OPENMARKOV TUTORIAL

www.openmarkov.org

Version 0.2.0

June 15, 2017

CISIAD
UNED

This version has been written under the pressure of time. It needs a careful revision to improve the style, detect erratas, check the cross-references, etc. We will replace it with a new one as soon as possible.

Contents

List of Figures

List of Tables

Introduction

A probabilistic graphical model (PGM) consists of a probability distribution, defined on a set of variables, and a graph, such that each node in the graph represents a variable and the graph represents (some of) the independencies of the probability distribution. Some PGMs, such as Bayesian networks [25], are purely probabilistic, while others, such as influence diagrams [16], include decisions and utilities.

OpenMarkov (see `www.openmarkov.org`) is an open source software tool developed at the Research Center for Intelligent Decision-Support Systems (CISIAD) of the Universidad Nacional de Educación a Distancia (UNED), in Madrid, Spain.

This tutorial explains how to edit and evaluate PGMs using OpenMarkov. It assumes that the reader is familiar with PGMs. An accessible treatment of PGMs can be found in the book by Neapolitan [24]; other excellent books are [19, 8, 18, 20, 23]. In Spanish, an introductory textbook on PGMs, freely available on internet, is [9].

In this tutorial different fonts are used for different purposes:

- Sans Serif is used for the elements of OpenMarkov's graphical user interface (the menu bar, the network window, the OK button, the Options dialog...).
- *Emphasized style* is used for generic arguments, such as the names of variables (*Disease*, *Test...*) and their values (*present*, *absent...*), types of networks (*Bayesian network*, *influence diagram...*), etc.
- **Bold** is used for highlighting some concepts about probabilistic graphical models (**finding**, **evidence**...).
- `Typewriter` is used for file names, extensions, and URLs (`decide-therapy.pgmx`, `.csv`, `www.openmarkov.org`...).

Finally, we should mention that the sections marked with an asterisk (for example, Sec. 4.4) are intended only for advanced users.

Chapter 1

Bayesian networks: edition and inference

1.1 Overview of OpenMarkov's GUI

This section offers a brief overview of OpenMarkov's graphical user interface (GUI). The main screen has the following elements (see Figure 1.1):

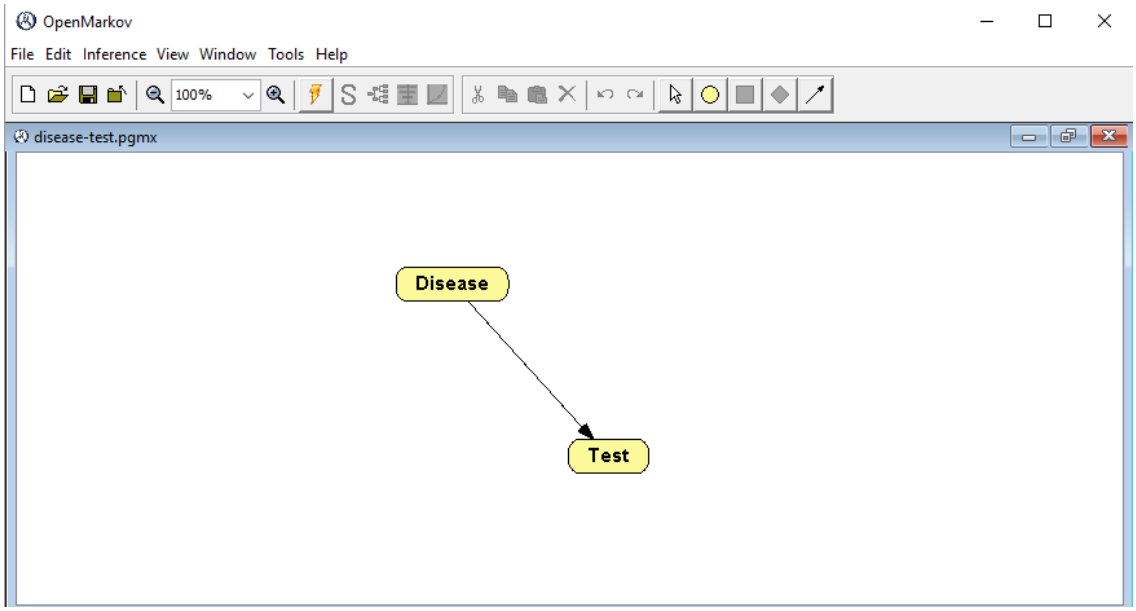


Figure 1.1: A Bayesian network consisting of two nodes.

1. The menu bar, which contains seven options: File, Edit, Inference, View, Window, Tools, and Help.
2. The first toolbar, located just under the menu bar, contains several icons for the main operations: Create new network (), Open (), Save (), Load evidence (), Zoom (100%), Inference (), Optimal strategy (), Decision tree (), Sensitivity analysis (), and Cost-effectiveness analysis ().
3. The edit toolbar, located at the right of the first toolbar, contains six icons for several operations: Cut (), Copy (), Paste (), Undo (), and Redo (), as well as five icons for choosing the edit tool, which may be one of the following:

- Select object ();
- Insert chance node ();
- Insert decision node ();
- Insert utility node ();
- Insert link ().

4. The network window is displayed below these toolbars. Figure 1.1 shows the network `BN-disease-test.pgmx`, which will be used as a running example along this tutorial.

1.2 Editing a Bayesian network

Let us solve the following problem using OpenMarkov.

Example 1.1. For a disease whose prevalence is 14% there exists a test with a sensitivity of 70% and a specificity of 91%. What are the predictive values of the test for that disease?

In this problem there are two variables involved: *Disease* and *Test*. The disease has two values or states, *present* and *absent*, and the test can give two results: *positive* and *negative*. Since the first variable causally influences the second, we will draw a link *Disease*→*Test*.

1.2.1 Creation of the network

In order to create a network in OpenMarkov, click on the icon Create a new network () in the first toolbar. This opens the Network properties dialog. Observe that the default network type is *Bayesian network* and click OK.

1.2.2 Structure of the network (graph)

In order to insert the nodes, click on the icon Insert chance node () in the edit toolbar.

Then click on the point were you wish to place the node *Disease*, as in Figure 1.1. Open the Node properties window by double-clicking on this node. In the Name field write *Disease*. Click on the Domain tab and check that the values that OpenMarkov has assigned by default are *present* and *absent*, which are just what we wished. Please note that we are following the convention of capitalizing the names of the variables and using lower case names for their states. Then click OK to close the window.

Instead of creating a node and then opening the Node properties window, it is possible to do both operations at the same time by double-clicking on the point where we wish to create the node. Use this shortcut to create the node *Test*. After writing the Name of the node, *Test*, open the Domain tab, click Standard domains and select negative-positive. Then click OK.

Complete the graph by adding a link from the first node to the second: select the Insert link tool () and drag the mouse from the origin node, *Disease*, to the destiny, *Test*. The result must be similar to Figure 1.1.

1.2.3 Saving the network

It is recommended to save the network on disk frequently, especially if you are a beginner in the use of OpenMarkov. The easiest way to do it is by clicking on the Save network icon (); given that this network has no name yet, OpenMarkov will prompt you for a File name; type *BN-disease-test* and observe that OpenMarkov will add the extension `.pgmx` to the file name, which stands for *Probabilistic Graphical Model in XML*; it indicates that the network is encoded in *ProbModelXML*, OpenMarkov's default format (see `www.ProbModelXML.org`).

1.2.4 Selecting and moving nodes

If you wish to select a node, click on the Selection tool icon ( ) and then on the node. You can select several nodes individually, press the Control or Shift key while clicking on them. It is also possible to select several nodes by dragging the mouse: the nodes whose center lies inside the rectangle drawn will be selected.

If you wish to move a node, click on the Selection tool icon and drag the node. It is also possible to move a set of selected nodes by dragging any of them.

1.2.5 Conditional probabilities

Once we have the nodes and links, we have to introduce the numerical probabilities. In the case of a Bayesian network, we must introduce a conditional probability table (CPT) for each node.

The CPT for the variable *Disease* is given by its prevalence, which, according with the statement of the example, is $P(Disease{=}present) = 0.14$. We introduce this parameter by right-clicking on the *Disease* node, selecting the Edit probability item in the contextual menu, choosing *Table* as the Relation type, and introducing the value *0.14* in the corresponding cell. If we leave the edition of the cell (by clicking a different element in the same window or by pressing Tab or Enter), the value in the bottom cell changes to 0.86, because the sum of the probabilities must be one (see Figure 1.2).

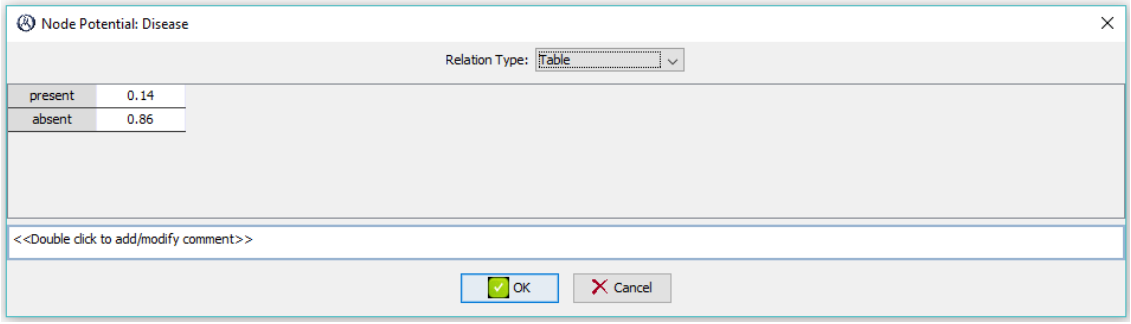


Figure 1.2: Conditional probability for the variable *Disease*.

The CPT for the variable *Test* is built by taking into account that the sensitivity (90%) is the probability of a positive test result when the disease is present, and the specificity (93%) is the probability of a negative test result when the disease is absent (see Figure 1.3):

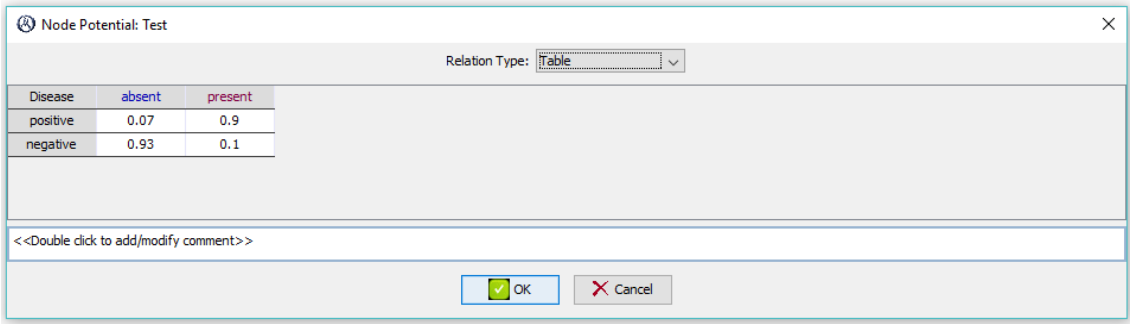


Figure 1.3: Conditional probability for the variable *Test*.

