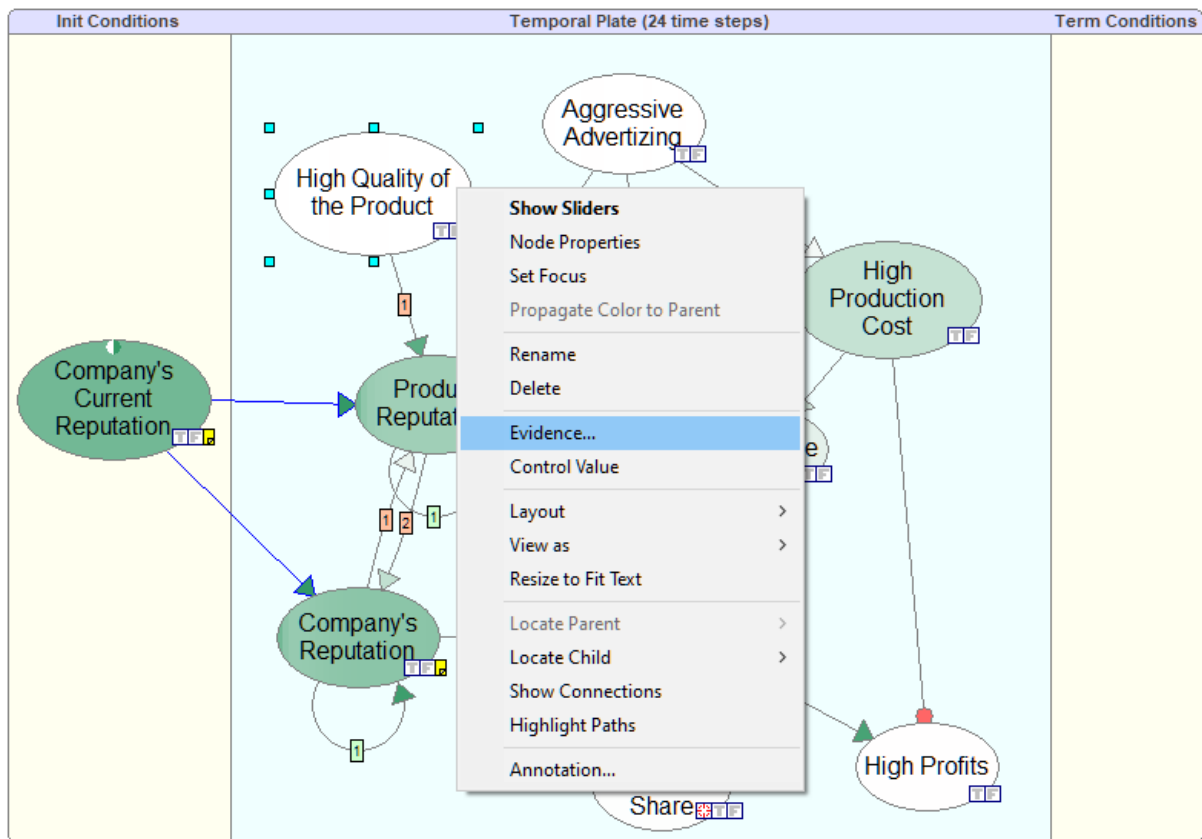
Inference in a DBN, similarly to inference in a BN, amounts to calculating the impact of observation of some of its variables on the probability distribution over other variables. The additional complication is that both evidence and the posterior probability distribution is indexed by time. We will go through the example used in the previous sections to demonstrate setting evidence, running an algorithm, and viewing the results.

Setting temporal evidence

We can enter into the *Product Temporal.qdsl* model evidence stating that the initial six months of the time horizon the product is going to be of poor quality. This means that the evidence vector for the node *High Quality of the Product* is as follows:

$$\text{High Quality of the Product}[0:6] = [\text{false}, \text{false}, \text{false}, \text{false}, \text{false}, \text{false}].$$

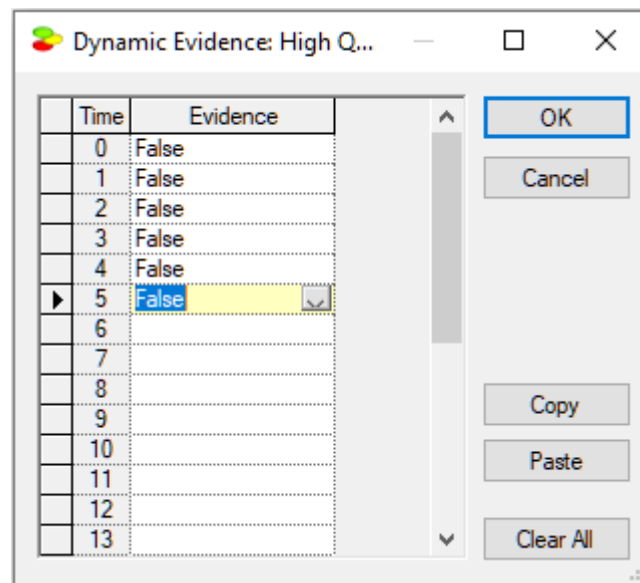
To enter this evidence, we right-click on the *High Quality of the Product* node and select *Evidence...*



This invokes the *Dynamic Evidence* dialog

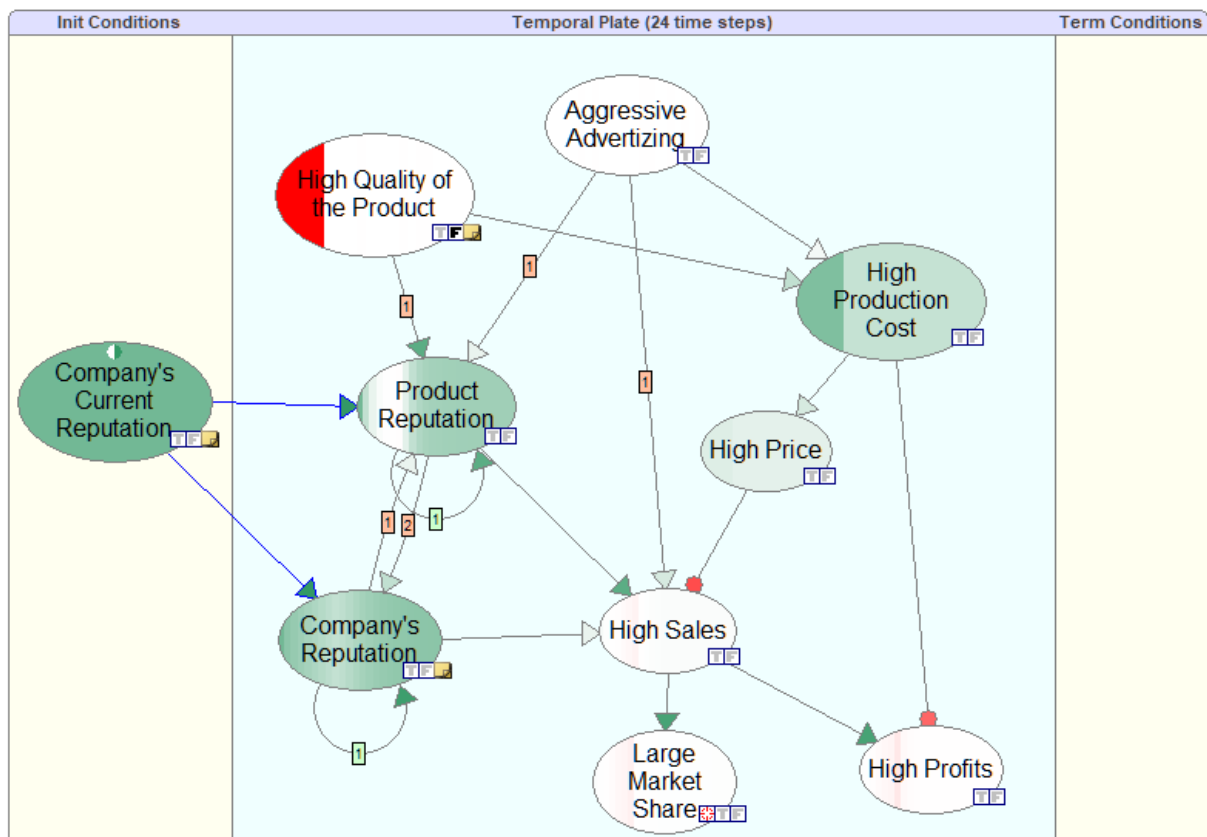
Time	Evidence
0	[Dropdown]
1	[Dropdown]
2	[Dropdown]
3	[Dropdown]
4	[Dropdown]
5	[Dropdown]
6	[Dropdown]
7	[Dropdown]
8	[Dropdown]
9	[Dropdown]
10	[Dropdown]
11	[Dropdown]
12	[Dropdown]
13	[Dropdown]