A shortcut for opening the potential associated to a node (either a probability or a utility potential) is to alt-click on the node.

1.3 Inference

Click the Inference button ( ) to switch from **edit mode** to **inference mode**. As the option propagation is set to *automatic* by default, OpenMarkov will compute and display the prior probability of the value of each variable, both graphically (by means of horizontal bars) and numerically—see Figure 1.4. Note that the Edit toolbar has been replaced by the Inference toolbar, whose buttons are described below.

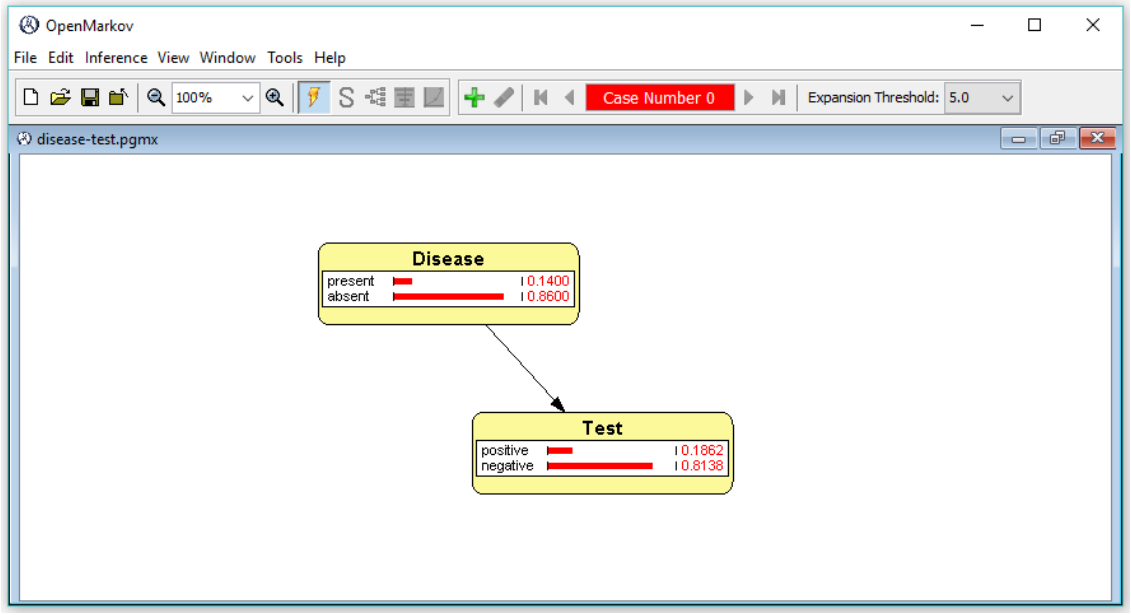


Figure 1.4: Prior probabilities for the network `BN-disease-test.pgmx`.

1.3.1 Entering findings

A **finding** is the assignment of a value to a variable as a consequence of an observation. A set of findings is called an **evidence case**. The probabilities conditioned on a set of findings are called **posterior probabilities**.

For example, a finding may be the test has given a positive result: $Test = positive$. Introduce it by double-clicking on the state *positive* of the node *Test* (either on the string, or on the bar, or on the numerical value) and observe that the result is similar to Figure 1.5: the node *Test* is colored in gray to denote the existence of a finding and the probabilities of its states have changed to 1.0 and 0.0 respectively. The probabilities of the node *Disease* have changed as well, showing that $P(Disease{=}present \,|\, Test{=}positive) = 0.6767$ and $P(Disease{=}absent \,|\, Test{=}positive) = 0.3233$. Therefore we can answer the first question posed in Example 1.1: the **positive predictive value** (PPV) of the test is 67.67%.

An alternative way to introduce a finding would be to right-click on the node and select the Add finding option of the contextual menu. A finding can be removed by double-clicking on the corresponding value or by using the contextual menu. It is also possible to introduce a new finding that replaces the old one.

1.3.2 Comparing several evidence cases

In order to compute the **negative predictive value** (NPV) of the test, click on the icon Create a new evidence case ( ) and introduce the finding $\{Test = negative\}$. The result must be similar to that of Figure 1.6. We observe that the NPV is $P(Disease{=}absent \,|\, Test{=}negative) = 0.9828$.

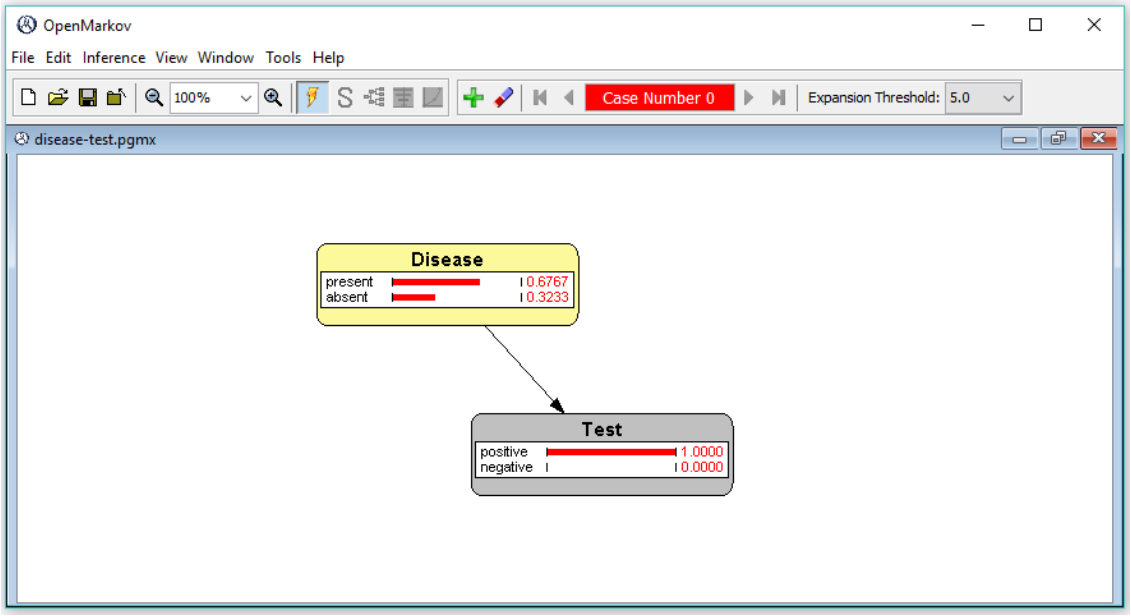


Figure 1.5: Posterior probabilities for the evidence case $\{Test = positive\}$.

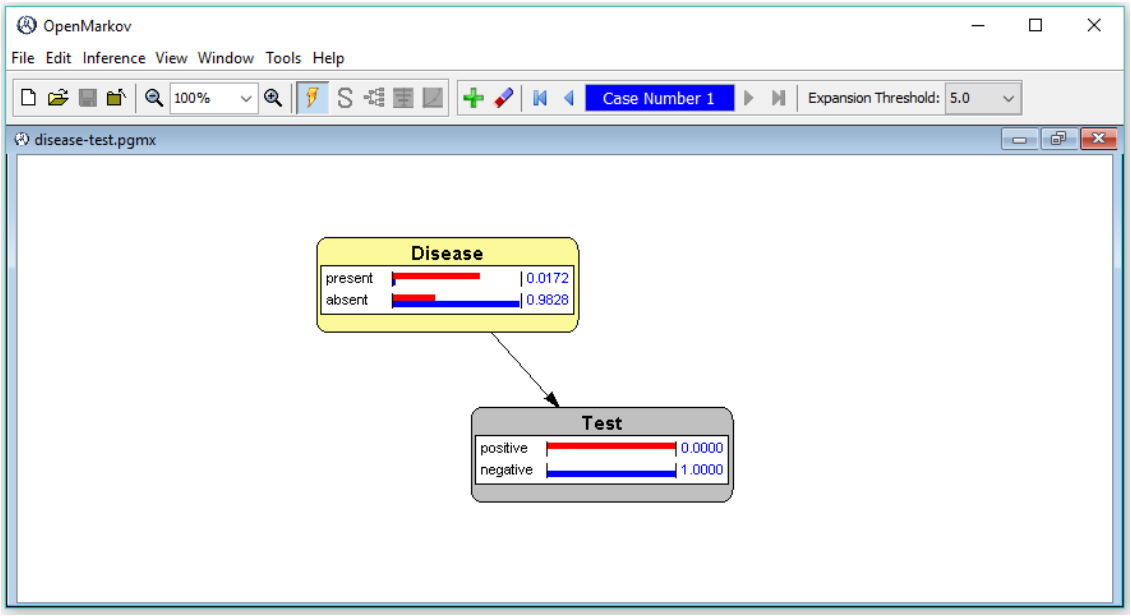


Figure 1.6: Posterior probabilities for the evidence case $\{Test = negative\}$.

In this example we have two evidence cases: $\{Test = positive\}$ and $\{Test = negative\}$. The probability bars and the numeric values for the former are displayed in red and those for the second in blue.

1.4 Canonical models

Some probabilistic models are called **canonical** because they can be used as elementary blocks for building more complex models [25]. There are several types of canonical models: *OR*, *AND*, *MAX*, *MIN*, etc. [10]. OpenMarkov is able to represent several canonical models and take profit of their properties to do the inference more efficiently. In order to see an example of a canonical

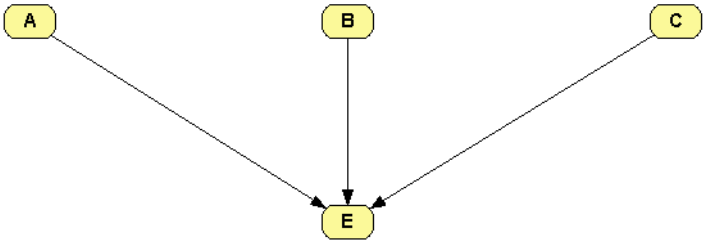
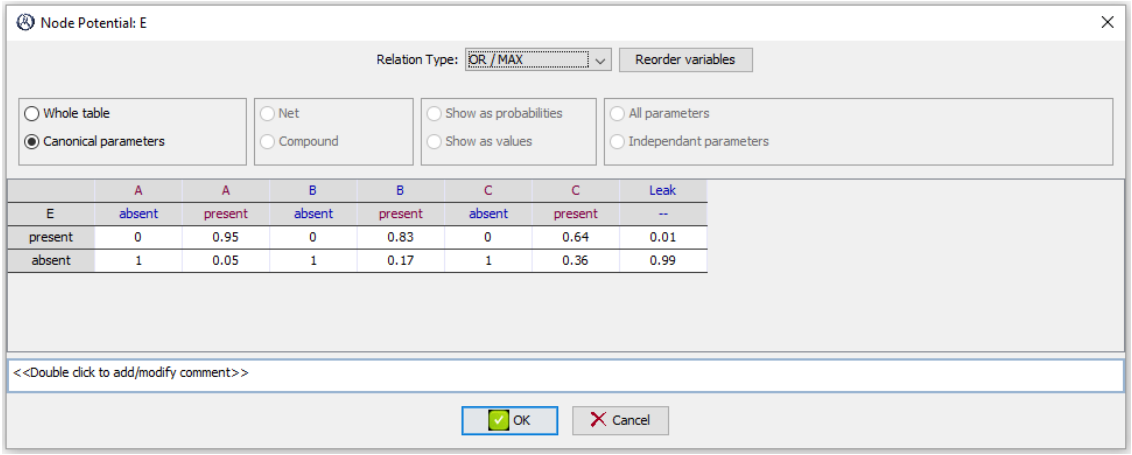


Figure 1.7: A network containing four nodes. The probability of E will be specified using a noisy OR model.



	A	A	B	B	C	C	Leak
E	absent	present	absent	present	absent	present	--
present	0	0.95	0	0.83	0	0.64	0.01
absent	1	0.05	1	0.17	1	0.36	0.99

Figure 1.8: Canonical parameters of a noisy OR model.

model, follow these steps:

1. Build the network shown in Figure 1.7.
2. Open the potential for node E, set the Relation type to *OR / MAX* and introduce the values shown in Figure 1.8. The value *0.95* means that the probability that A causes E when the other parents of E are absent is 95%. The value *0.01* means that the probability that the causes of E not explicit in the model (i.e., the causes different from A, B, and C) produce E when the explicit causes are absent is 1%.
3. Select the radio button Whole table to make OpenMarkov show the conditional probability table for this node.

Chapter 2

Learning Bayesian networks

2.1 Introduction

There are two main ways to build a Bayesian network. The first one is to do it **manually**, with the help of a domain expert, defining a set of variables that will be represented by nodes in the graph and drawing causal arcs between them, as explained in Section 1.2. The second method to build a Bayesian network is to do it **automatically**, learning the structure of the network (the directed graph) and its parameters (the conditional probabilities) from a dataset, as explained in Section 2.2.

There is a third approach, **interactive learning** [4], in which an algorithm proposes some modifications of the network, called **edits** (typically, the addition or the removal of a link), which can be accepted or rejected by the user based on their common sense, their expert knowledge or just their preferences; additionally, the user can modify the network at any moment using the graphical user interface and then resume the learning process with the edits suggested by the learning algorithm. It is also possible to use a **model network** as the departure point of any learning algorithm, or just to indicate the positions of the nodes in the network learned, or to impose some links, etc. This approach is explained in Section 2.3.

When learning any type of model, it is always wise to gain insight about the dataset by inspecting it visually. The networks used in this chapter are in the format **Comma Separated Values (CSV)**. They can be opened with a text editor, but this way it is very difficult to see the values of the variables. A better alternative is to use a spreadsheet, such as OpenOffice Calc or LibreOffice Calc. In some regional configurations, Microsoft Excel does not open these files properly because it assumes that in `.csv` files the values are separated by semicolons, because the comma is used as the decimal separator; a workaround to this problem is to open the file with a text editor, replace the commas with semicolons, save it with a different name, and open it with Microsoft Excel.

2.2 Basic learning options

2.2.1 Automatic learning

In this first example we will learn the *Asia* network [22] with the hill climbing algorithm, also known as search-and-score [15]. As a dataset we will use the file `asia10K.csv`, which contains 10,000 cases randomly generated from the Bayesian network `BN-asia.pgmx` (Figure 2.1).

1. Download onto your computer the file `www.openmarkov.org/learning/datasets/asia10K.csv`.

2. Open the dataset with a spreadsheet, as explained in Section 2.1.

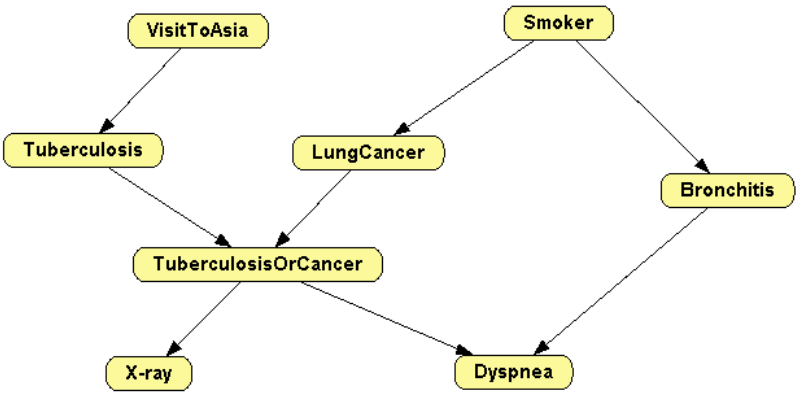


Figure 2.1: Network *Asia* proposed in [22].

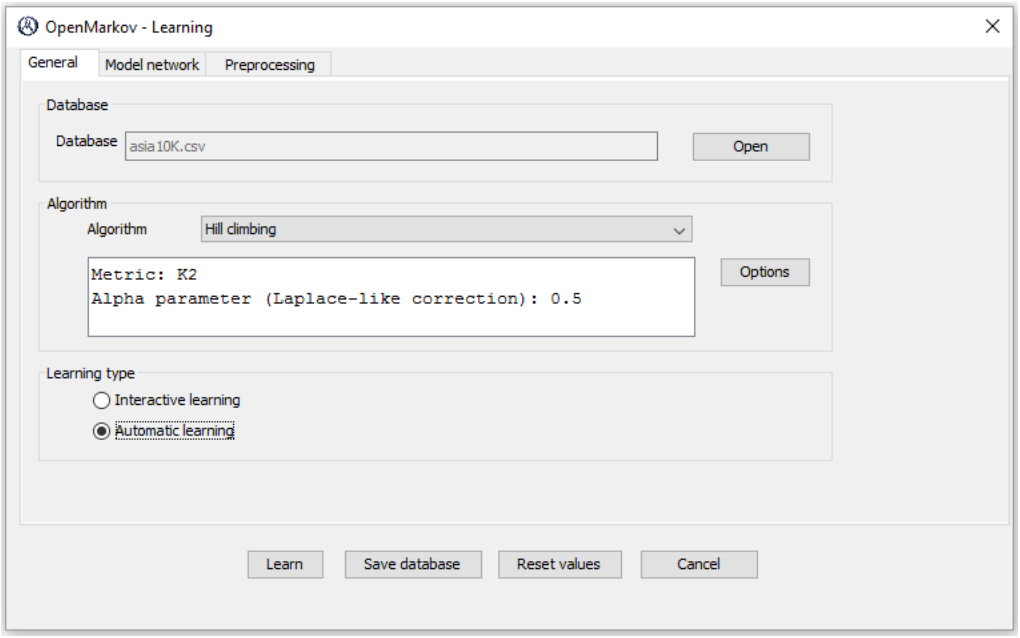


Figure 2.2: OpenMarkov's Learning dialog.

3. In OpenMarkov, select Tools ▷ Learning. This will open the Learning dialog, as shown in Figure 2.2.

4. In the Database field, select the file `asia10K.csv` you have downloaded.

5. Observe that the default learning algorithm is *Hill climbing* and the default options for this algorithm are the metric *K2* and a value of 0.5 for the α parameter, used to learn the numeric probabilities of the network (when $\alpha = 1$, we have the Laplace correction). These values can be changed with the Options button.

6. Select Automatic learning and click Learn.

The learning algorithm builds the networks and OpenMarkov arranges the nodes in the graph in several layers so that all the links point downwards—see Figure 2.3.

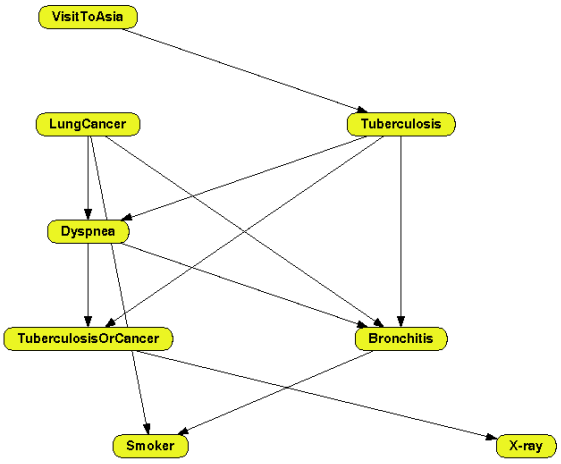


Figure 2.3: Network *Asia* learned automatically.

2.2.2 Positioning the nodes with a model network

For those who are familiar with the *Asia* network as presented in the literature [22], it would be convenient to place the nodes of the network learned in the same positions as in Figure 2.1 to see more easily which links differ from those in the original network. One possibility is to drag the nodes after the network has been learned. Another possibility is to make OpenMarkov place the nodes as in the original network; the process is as follows:

1. Download the network `www.cisiad.uned.es/ProbModelXML/examples/bn/BN-asia.pgmx`.
2. Open the dataset `asia10K.csv`, as in the previous example.
3. Select Automatic learning.
4. In the Model network tab, select Load model network from file, click Open and select your file `BN-asia.pgmx`.
5. Select Use the information of the nodes.
6. Click Learn.

The network learned has the same links as in Figure 2.3, but the nodes are in the same positions as in Figure 2.1.

The facility for positioning the nodes as in the model network is very useful even when we do **not** have a network from which the data has been generated: if we wish to learn several networks from the same dataset—for example by using different learning algorithms or different parameters—we drag the nodes of the first network learned to the positions that are more intuitive for us and then use it as a model for learning the other networks; this way all of them will have their nodes in the same positions.

2.2.3 Discretization and selection of variables

In OpenMarkov there are three options for preprocessing the dataset:

- selecting the variables to be used for learning,
- discretizing the numeric variables (or some of them), and
- treating missing values.