We enter the evidence vector as specified above:



Viewing results: Node gradients

Once we exit the *Dynamic Evidence* dialog, QGeNIe calculates the impact of the observations. The following screen shot of the model shows how the truth of the propositions represented by the nodes changes with time.



We can see that the first six months of poor quality product resulted in lower production costs but also in lowering the reputation of the product and lowering the company's reputation.

Viewing results: Active time step

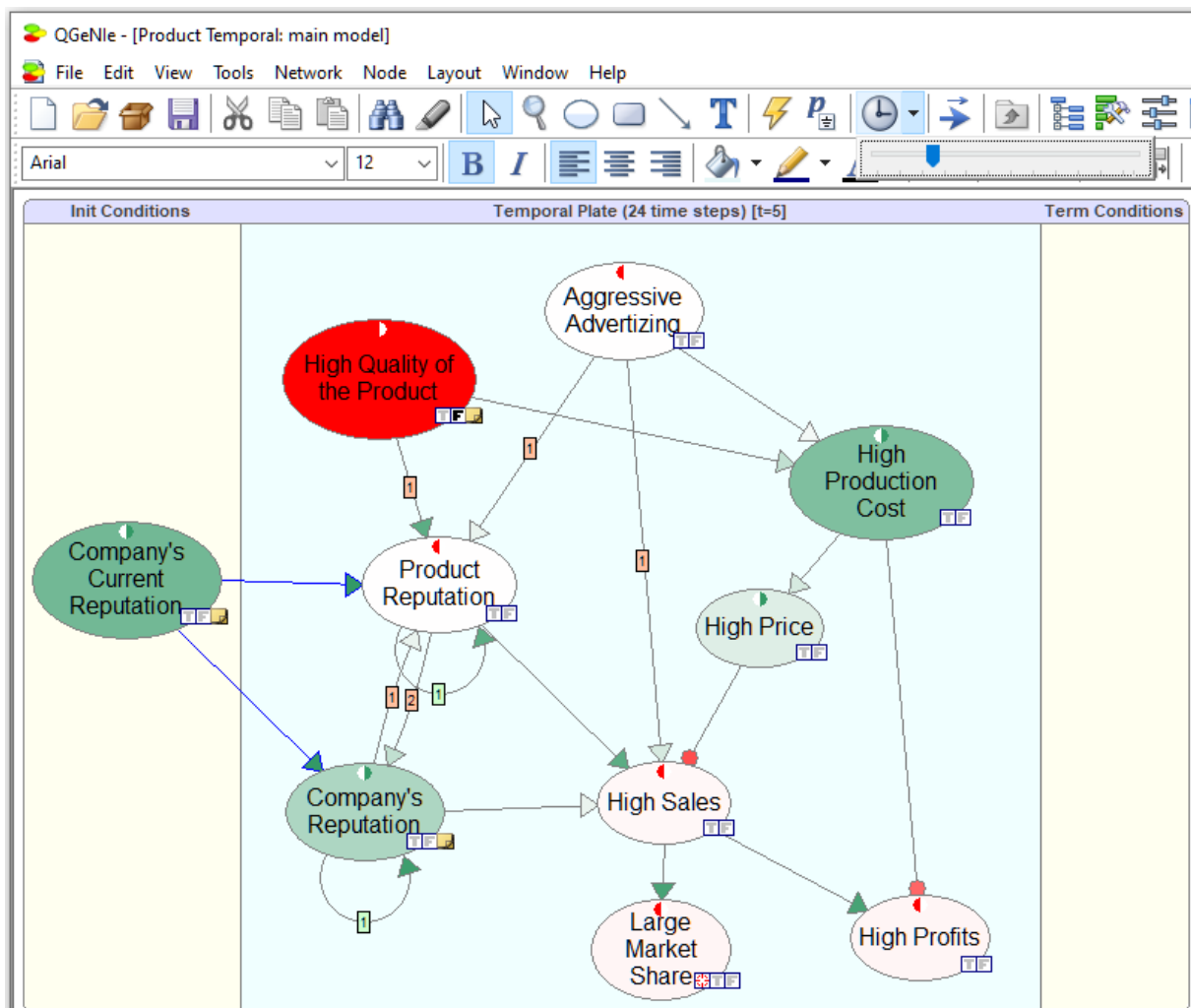
There is an additional, way of observing the results. QGeNle allows for walking through the time steps and observing how the colors of nodes change. This is invoked by pressing the *Active time step* icon (



). This shows a small slides for walking through the time steps.