In this example we will illustrate the first two options; the third one is explained in Section 2.2.4.

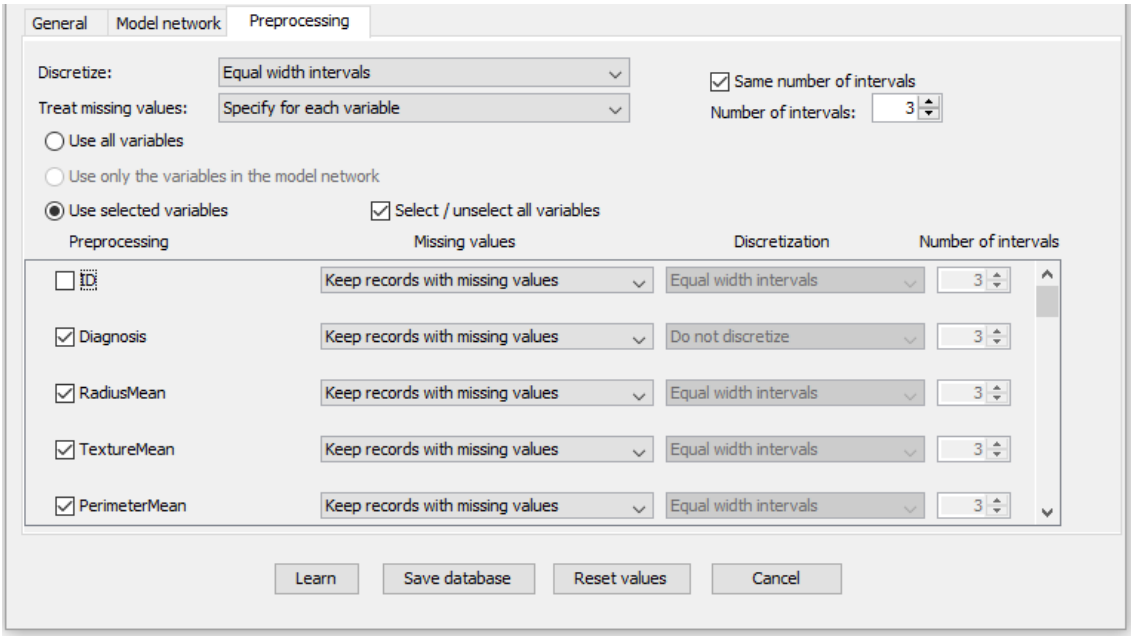


Figure 2.4: Selection and discretization of variables.

1. Download the dataset *Wisconsin Breast Cancer*, which has been borrowed from the *UCI Machine Learning Repository*.

2. Open it with a spreadsheet and observe that all the variables are numeric except *Diagnosis*.

3. Open this dataset in OpenMarkov's Learning dialog.

4. Select Automatic learning.

5. In the tab Preprocessing, select Use selected variables and and uncheck the box of the variable *ID*, as shown in Figure 2.4. (Many medical databases contain administrative variables, such as the patient ID, the date of admission to hospital, the room number, etc., which are irrelevant for diagnosis and therefore should be excluded when learning a model.) If we had specified a model network, the option Use only the variables in the model network would be enabled.

6. In the Discretize field, common to all variables, select *Equal width intervals*,[1] check the box Same number of intervals[2] and increase the Number of intervals to *3*. This means that every numeric variable will be discretized into three intervals of equal width, as we will verify after learning the network.

7. Observe that the discretization combo box for the variable *Diagnosis* in the column Discretization says *Do not discretize*—even though in the Discretize field, common to all the variables, we have chosen *Equal width intervals*—because this variable is not numeric and hence cannot be discretized.

8. Click Learn. The result is shown in Figure 2.5.

9. At the Domain tab of the Node properties dialog of the node *RadiusMean* (placed at the lower left corner in the graph), observe that this variable has three states, as shown in Figure 2.6,

[1] Another option is *Equal frequency intervals*, which can be used to create a set of intervals for each variable such that the amount of database records for that variable in every interval would be the same. The option *Do not discretize* would treat every numeric value as a different state (as if it were a string) of the corresponding variable.

[2] Leaving this option unchecked will allow you to choose a different number of intervals for each variable.

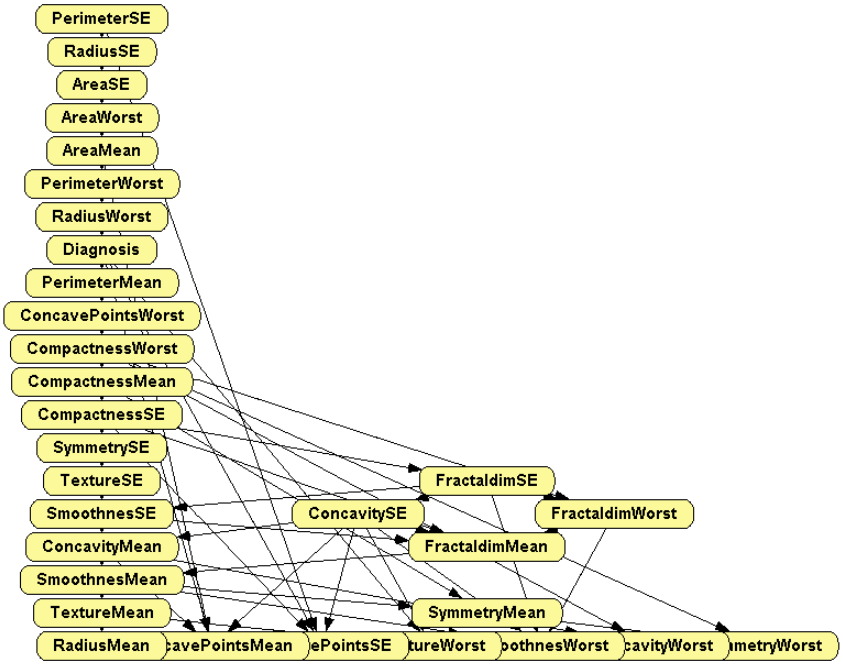


Figure 2.5: Network learned with the *Wisconsin Breast Cancer* dataset.

corresponding to three intervals of the same width; the minimum of the first interval, 6.981, is the maximum value for this variable in the dataset, and the maximum of the third interval, 28.11 is the maximum in the dataset—as you can easily check.

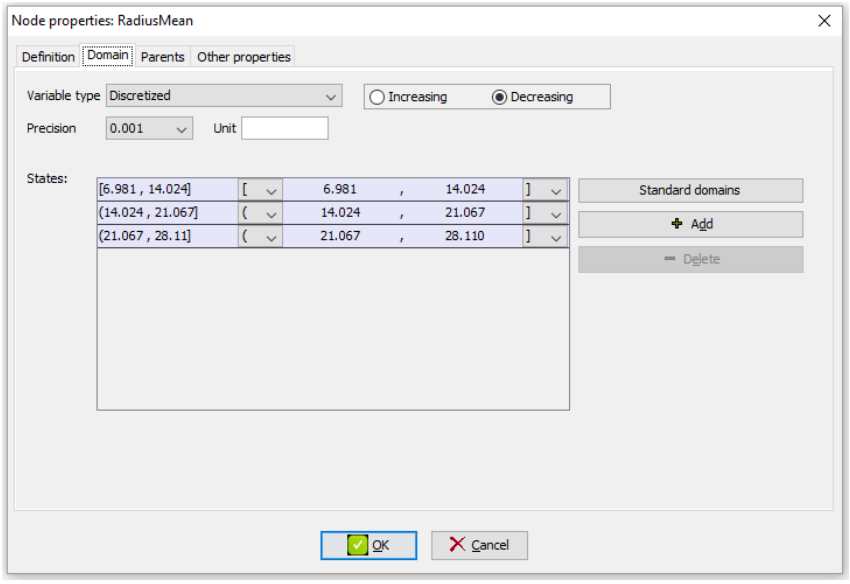


Figure 2.6: Intervals obtained for the discretized variable *RadiusMean*.

10. Repeat the previous step several times to examine the domains of the other numeric variables of the network learned.

2.2.4 Treatment of missing values

The following example explains the two rudimentary options that OpenMarkov offers currently for dealing with missing values: the option *Erase records with missing values* ignores every register that contains at least one missing value, while the option *Keep records with missing values* fills every empty cell in the dataset with the string "missing", which is then treated as if it were an ordinary value. The drawback of the first one is that it may leave too few records in the dataset, thus making the learning process completely unreliable. A drawback of the second option is that assigning the string "missing" to a numeric variable converts it into a finite-state variable, which implies that each numeric value will be treated as a different state; it is also problematic when using a model network, because in general this network does not have the state "missing" and in any case this state would have a different meaning.

In the following example we will use a combination of both options: for the two variables having many missing values (*workclass* and *occupation*) we will select the option *Keep records with missing values*, because the other option would remove too many records; for the other variables we will select the option *Erase records with missing values*, which will remove only 583 cases, namely 2% of the the records in the dataset.

1. Download the *Adult* dataset, which has also been borrowed from the *UCI Machine Learning Repository*.

2. Open it with a spreadsheet and observe that most missing values are located on the variables *workclass* and *occupation*.

3. Open it with OpenMarkov's Learning dialog.

4. Select Automatic learning.

5. Switch to the Preprocessing tab. In the Discretize field select *Equal width intervals*; check the box for Same number of intervals and increase the Number of intervals to *3*. In the list of variables, observe that the content of the Discretization column is *Equal width intervals* for the 6 numeric variables and *Do not discretize* for the 9 finite-state variables.

6. Observe that the default value of the field Treat missing values is *Specify for each variable.*

7. In the Missing values column, select *Erase records with missing values* for all variables except *workclass* and *occupation*, as mentioned above.

8. Click Learn. The result is shown in Figure 2.7.

2.3 Interactive learning

In the previous section we have explained the main options for learning Bayesian networks with OpenMarkov, which are common for automatic and interactive learning. In this section we describe the specific options for interactive learning.

2.3.1 Learning with the hill climbing algorithm

In this first example we will learn the *Asia* network with the *hill climbing* algorithm, as we did in Section 2.2.1, but in this case we will do it interactively.

1. Open the dataset `asia10K.csv`.

2. Observe that by default the Learning type is Interactive learning.

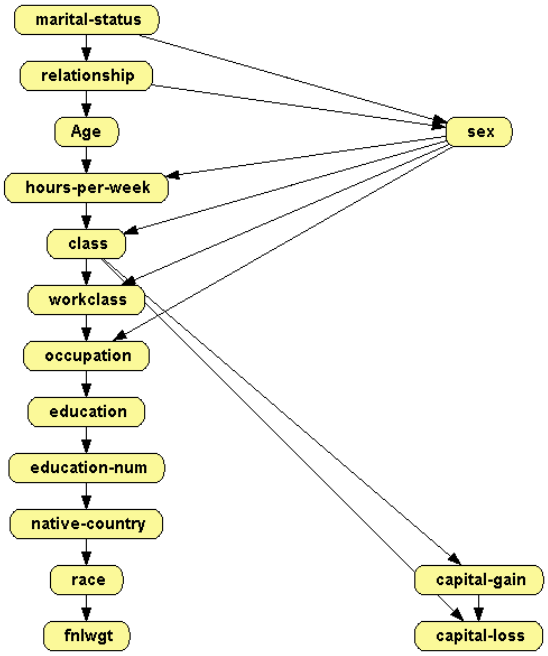


Figure 2.7: Network learned with the *Adult* dataset.

3. Click Learn. You will see a screen similar to the one in Figure 2.8. Observe that OpenMarkov has arranged the nodes in a circle to facilitate the visualization of the links during the leaning process. The Interactive learning window shows the score of each edit—let us remember that we are using the metric *K2*. Given that the hill climbing algorithm departs from an empty network, the only possible edits are the addition of links.

4. If we clicked Apply edit, OpenMarkov would execute the first edit in the list, namely the addition of the link *Dyspnea→Bronchitis*. However, bronchitis is a disease and dyspnea is a symptom. Therefore, it seems more intuitive to choose instead the second edit proposed by the learning algorithm, namely the addition of the link *Bronchitis→Dyspnea*, whose score is almost as high of that of the first. Do it by either selecting the second edit and clicking Apply edit or by double-clicking on it.

5. Observe that after adding the link *Bronchitis→Dyspnea*, OpenMarkov generates the list of possible edits for the new network and recomputes the scores.

6. Click Apply edit to select the first edit in the list, which is the addition of the link *LungCancer→TuberculosisOrCancer*. The reason for this link is that when *LungCancer* is true then *TuberculosisOrCancer* is necessarily true.

7. For the same reason, the network must contain a link *Tuberculosis→TuberculosisOrCancer*. Draw it by selecting the Insert link tool ( ) and dragging from *Tuberculosis* to *TuberculosisOrCancer* on the graph. Observe that OpenMarkov has updated again the list of edits even though this change has been made on the graph instead of on the list of edits.

8. Click Complete phase or Finish to terminate the algorithm, i.e., to iterate the search-and-score process until there remains no edit having a positive score. In this case, in which the algorithm has only one phase, the only difference between these two buttons is that Finish closes the Interactive learning window.

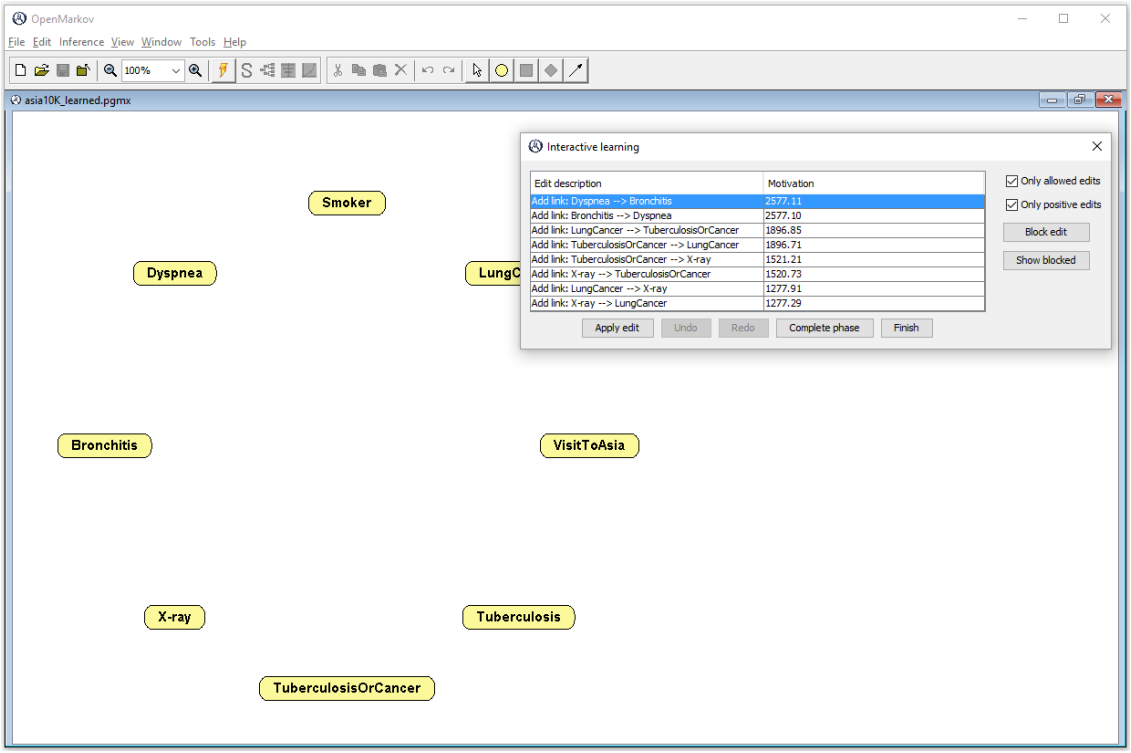


Figure 2.8: Initial state of the interactive hill climbing algorithm for the dataset *Asia*.

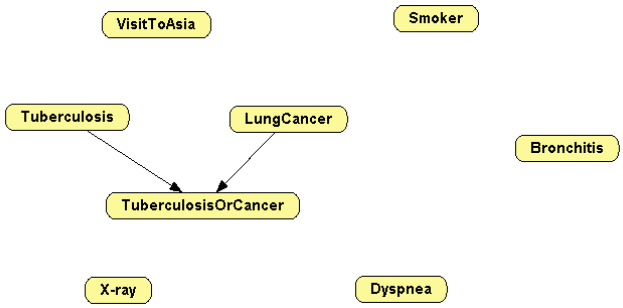


Figure 2.9: A version of the network *Asia* used to impose some links on the learning process.

2.3.2 Imposing links with a model network

There is another use of the model network that can be applied also in automatic learning, but we explain it here because its operation is more clear in interactive learning. We will also use the following example to illustrate two additional options of interactive learning: Only allowed edits and Only positive edits.

Assuming that you wish to impose that the network learned has the links *LungCancer*→ *TuberculosisOrCancer* and *Tuberculosis*→ *TuberculosisOrCancer*, whatever the content of the dataset, proceed as follows:

1. Open the Bayesian network `BN-asia.pgmx`.
2. Remove all the other links, as in Figure 2.9.
3. Open the dataset `asia10K.csv`.
4. In the Model network tab, select Use the network already opened and Start learning from model

network.

5. Uncheck the boxes Allow link removal and Allow link inversion. Leave the box Allow link addition checked.

6. Click Learn.

7. In the Interactive learning window, uncheck the box Only allowed edits. Observe that a new edit is added on top of the list: the addition of link *TuberculosisOrCancer→Tuberculosis*, which is not allowed because it would create a cycle in the Bayesian network. Check this box again.

8. In the same window, uncheck the box Only positive edits. Click Apply edit six times. Observe that some edits having negative scores are shown at the end of the list. Check that box again.

9. Click Finish.

2.3.3 Learning with the PC algorithm

Finally, we will learn a network from the `asia10K.csv` dataset using the *PC* algorithm [28] to observe how it differs from *hill climbing*. In order to understand the following example, let us remember that the PC algorithm departs from a fully-connected undirected network and removes every link that is not supported by the data. If two variables, X and Y, are independent, i.e., if the correlation between them is 0, then the link X–Y is removed. If there is some correlation, a statistical test of independence is performed to decide whether this correlation is authentic (i.e., a property of the general population from which the data was drawn) or spurious (i.e., due to the hazard, which is more likely in the case of small datasets). The null hypothesis (H_0) is that the variables are conditionally independent and, consequently, the link X–Y can be removed. The test returns a value, p, which is the likelihood of H_0.[3] When p is smaller than a certain value, called **significance level** and denoted by α, we reject H_0, i.e., we conclude that the correlation is authentic and keep the link X–Y.[4] In contrast, when p is above the threshold α we cannot discard the null hypothesis, i.e., we maintain our assumption that the variables are independent and, consequently, remove the link X–Y. The higher the p, the more confident we are when removing the link. The PC algorithm performs several tests of independence; first, it examines every pair of variables $\{X, Y\}$ with no conditioning variables, then with one conditioning variable, and so on.

The second phase of the algorithm consists in orienting some pairs of links head-to-head in accordance with the results of the test of independence, as explained below. Finally, in the third phase the rest of the links are oriented one by one.

In this example we apply the PC algorithm to the dataset `asia10K`, which we have already used several times in this chapter.

1. Open the dataset `asia10K.csv`.

2. Select the *PC* algorithm. Observe that by default the **independence test** is *Cross entropy* and the **significance level** is 0.05.

3. Click Learn. Observe that this algorithm departs from a completely connected undirected network, as shown in Figure 2.10.[5] In the first phase of the algorithm, each edit shown in the Interactive learning window consists in removing a link due to a relation of conditional independence; the motivation of each edit shows the conditioning variables (within curly brackets) and the p-value returned by the test. The list of edits shows first the relations in

[3] More precisely, p is the probability of finding the observed correlation (or a bigger correlation) conditioned on H_0 being true.

[4] Please note that in Section 2.2.1, item 5, we used the symbol α to denote a parameter used to estimate the probabilities *after* having learned the structure of the network. It has nothing to do with the significance level, also denoted by α in the literature, used by the PC algorithm.

[5] It would be possible to run the PC algorithm departing from a model network, but in this case the result would be unpredictable, as it would be difficult to find a theoretical justification for this approach.