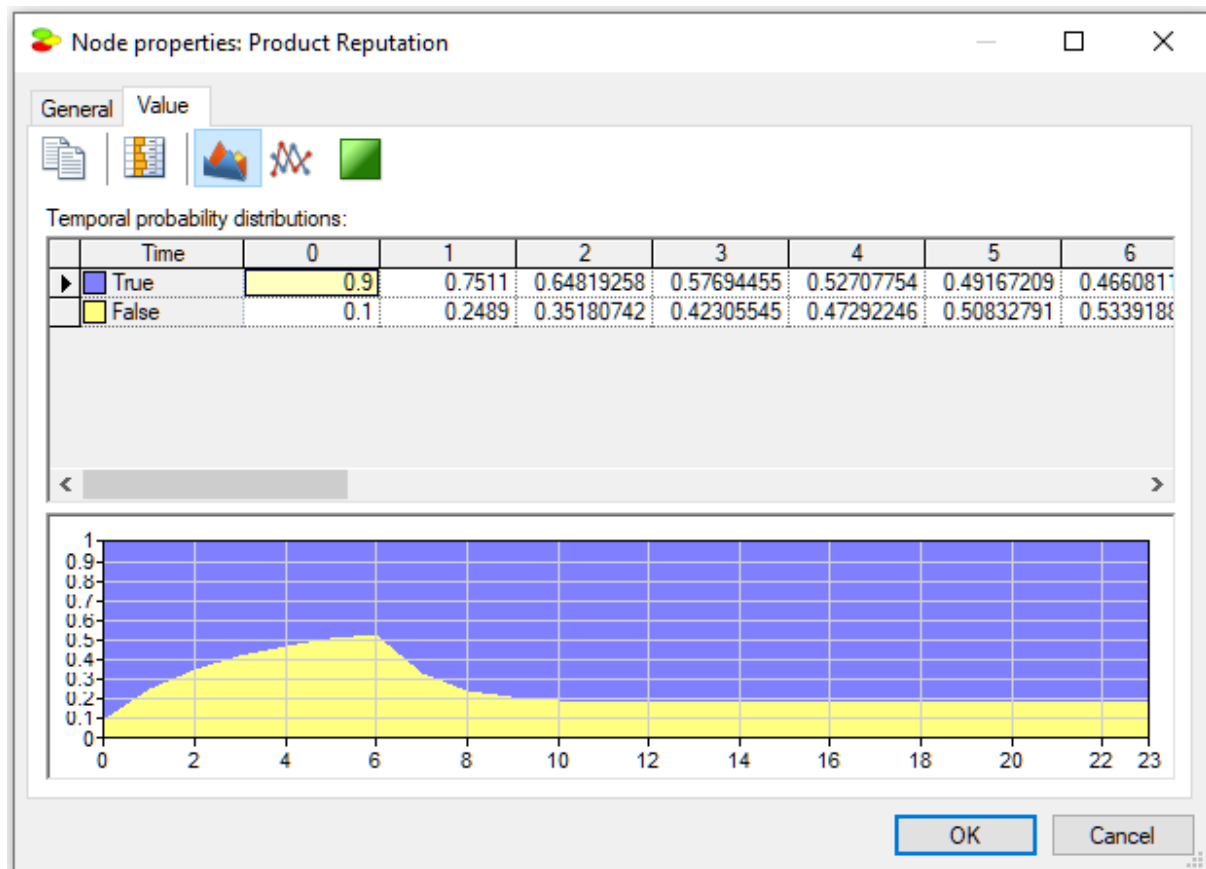


As we move the slider, the colors of nodes change. The following screen shot shows the colors (i.e., probabilities) at $t=6$. The time step is displayed in the header line of the temporal plate.