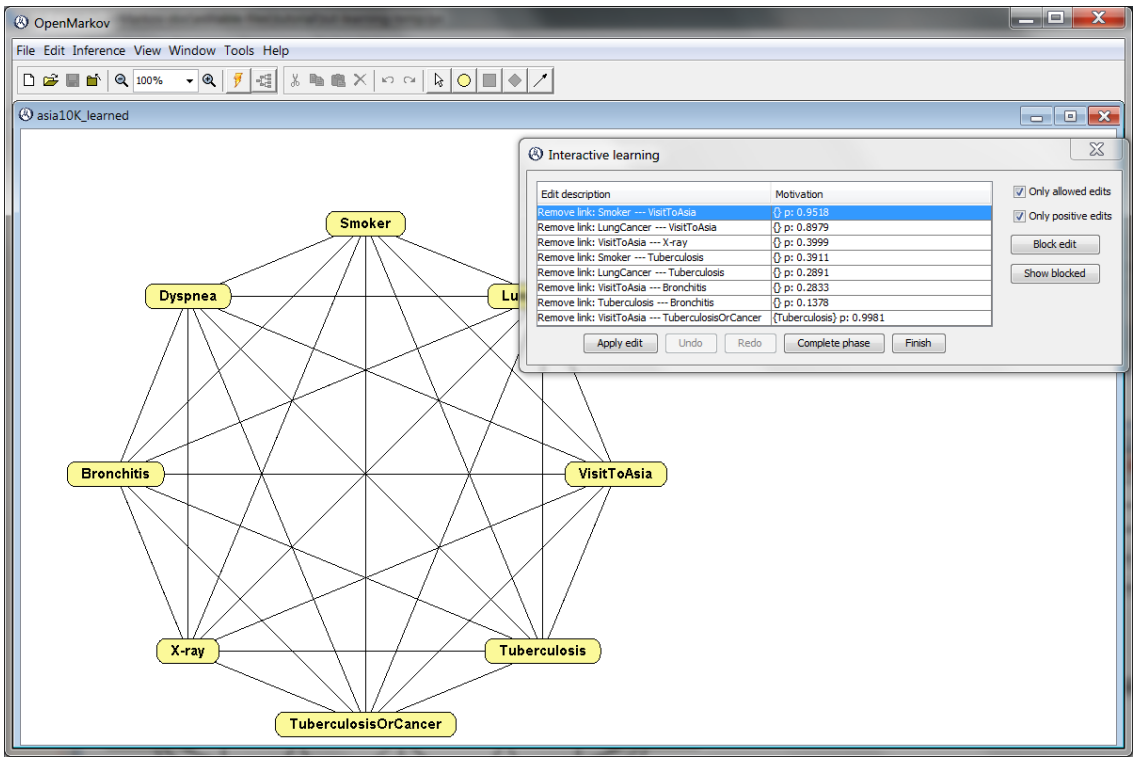


Figure 2.10: Initial state of the interactive PC algorithm for the database *Asia*.

which there is no conditioning variable; in this case, there are 7 relations. Within this group, the edits having higher p-values are shown at the top of the list, because these are the links we can remove more confidently.

4. Observe that for all these edits the p-value is above the threshold 0.05. If the checkbox *Only positive edit* were unchecked, the list would also include the edits having a p-value smaller than the threshold; in this example, there are 6 edits with $p = 0.000$. Of course, it does not make sense to apply those edits. Therefore, we should keep the box checked.

5. Click Apply edit 7 times to execute the edits corresponding to no conditioning variables.

6. Observe that now the first edit proposed is to remove the link *VisitToAsia*– *TuberculosisOrCancer* is *"{Tuberculosis} p: 0.9981"*, which means that *VisitToAsia* and *TuberculosisOrCancer* are conditionally independent given *Tuberculosis*—more precisely, that the test aimed at detecting the correlation between them, conditioned on *Tuberculosis*, has returned a p-value of 0.9981, which is also above the significance level.

7. Click Apply edit 12 times to execute the edits corresponding to one conditioning variable. Now you may click Apply edit 3 times to execute the edits corresponding to two conditioning variables. Alternatively, you may click Complete phase to execute all the remove-link edits. (If you click Apply edit when there is only one remaining edit in the current phase, OpenMarkov automatically proceeds to the next phase after executing it.)

8. The second phase of the algorithm consists in orienting some pairs of links head-to-head based on the tests of independence carried out in the first phase [25, 28]. You can execute those edits one by one or click Complete phase to proceed to third phase, in which the rest of the links will be oriented. It also possible to click Finish at any moment to make the algorithm run automatically until it completes its execution.

2.4 Exercises

As an exercise, we invite you to experiment with the *Asia* dataset using different significance levels for the PC algorithm, different metrics for the hill climbing algorithm, different model networks to impose or forbid some links, etc., and compare the resulting networks. You can also try blocking and unblocking some of the edits proposed by the algorithm (with the buttons Block edit and Show blocked), selecting other edits than those on the top of the list, adding or removing some links on the graph, etc.

You can also experiment with the *Alarm* dataset (`www.openmarkov.org/learning/datasets/alarm10K.csv`), which contains 10,000 records randomly generated from the *Alarm* network [3], a very famous example often used in the literature on learning Bayesian networks.

Finally, you can use your favorite network, generate a dataset by sampling from it (Tools▷ Generate database) and try to learn it back with different algorithms.

Chapter 3

Decision analysis networks

This chapter describes another type of probabilistic graphical model, called decision analysis network (DAN) [11, 12]. They may contain three types of nodes: chance, decision, and utility. Decision nodes represent the options that the decision maker can choose; chance nodes represent events or properties that the decision maker cannot control directly; utility nodes represent the decision maker values; for this reason they are sometimes called "value nodes". Therefore DANs can be seen as an extension of Bayesian networks, which only contain chance nodes. DANs are similar to influence diagrams, but have several advantages which will become clear in the next chapter.

Important notice: Please take into account that the evaluation of DANs does not work properly in version 0.2.0 of OpenMarkov. For this purpose you must use **version 0.1.6**.

3.1 An example with one decision

In order to explain the edition and evaluation of DANs in OpenMarkov, let us consider the following example.

Example 3.1. For the disease mentioned in Example 1.1 there are two therapies whose effectiveness is indicated in Table 3.1.[1] We assume that the test mentioned in that example is always performed before deciding about the therapy. In what cases should each therapy be applied?

	Disease = present	*Disease = absent*
no therapy	1.2	10.0
therapy 1	4.0	9.9
therapy 2	6.5	9.3

Table 3.1: Effectiveness of the therapies.

We observe in Table 3.1 that the best scenario occurs the disease is absent and no therapy is applied (effectiveness = 10). In the worst scenario, the patient is suffering from the disease but does not receive any therapy (effectiveness = 1.2). If a sick person receives the first therapy the effectiveness increases to 4.0, while it increases to 6.5 with the second therapy. If a healthy person receives the first therapy (by mistake) the effectiveness decreases to 9.9 as a consequence of the side effects. The second therapy, which is more aggressive, makes the effectiveness decrease to 9.3 for healthy people. The problem is that we do not know with certainty whether the disease is present or not, we only know the result of the test, which sometimes gives false positives and false negatives. For this reason we will perform a decision analysis to find out the best intervention in each case.

[1] At this moment we do not care about the scale used to measure effectiveness. We will discuss it in Chapter 5when studying multicriteria decision making.

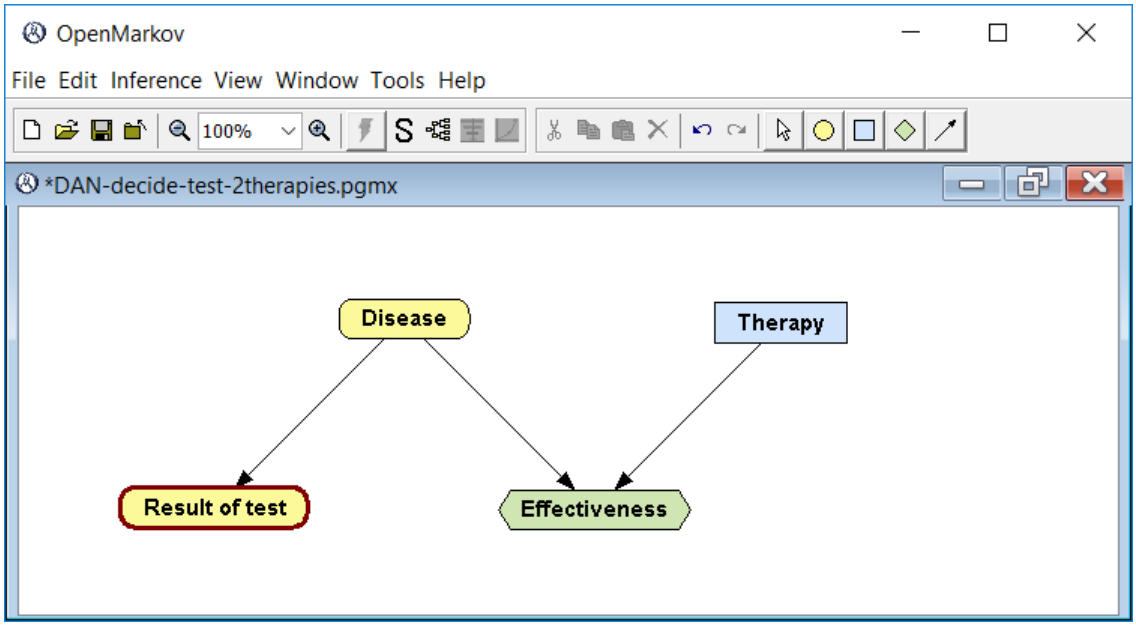


Figure 3.1: A DAN for deciding about the therapy (Example 3.1).

3.1.1 Editing the decision analysis network

The edition of DANs is similar to that of Bayesian networks. The main difference is that now the icons for inserting decision nodes and utility nodes are enabled, as shown in Figure 3.1. It would be possible to build a DAN for our problem from scratch, but in this case we will depart from the Bayesian network for the Example 1.1.

1. Open the network `BN-disease-test.pgmx`. Open its contextual menu by right-clicking on the background, click Network properties, and in the Network type field select *Decision analysis network*. Observe that the icons for inserting decision nodes (▭) and utility nodes (◇) are now active.

2. Click on the Insert decision node icon and double-click on the place where you wish to insert the node *Therapy*. In the Name field, type *Therapy*. Click on the Domain tab and set the values (states) to *no therapy*, *therapy 1*, and *therapy 2*; OpenMarkov's default values for decision nodes are *yes* and *no*. Modify these states by double-clicking on their names and add a new state using the Add button on the right side of the Node properties dialog.[2]

3. Click on the Insert utility node icon, double-click on the place where you wish to insert the node, and in the Name field, type *Effectiveness*.

4. Draw the links *Disease*→*Effectiveness* and *Therapy*→*Effectiveness* because, according with Table 3.1, the effectiveness depends on *Disease* and *Therapy*. The result should be similar to that in Figure 3.1.

5. Open the potential for the node *Effectiveness* by selecting Edit utility in the contextual menu or by alt-clicking on the node. In the Relation type field select *Table*. Introduce the values given in Table 3.1, as shown in Figure 3.2.

6. Open the Node properties dialog for *Test* and check the box Always observed to indicate that the result of the test is known before making the decision.