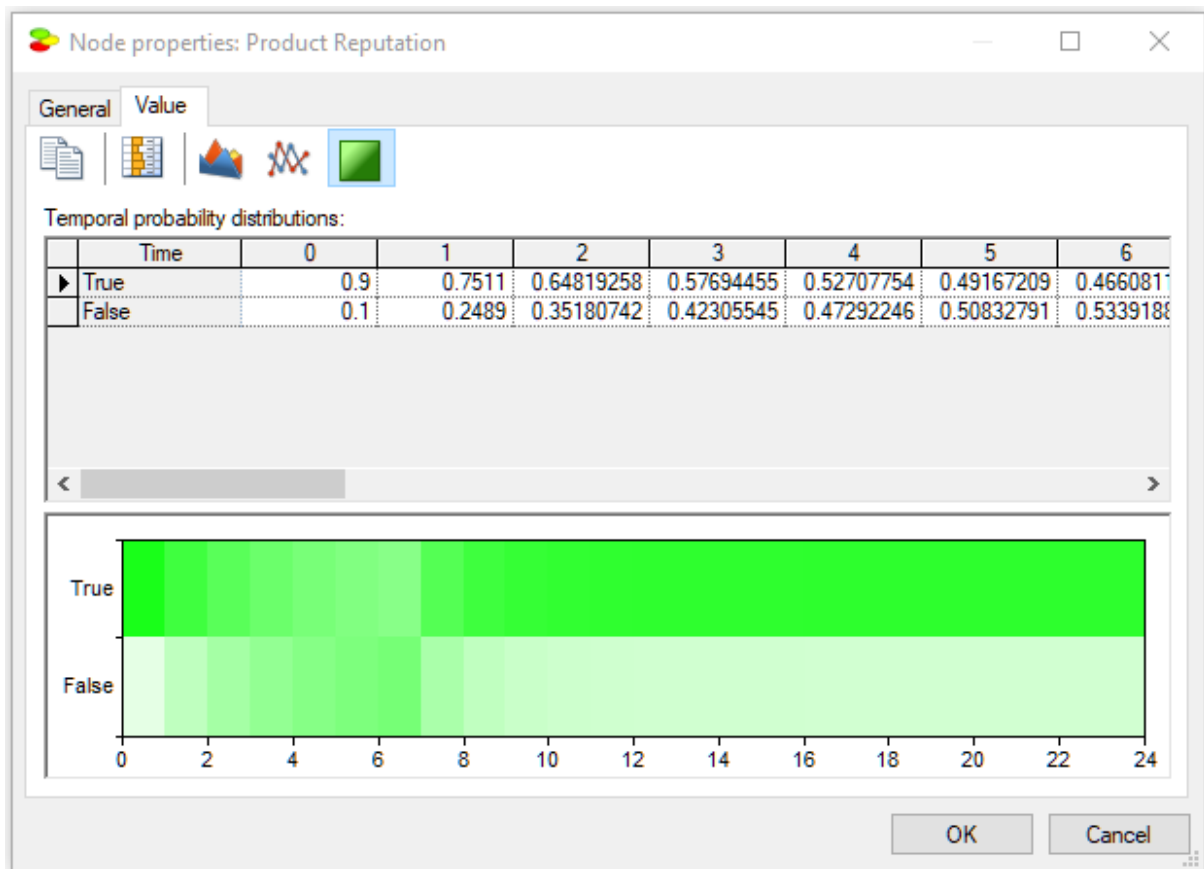



Viewing results: Value tab

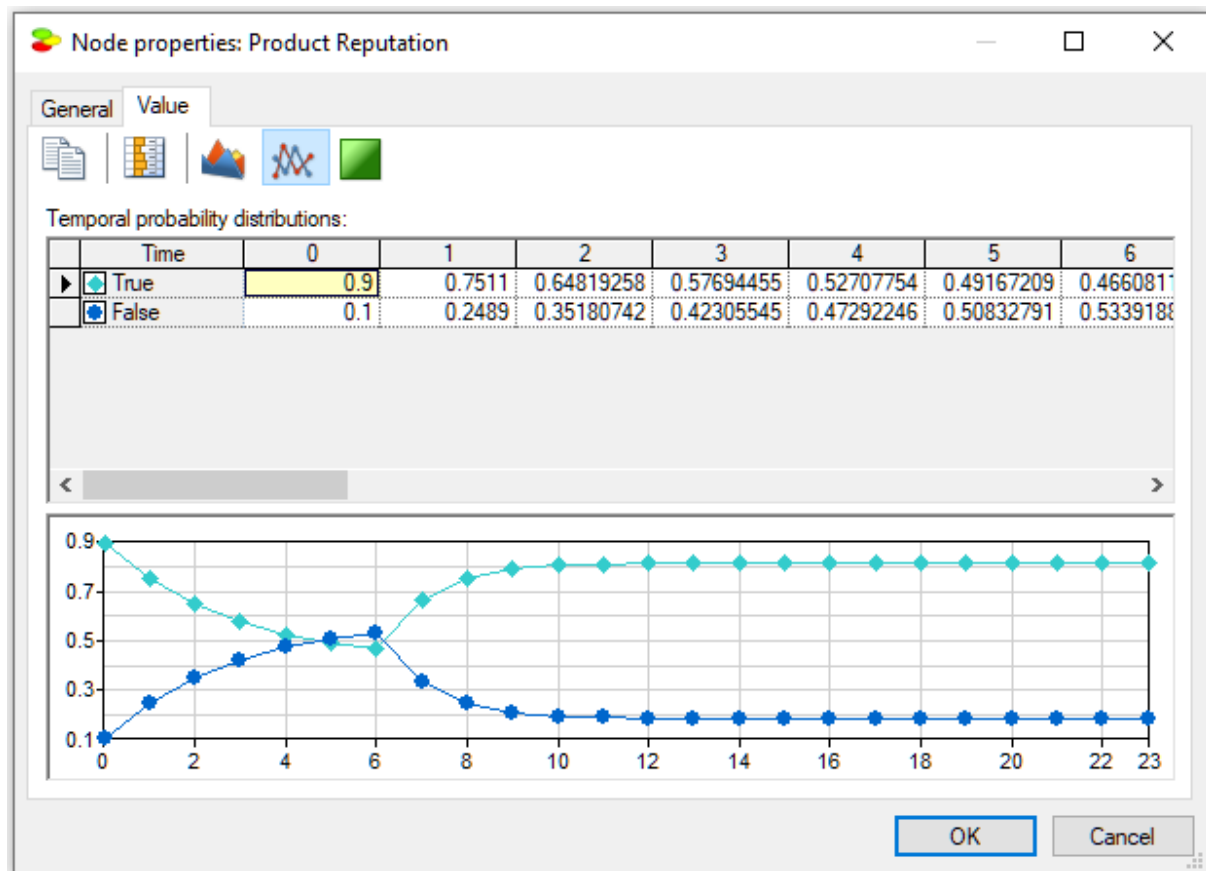
We can view the temporal beliefs in the *Value Tab* of a node as a spreadsheet indexed by the time steps. In addition to the numerical values of the marginal posterior probabilities over time, we can view the results graphically. Pressing the *Area chart* (📊) button displays the posterior marginal probabilities graphically:



Pressing the *Contour plot* (📈) button displays the posterior marginal beliefs as a contour plot with probabilities displayed by colors. Hovering over individual areas shows the numerical probabilities corresponding to the areas/colors.

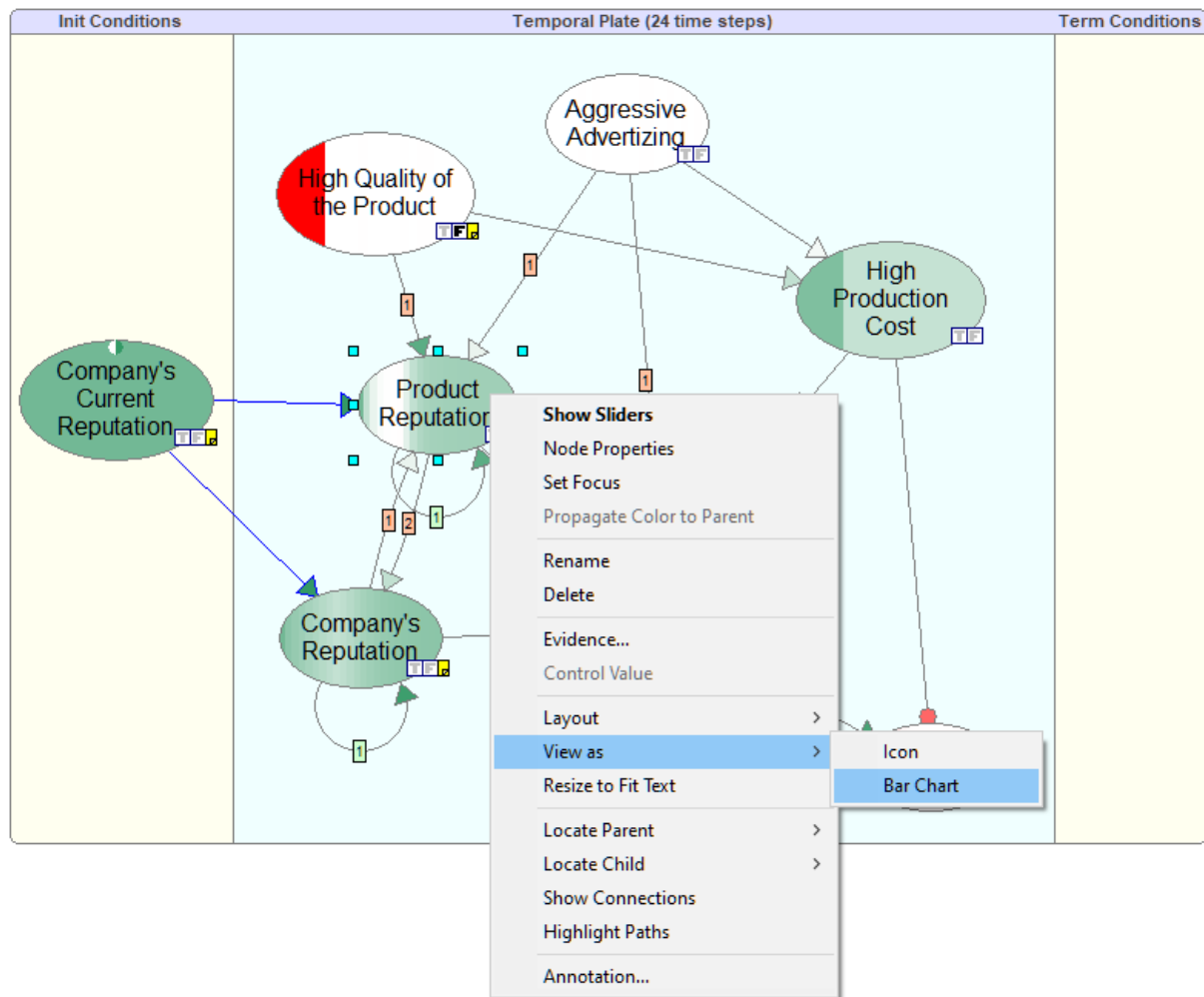


Finally, pressing the Time series () button shows the posteriors as a time series plot (a curve for each of the two states of the variable):