[2]In the current version of OpenMarkov, when editing the name of a state it is necessary to press Tab or click on another state to make the changes effective. If you click any button while editing the name, the change will be lost. We will correct it in future versions.

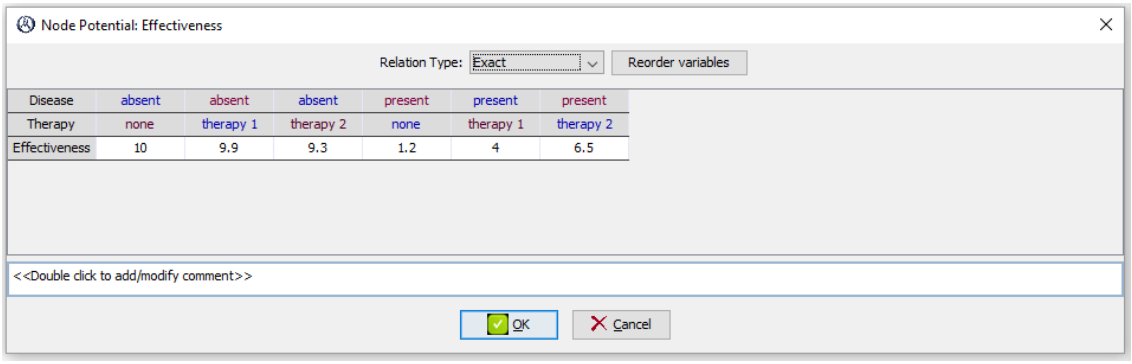


Figure 3.2: Utility table for the node *Effectiveness*.

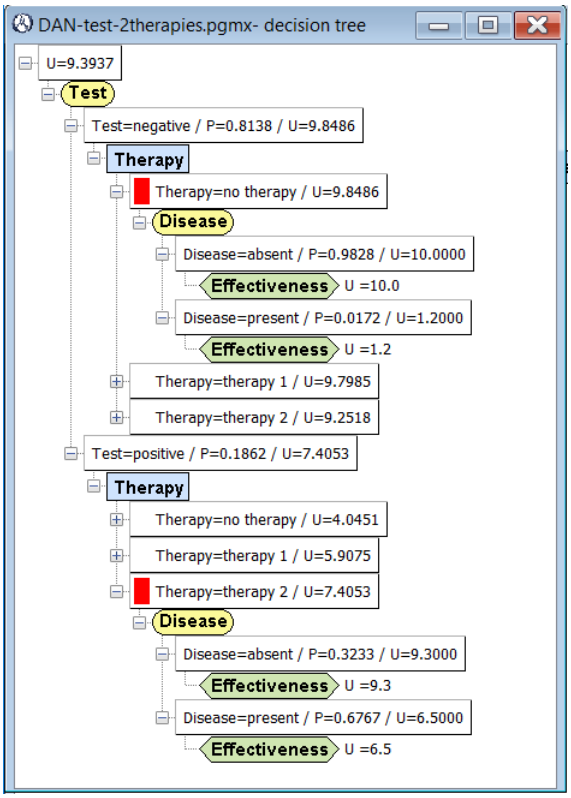


Figure 3.3: Decision tree for Example 3.1. It results from the DAN in Figure 3.1.

Please remember that the nodes *Disease* and *Test* have the potentials we assigned to them when building the Bayesian network, namely, a conditional probability table for each node. The utility node has the potential we have just assigned to it. The node *Therapy* does not need any potential because it is a decision. Therefore, the DAN is complete. Save it as `DAN-test-2therapies.pgmx`.

3.1.2 Equivalent decision tree

Given a DAN, it is possible to expand an equivalent decision tree, as follows:

1. Click on the Decision tree icon ( ) in the first toolbar.

2. Observe that:

(a) Every branch outgoing from a chance node displays its probability, and the probabilities of all the branches going out from each chance node sum 1.

(b) The probability of a positive result of the test is 0.1862, the same as in Figure 1.4. When the test is positive, the probability of having the disease is 0.6767, the same as in Figure 1.5; when it is negative, the probability is 0.0172, the same as in Figure 1.6.

(c) The leaves of the tree represent the utilities given in the statement of the problem (Table 3.1).

(d) The root of the tree shows the expected utility for this model.

(e) The utilities of inner nodes are the result of applying the roll-back algorithm [27, 26].

(f) A red box inside a node denotes the optimal choice for a decision in a scenario. Thus, when the test has given a negative result, the optimal decision is not to apply any therapy; when the result is positive, the optimal decision is to apply the second therapy. We will later see that the optimal policy depends on the numerical parameters of the model.

3. Collapse the suboptimal branches—i.e., those that go out of a decision node and do not have a red mark—by click the "−" widget. Check that you obtain the same tree as in Figure 3.3.

3.1.3 Optimal strategy

The optimal strategy is display as a tree indicating which option to choose for each decision in each scenario. If you click on the Optimal strategy icon (S) in the first toolbar, you will obtain the tree shown in Figure 3.4, which means that the optimal strategy is to apply the second therapy when the test is positive and no therapy when it is negative. A strategy tree is similar to a decision tree, but there are also some differences:

1. Strategy trees have three types of nodes: chance, decision, and *nil*, but *nil* nodes drawn.

2. The white boxes that you observe in Figure 3.4 are the labels of the branches; each branch corresponds to one or more states of the variable from which the branch goes out.

3. Several chance nodes in the tree may correspond to the same node in the DAN; for example, the nodes *Therapy*.

4. The parent of a *nil* node is always a decision node, which implies that the last node shown in each branch is of type decision.

5. Each decision node has exactly one child; the label of its outgoing branch denotes the optimal choice for that scenario.

6. The ancestors of a decision node in the tree (i.e., the nodes at its right) correspond to the variables whose values are known when making the decision. The variables whose value is never observed (for instance, *Disease*) do not appear in the strategy tree.

3.1.4 Order of the variables in the decision tree and the strategy tree

If a node appears between the root and a decision in the decision tree it means that the value of that node is known before making the decision. For example, *Test*, which is at the right of the decision *Therapy* in Figure 3.3 because we indicated that *Test* is Always observed. If a node appears between a decision and the leaves, its value is not know when making the decision. For example, *Disease* appears at the right of *Therapy* because its value is never observed. The order in the strategy tree is the same, with the only difference that a variable that is never observed does not appear in the tree, as mentioned above.

You can uncheck the box Always observed for the node *Test* in the DAN and see that the order is that now the order of the variables in the decision tree is not *Test–Therapy–Disease* but

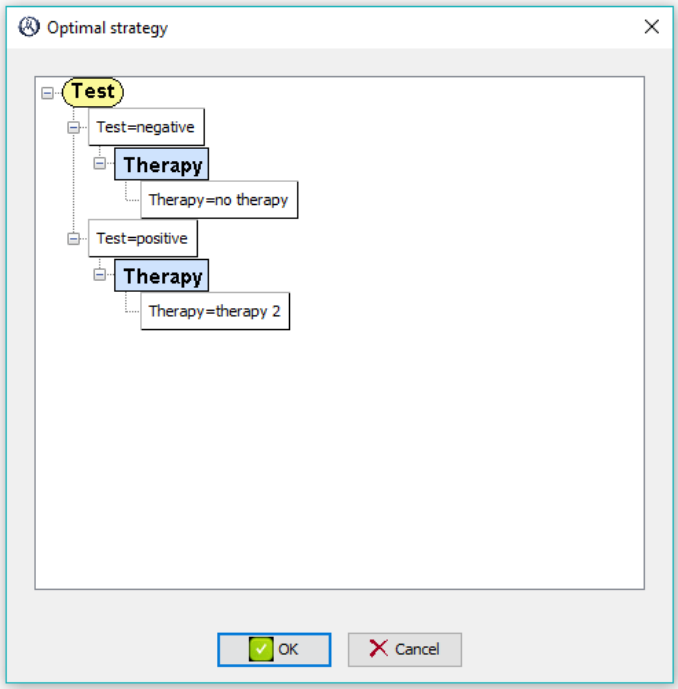


Figure 3.4: Optimal strategy for Example 3.1. It also results from the DAN in Figure 3.1. Compare it with the decision tree in Figure 3.3.

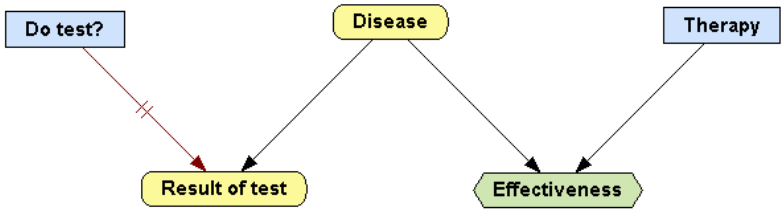


Figure 3.5: A DAN with two decisions (Example 3.2).

Therapy–Disease–Test, which means that the values taken by these variables are not known when making the decision. Consequently, these variables do not appear in the strategy tree.

Now check the box Always observed for the node *Disease* and see that the order is *Therapy–Disease–Test* in the decision tree and *Therapy–Disease–Test* in the strategy tree.

If we mark both *Disease* and *Test* as Always observed, both appear at the left of *Therapy* in the decision tree, but *Test* does not appear in the optimal strategy because once we know with certainty whether the patient has the disease or not, the result of the test is irrelevant.

3.2 An example with two decisions

In this section we introduce an example with two decisions.

Example 3.2. In the context of the previous example, is it worth doing the test?

Please note that, given that we have not assigned any economic cost nor any side effect to the test, it will never cause any harm doing it. The issue is, therefore, whether it will bring any profit. We can answer this question by introducing another decision in the model, as follows—see Figure 3.5.

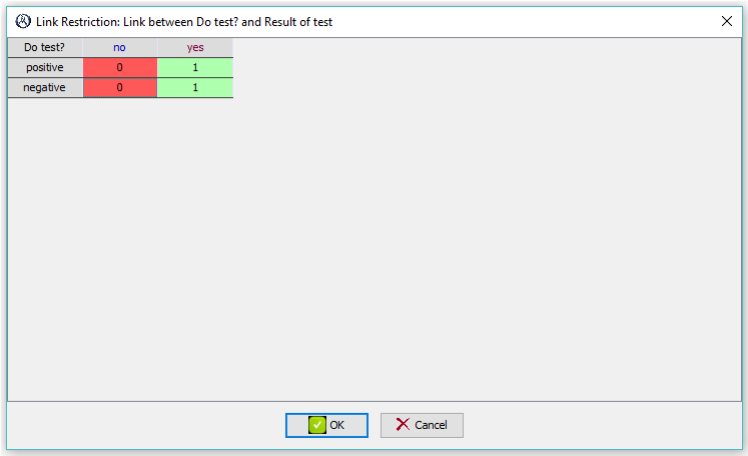


Figure 3.6: Restrictions for the link *Do test E?* → *Result of E*. They mean that when the test is not done it can give neither a positive nor a negative result.