Viewing results: On-screen bar charts

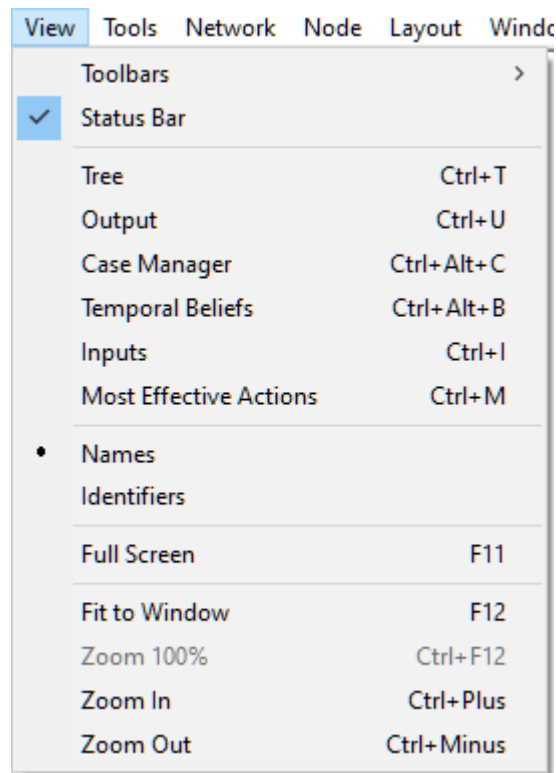
Marginal posterior probabilities can be also shown on the screen permanently by the changing the node view to *Bar chart*. This can be accomplished by selecting nodes of interest and then changing the view of the nodes through the *View as/Bar Chart* option.



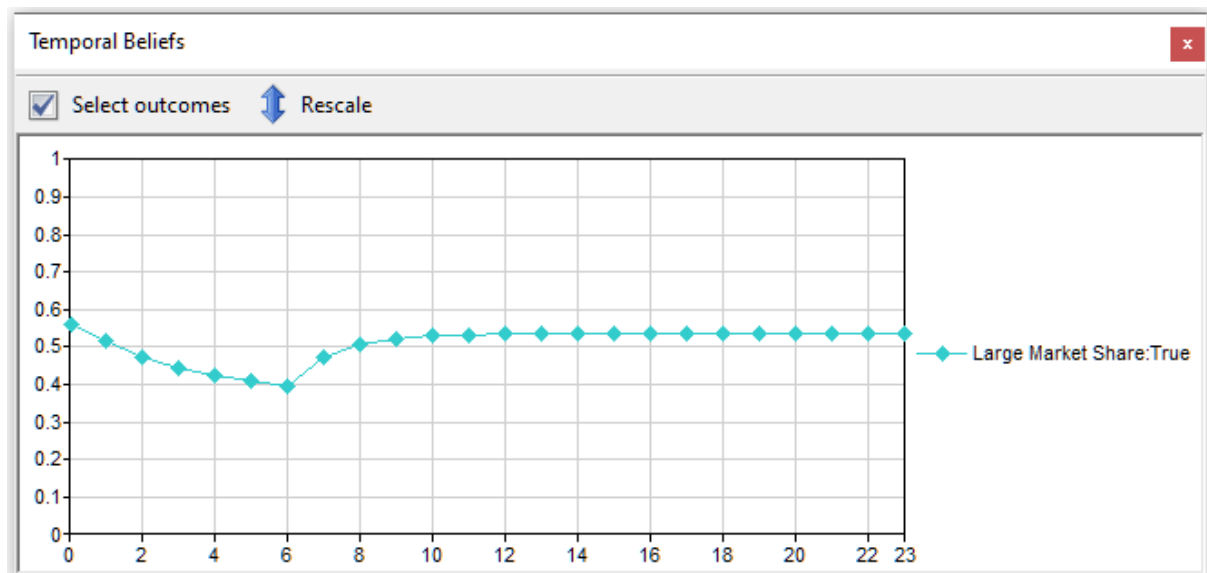
The *Bar chart* view allows for displaying the temporal posterior marginal probabilities on the screen permanently.



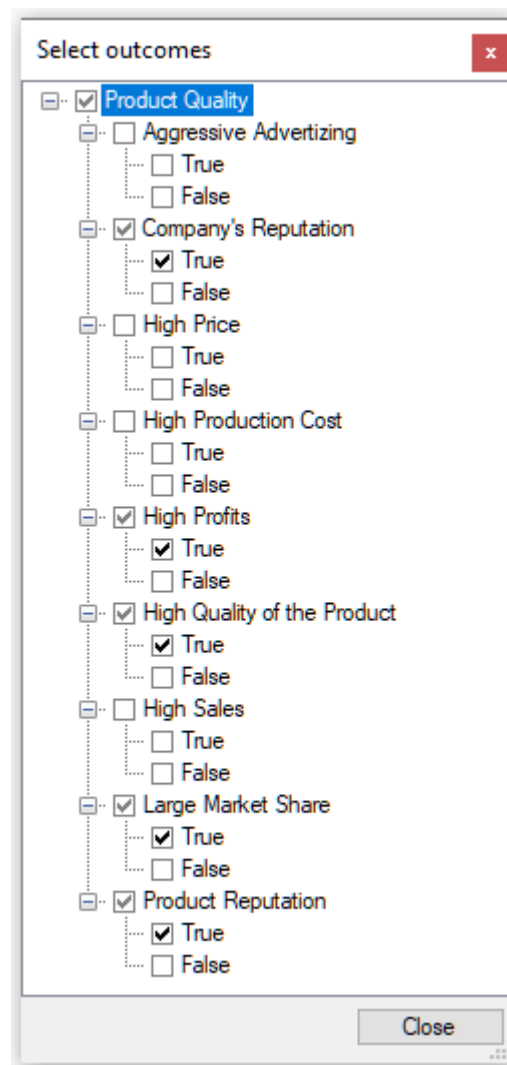
QGeNle Modeler



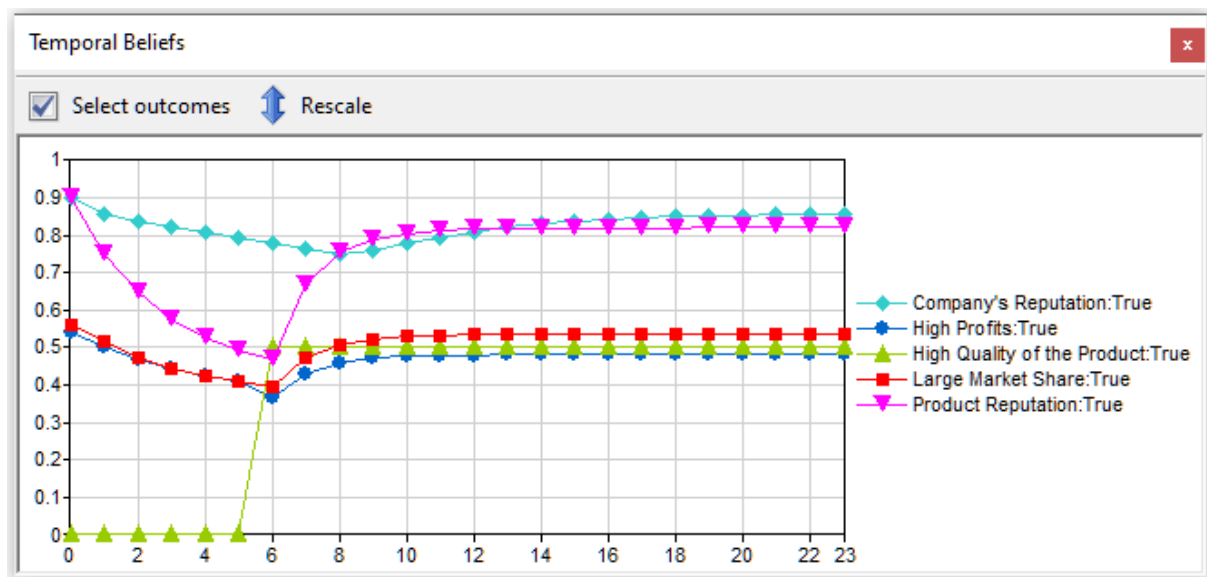
This opens the *Temporal Beliefs* window with the probability of the *Focus* node (in our model, it is the node *Large Market Share*) displayed



We can select propositions for which probabilities we want displayed by pressing the *Select outcomes* button on the top on this window. This pens a dialog that allows for selecting individual node states.

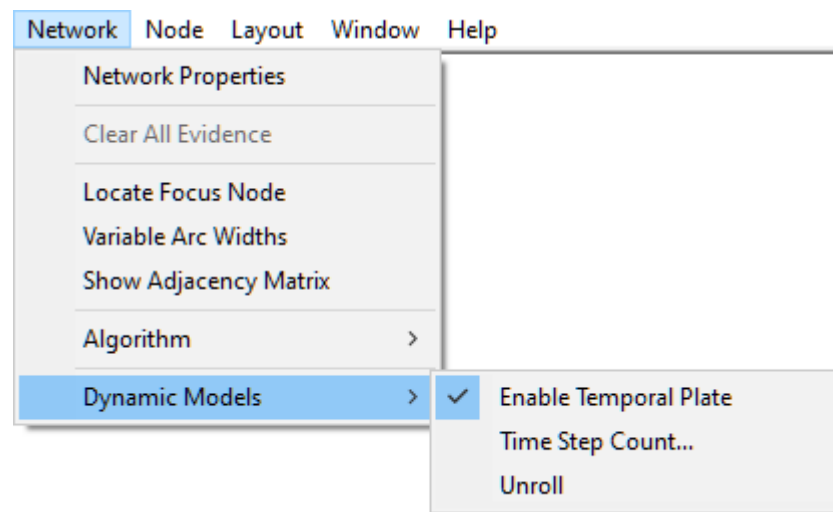


As we select states to be displayed, the *Temporal Beliefs* window shows them

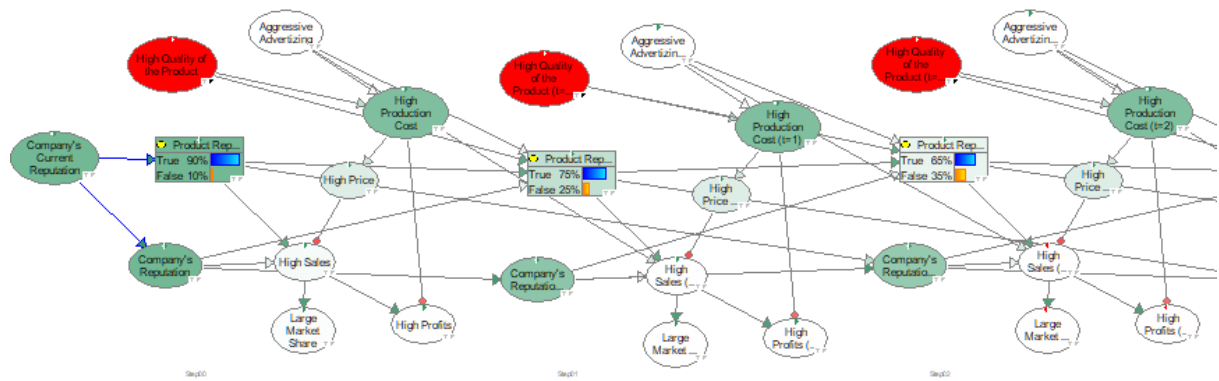


Unrolling the DBN

As we mentioned above, for the purpose of inference, QGeNIe converts the DBN into a Bayesian network and updates the beliefs using the selected belief updating algorithm. It can be useful, for example for model debugging purposes, to explicitly unroll a temporal network. QGeNIe provides this possibility through the *Network-Dynamic Models-Unroll* option.



QGeNIe creates a new network that has the temporal network unrolled for the specified number of time-slices. It is possible to locate a node in the temporal network from the unrolled network by right-clicking on the node in the unrolled network and selecting -> Locate Original in DBN from the context-menu. The unrolled network that is a result from unrolling the temporal network is cleared from any temporal information whatsoever. It can be edited, saved and restored just like any other static network. Figure below shows the unrolled network representation of a temporal network and how the original DBN can be located back from the unrolled network.



This page is intentionally left blank.
Remove this text from the manual
template if you want it completely blank.

Resources

7 Resources

7.1 Books

A good source of elementary knowledge of [Bayesian networks](#) is the path-breaking book by Pearl (1988). A good, thorough synthesis of the theoretical aspects of Bayesian networks and probabilistic decision support systems, useful for readers interested in building a reasoning system from the ground up, is the book by Neapolitan (1990), regrettably out of print. Jensen's (1996) book is heartily recommended as a good source of knowledge for both builders and users of Bayesian reasoning systems. A more recent addition to the books on graphical modeling is Cowell, Dawid, Lauritzen & Spiegelhalter (1999), a heartily recommended reading for anybody who wants to develop a thorough understanding of the methods that are at the foundation of graphical probabilistic systems.

de Finetti (1970) and Savage (1954) are recommended for the foundations of [Bayesian probability theory](#) and decision theory.

Readers interested in practical aspects of decision support, especially in the context of policy making, are recommended the superb book by Morgan & Henrion (1989).

Russell and Norvig (1995) is a good textbook covering application of probabilistic methods in Artificial Intelligence.

For readers interested in graphical probabilistic models in general, the thorough book by Whittaker (1989), covering directed and undirected probabilistic graphs, may be of interest.

An up to date list of textbooks covering the field of [decision analysis](#) can be found on [BayesFusion's web site](#).

7.2 Research papers

While there are a good number of excellent papers covering the topic of decision-analytic decision support, here are some of our favorites.

An introductory paper on [Bayesian networks](#), useful for beginners is (Charniak, 1991).

Overview papers by Horvitz et al. (1988), Cooper (1989), Henrion et al. (1991), Spiegelhalter et al. (1993) and Matzkevich & Abramson (1995) are accessible introductions to the use of probabilistic and decision-analytic methods in decision support systems.

Users interested in practical applications of Bayesian networks are directed to the March 1995 special issue of the *Communications of the ACM* journal, edited by Heckerman, Mamdani and Wellman (1995).

(Henrion, 1987) is a manifesto arguing convincingly for the use of probabilistic methods in artificial intelligence.

(Henrion, 1989) is a practical introduction to problems related to building probabilistic models. Another place to look at is a special issue of the *IEEE Transaction of Knowledge and Data Engineering* journal on building probabilistic models (Druzdzel & van der Gaag, 2000).

7.3 Conferences

Probably the best source for the state of the art research in graphical probabilistic models are proceedings of the annual *Conference on Uncertainty in Artificial Intelligence (UAI)*. Proceedings of UAI conferences are available electronically from Decision System Laboratory's web site located at

<https://dslpitt.org/uai/>

A similarly prestigious conference on the topic of probabilistic graphical models is the bi-annual *European Conference on Probabilistic Graphical Models (PGM)*. It brings together researchers interested in all aspects of graphical models for probabilistic reasoning, decision making, and learning. The web site for the 2018 conference, listing web sites for all previous PGM conference and their on-line proceedings, is at the following location:

<http://pgm2018.utia.cz/>

Another conference devoted completely to Bayesian networks is the *Annual Conference of the Australasian Bayesian Network Modelling Society (ABNMS)*. The web site for the 2019 conference, listing web sites for all previous ABNMS conference is at the following location:

<http://abnms.org/conferences/abnms2019/>

Other relevant conferences are *AAAI National Conferences on Artificial Intelligence (AAAI)*, *International Joint Conferences on Artificial Intelligence (IJCAI)*, *Conferences on Knowledge Discovery and Data Mining (KDD)*, *Workshop on Artificial Intelligence and Statistics, Uncertainty Track of the Florida Artificial Intelligence Conferences (FLAIRS)*, *European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty (ECSQARU)*, *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU)*, and *European Conference on Artificial Intelligence (ECAI)*.

7.4 Model repository

An important class of World Wide Web resources that may be of interest to QGeNIe users are model repositories. They are a great inspiration to model builders and source of models for the purpose of testing algorithms. Here is where you can find BayesFusion's model repository:

<https://repo.bayesfusion.com/>

The repository is interactive and allows for entering evidence and performing Bayesian updating. Furthermore, you should be able to download any of the models included in the repository to a local disk and open it with QGeNIe. BayesFusion's interactive model repository is powered by [BayesBox](#), another BayesFusion's product, allowing our clients to create local and secure model repositories.

7.5 Social Media

Please visit [BayesFusion's YouTube channel](#) for useful recordings that may help you in learning more about decision-theoretic methods in intelligent systems, GeNIe and QGeNIe.

We use [BayesFusion's Twitter account](#) to communicate with our users important news, such as new software releases. Following us on Twitter will ensure that you are up to date on what is happening at BayesFusion.

[BayesFusion's Facebook account](#) contains important news releases. We make sure that important news that are listed on our web site find their way to our Facebook page.

[BayesFusion's LinkedIn account](#) contains basic information about BayesFusion. We have started a LinkedIn group [GeNIe Users](#) that is an open forum for our uses.

Finally, [BayesFusion Forum](#) is a good place to interact with us and other GeNIe and SMILE users.

7.6 References

Charniak, Eugene (1991). *Bayesian networks without tears*. AI Magazine, 12(4):50-63.

Cooper, Gregory F. (1989). *Current research directions in the development of expert systems based on belief networks*. Applied Stochastic Models and Data Analysis, 5(1):39-52.

Cooper, Gregory F. (1990). *The computational complexity of probabilistic inference using Bayesian belief networks*. Artificial Intelligence, 42(2-3):393-405.

Cowell, Robert G., A. Philip Dawid, Steffen L. Lauritzen & David J. Spiegelhalter (1999). *Probabilistic Networks and Expert Systems*. Springer-Verlag New York, Inc.: New York, NY.

Dagum, Paul & Michael Luby (1993). *Approximating probabilistic inference in Bayesian belief networks is NP-hard*. Artificial Intelligence, 60(1):141-153.

Dawes, Robyn M. (1988). *Rational Choice in an Uncertain World*. Hartcourt Brace Jovanovich, Publishers.

Dawid, A. Philip (1979). *Conditional independence in statistical theory*. Journal of the Royal Statistical Society, Series B (Methodological), 41:1-31.

Dawid, A. Philip (1992). *Applications of a general propagation algorithm for probabilistic expert systems*. Statistics and Computing, 2:25-36.

de Finetti, Bruno (1970). *Theory of Probability*. John Wiley and Sons, New York.

Marek J. Druzdzel (2009). *Rapid modeling and analysis with QGeNIe*. In Proceedings of the International Multiconference on Computer Science and Information Technology (IMCSIT-2009), Mragowo, Poland, pages 101-108.

Diez, F. Javier & Marek J. Druzdzel (2006). *Canonical probabilistic models for knowledge engineering*. Technical Report CISIAD-06-01. UNED, Madrid, 2006 (available at <http://www.ia.uned.es/~fjdiez/papers/canonical.html>).

Druzdzel, Marek J. & Linda C. van der Gaag (2000). *Building probabilistic networks: 'Where do the numbers come from?'* Guest editors' introduction. IEEE Transactions on Knowledge and Data Engineering, 12(4):481-486.

Eades, P. (1984). *A heuristic for graph drawing*. Congressus Numerantium, 41, page 149–160.

Hayes, B. C. and J. I. Sands (1997). *Doing windows: Non traditional military responses to complex emergencies*, Decision Support Department, Center for Naval Warfare Studies, U.S. Naval War College, Newport, RI, USA, DSD Research Report 97–1.

Heckerman, David, Abe Mamdani & Michael P. Wellman (1995). *Real-world applications of Bayesian networks*. Communications of the ACM, 38(3):24-26.

Henrion, Max (1987). *Uncertainty in Artificial Intelligence: Is Probability Epistemologically and Heuristically Adequate?* In: Mumpower J.L., Renn O., Phillips L.D., Uppuluri V.R.R. (eds) Expert Judgment and Expert Systems. NATO ASI Series (Series F: Computer and Systems Sciences), vol 35. Springer, Berlin, Heidelberg

Henrion, Max (1989). *Some practical issues in constructing belief networks*. Kanal, L.N., Levitt, T.S. & Lemmer, J.F. (eds.), Uncertainty in Artificial Intelligence 3. Elsevier Science Publishers B.V. (North Holland), pages 161-173.

Henrion, Max (1990). *An introduction to algorithms for inference in belief nets*. In Henrion, M., Shachter, R.D., Kanal, L.N. & Lemmer, J.F. (eds.), Uncertainty in Artificial Intelligence 5, Elsevier Science Publishers B.V. (North Holland), pages 129-138.

Henrion, M., John S. Breese & Eric J. Horvitz (1991). *Decision analysis and expert systems*. AI Magazine, 12(4):64-91.

Horvitz, Eric J., John S. Breese & Max Henrion (1988). *Decision theory in expert systems and artificial intelligence*. International Journal of Approximate Reasoning. 2(3):247-302.

Hulst, Joris (2006). *Modeling physiological processes with dynamic Bayesian networks*. M.Sc. thesis, Delft University of Technology, Delft, The Netherlands.

Jensen, Finn V. (1996). *An Introduction to Bayesian Networks*. Springer Verlag, New York.

Jensen, Finn V., Steffen L. Lauritzen & Kristian G. Olsen (1990). *Bayesian updating in recursive graphical models by local computations*. Computational Statistics Quarterly, 4:269-282.

Kernighan, Brian W. & Dennis M. Ritchie (1988). *The C Programming Language*. Prentice Hall PTR, 2nd edition.

Lauritzen, Steffen L. & David J. Spiegelhalter (1988). *Local computations with probabilities on graphical structures and their application to expert systems* (with discussion). Journal of the Royal Statistical Society, Series B (Methodological), 50(2):157-224.

Lin, Yan & Marek J. Druzdzel (1997). *Computational advantages of relevance reasoning in Bayesian belief networks*. In Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence (UAI-97), Morgan Kaufmann: San Mateo, CA, pages 342-350.

Paul P. Maaskant (2006). *A causal model for qualitative reasoning*. M.Sc. thesis, Man-machine interaction group, Delft University of Technology (available at <http://www.kbs.twi.tudelft.nl/Publications/MSc/2006-PaulMaaskant-Msc.html>).

Paul P. Maaskant and Marek J. Druzdzel (2008). *An ICI Model for opposing influences*. In Proceedings of the Fourth European Workshop on Probabilistic Graphical Models (PGM-08), Manfred Jaeger & Thomas D. Nielsen (eds.), Hirtshals, Denmark, pages 185-192.

Matzkevich, Izhar & Bruce Abramson (1995). *Decision analytic networks in artificial intelligence*. Management Science, 41(1):1-22.

Morgan, M. Granger & Max Henrion (1990). *Uncertainty: A Guide to Dealing with Uncertainty in Quantitative Risk and Policy Analysis*. Cambridge University Press, Cambridge.

Murphy, Kevin P. (2002). *Dynamic Bayesian Networks: Representation, Inference and Learning*. Doctoral dissertation, University of California, Berkeley.

Neapolitan, Richard E. (1990). *Probabilistic Reasoning in Expert Systems: Theory and Algorithms*. John Wiley & Sons, New York.

Pearl, Judea (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, Inc., San Mateo, CA.

Aníbal Pérez-Liñán (2008). “El dilema estrategico de la seguridad ciudadana y el Estado democrático de derecho.” In Alexandra Steinmetz (ed.), Informe Estado de la Región. State of the Nation—Region Program, Pavas, Costa Rica, <http://www.estadonacion.or.cr/> Chapter 12, pages 460–524.

Quinn, N.R., Jr & M.A. Breuer (1979). *A force directed component placement procedure for printed circuit boards*. IEEE Transactions on Circuits and Systems, CAS 26, pages 377–388.

Savage, Leonard (1954). *The Foundations of Statistics*. Dover, New York.

Simon, Herbert A. (1996). *The Sciences of the Artificial*. 3rd edition. MIT Press.

Spiegelhalter, David J., A. Philip Dawid, Steffen L. Lauritzen & Robert G. Cowell (1993). *Bayesian analysis in expert systems*. Statistical Science, 8(3):219-283.

Whittaker, Joe (1990). *Graphical Models in Applied Multivariate Statistics*. John Wiley & Sons, Chichester.

Yuan, Changhe & Marek J. Druzdel (2003). *An Importance Sampling Algorithm Based on Evidence Pre-propagation*. In Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence (UAI-03), Morgan Kaufmann: San Mateo, CA, pages 624-631.

This page is intentionally left blank.
Remove this text from the manual
template if you want it completely blank.

- A -

about GeNIe 34
 acknowledgments 43
 acknowledgment 42
 acyclic directed graph 47
 annotation 110
 Apple 39
 applications 168
 approximate belief updating 50
 arc 106, 182
 arc thickness 182

- B -

Bayesian approach 47
 Bayesian belief network 47
 Bayesian inference 50
 Bayesian network 47
 Bayesian probability 46
 Bayesian updating 50
 belief network 47
 belief updating 50
 books 230
 bug reporting 42
 building a qualitative Bayesian network 10
 building blocks of QGeNIe 54

- C -

Canonical Nodes 80, 95
 case manager 199
 cases 199
 CAST gate 95
 CAST node 79
 causal probabilistic network 47
 Chang 50
 changes in structure 50
 clustering algorithm 164
 conferences 231
 console 77
 controlling values 175
 Controlled status icon 107
 copyright notice 42
 create a node 10
 creating a qualitative Bayesian network 10

- D -

DBN 208, 209, 215
 decision analysis 46
 define properties 10
 del Favero 50
 DeMorgan gate 80
 DeMorgan node 79
 disclaimer 42

- E -

edge 106
 effects of changes 50
 effects of changes in structure 50
 enabling most effective actions calculation 203
 entering evidence 172
 EPIS 164
 EPIS Sampling 164
 error messages 77
 evidence 172
 example networks 37

- F -

Facebook 232
 File Menu 155
 Focus status icon 107
 Format Toolbar 134
 frequentist interpretation 46
 full-screen mode 132
 Fung 50

- G -

general tab 121
 graph layout 129, 138
 Graph View 57

- H -

Hardware and software requirements 37
 Help Menu 78
 Henrion 50

- I -

impact of observing values 50
Implied status icon 107
inference 215
inference algorithms 160
introduction 168
iOS 39

- J -

joint-tree algorithm 164

- K -

keyboard shortcuts 164
knowledge engineering 209

- L -

Layout Menu 129, 138
LinkedIn 232
literature list 232
loading models 148

- M -

Mac OS 39
MacIntosh 39
manipulable 202
manipulation 50, 175
marginal probability distribution 179
Menu Bar 56
model repository 231
models 231
most effective actions calculation 202, 205
movies 232

- N -

navigation 145
Network Menu 162
Network Property Sheets 112
Node Menu 161
node properties 202

Node Property Sheets 121
node status icons 107
node type 79

- O -

objectivist interpretation 46
observable 202
Observed status icon 107
observing 50
Output Window 77

- P -

parent ordering 129, 138
Peot 50
probability 46
propensity view 46

- Q -

QDSL file format 160
QGeNle version 37
QGeNle workspace 54
qualitative Bayesian network 170, 172, 175, 179, 182
qualitative Bayesian networks tutorial 10
qualitative dynamic Bayesian network 208, 209, 215

- R -

read me first 8
references 232
relevance-based decomposition 164
research papers 230
resources 230, 231, 232
retracting evidence 172

- S -

save model 10
saving models 148
selection 140
Shachter 50
size 132
SMILE 35
spring embedder 129, 138

- Standard Toolbar 119
- start here 8
- Status Bar 76
- stochastic sampling 50
- strength of influence 182
- subjectivist interpretation 46
- subjectivist view 46
- submodel 97
- submodel node 79
- Submodel Property Sheets 117

- T -

- text box 109
- Tools Menu 119
- transparent mode 121
- Tree View 72
- Twitter 232

- U -

- user properties 121
- using QGeNIe 168

- V -

- value tab 121
- view results 10
- viewing results 179

- W -

- warnings 77

- Y -

- YouTube 232

- Z -

- zooming 132