3.2.1 Editing the model: restrictions and revelation conditions

1. Open the network `DAN-test-2therapies.pgmx`.
2. Add the decision node *Do test E?* (we call it "*E*" because in the next example we will introduce another test). Keep the default states, *yes* and *no*.
3. Rename the node *Test* as *Result of E*, and uncheck the box Always observed.
4. Draw the link *Do test E?*→*Result of E*.
5. If *Do test E?* = *no*, the result of the test can be neither *positive* nor *negative*. We encode this information by right-clicking on the link *Do test E?*→*Result of E* and selecting Add restrictions. Click on the cells of the first column; their value will change from 1 to 0 and their color from green to red, as shown in the Figure 3.6.This means that *Do test* = *no* is incompatible with both *Result of E* = *yes* and *Result of E* = *yes*. Press OK.
6. Observe the short perpendicular double line on that link: it denotes a **total restriction**, i.e., it means that at least one of the values of the parent variable, *Do test*, is incompatible with *all* the values of the child variable, *Result of E*. (There would be a **partial restriction** if at least one of the states of the parent were incompatible with some states of the child but not with all of them. It would be denoted by a short perpendicular single line on the link.)
7. Open the probability table for *Result of E* and see that the cells corresponding to the above restrictions are colored in red and its probability is 0. You may reorder the parents of this variable, as in Figure 3.7, by clicking the Reorder variables button.
8. If the test is performed (*Do test E?* = *yes*), the *Result of E* is known. In order to establish this **revelation condition**, right-click on link *Do test E?*→*Result of E*, select Edit revelation conditions and check the box for yes. The link will be colored in dark red to show that it is a **revelation link**.
9. Save this DAN as `DAN-decide-test-2therapies.pgmx`.

3.2.2 Evaluating the model

1. Observe that the **optimal strategy**, shown in Figure 3.8, is to do the test and apply the second therapy if and only if it gives a positive result.

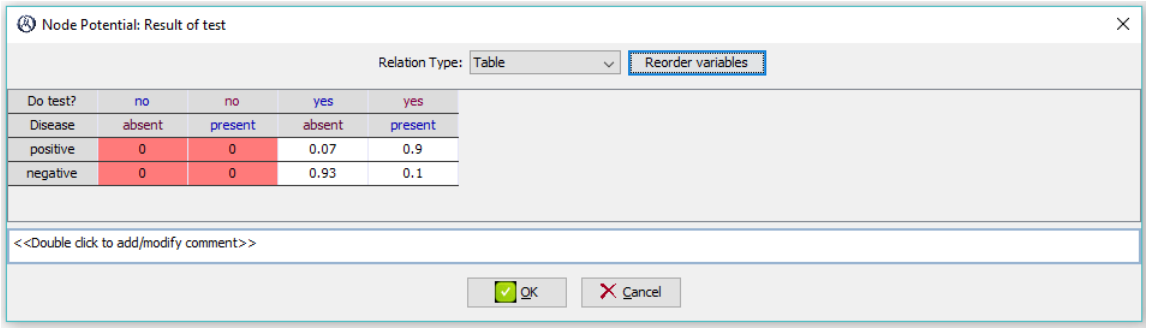


Figure 3.7: Conditional probability table for *Result of E*. Some cells are colored in red because of the restrictions shown in Figure 3.6 link restriction.

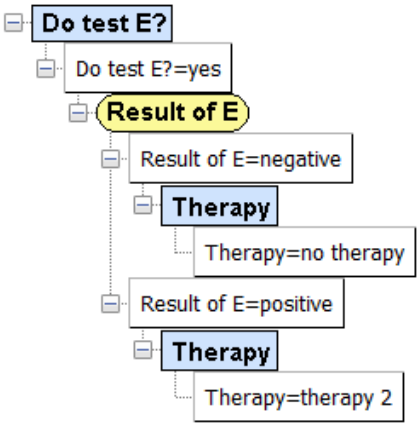


Figure 3.8: Strategy tree for Example 3.2. It results from the DAN in Figure 3.5.

2. In the **decision tree** you can check that the expected utility of doing the test, 9.3937, is higher than that of not doing it, 9.0740—see Figure 3.9—and the optimal branches (those marked with red rectangles) are the same as in the strategy tree.

3.3 A model with three partially-ordered decisions

3.3.1 Edition of the DAN and optimal strategy

Example 3.3. In the context of the previous example, let us consider a second test, C, with a sensitivity of 78% and a specificity of 91%; the discomfort it causes to the patient has been estimated in 0.001 units of effectiveness.

This problem can be modeled with a DAN as follows:

1. Open the network `DAN-decide-test-2therapies.pgmx`.
2. Add the decision node *Do test C?*. Keep the default states, *yes* and *no*.
3. Add the decision node *Result of C*. Click the Standard domains button in the Domain tab and select *negative–positive*.
4. Draw a link from *Disease* to *Result of C*.
5. Draw a link from *Do test C?* to *Result of C*. Using the contextual menu for this link, add the revelation condition for *Do test C?* = *yes* and the restrictions that indicate that *Do test*

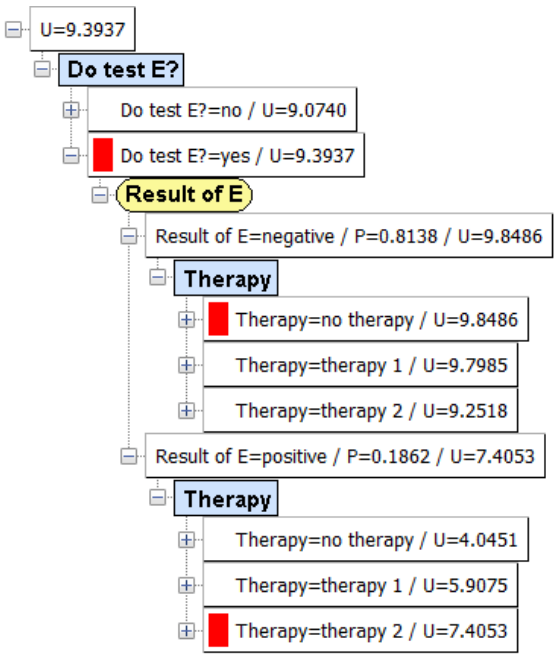


Figure 3.9: Decision tree for Example 3.2. It also results from the DAN in Figure 3.5. Compare it with the strategy tree in Figure 3.8.

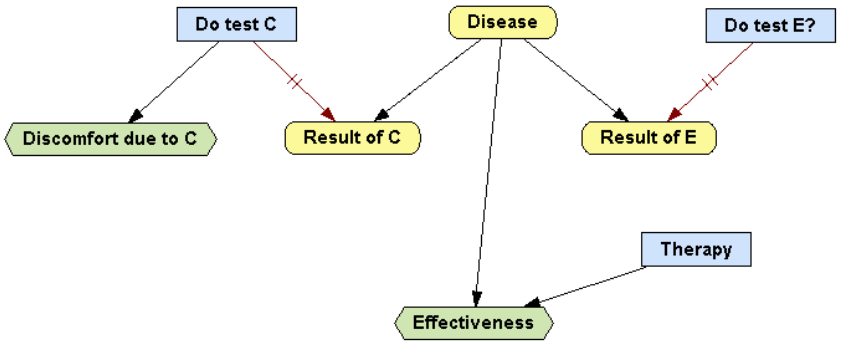


Figure 3.10: A DAN with three decisions (Example 3.3).

$C? = no$ is incompatible with the two states of *Result of C*. The graph will be similar to that in Figure 3.10.

6. Edit the probability table for *Result of C* and enter the sensitivity and specificity of this test. It will look like the table in Figure 3.7.

7. Save this DAN as `DAN-decide-2test-2therapies.pgmx`.

Note that in this DAN there is one decision for each test because they are compatible, i.e., it is possible to do both of them, only test E, only test C, or none. In contrast, there is only one decision node for the two therapies because we have assumed that they are mutually exclusive, i.e., it is not possible to apply both of them to the same patient.

Observe that, according to the DAN, the optimal strategy for this problem (Fig.) is: do test E; if it is positive, apply the second therapy; if it is negative, do the first test and if this is positive apply the first therapy.

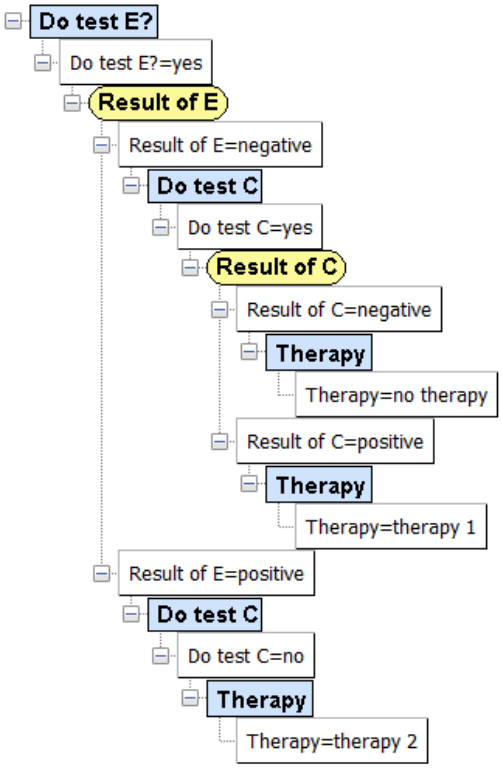


Figure 3.11: A strategy tree for Example 3.3. It results from the DAN in Figure 3.10.

3.3.2 Order of the decisions in the decision tree

Near the top of the decision tree (cf. Figure 3.12) there is a decision node, *OD*, which was not in the DAN. Why? The reason is that the DAN contains two decisions about the tests, and it is not clear which one should be made first. The node *OD*, whose name stands for "order of the decisions", reflects the need to evaluate the two possibilities: the branch $OD = Do\ test\ E?$ corresponds to deciding first about E, and $OD = Do\ test\ C?$ to deciding first about C. For this reason *OD* is called a **meta-decision**. We observe that the utility of the first branch, 9.4162, is slightly higher than that of the second, 9.4160, which implies that we must decide first about E.

The reader may wonder: if there are three decisions in the DAN and any of them might be the first one, why is there not a branch $OD = Therapy$ in the decision tree? In fact, the algorithm that evaluates the DAN did consider that possibility, but it realized that the decision *Therapy* in no case contributes any information (there is no revelation link outgoing from that node) and therefore it is never beneficial to make it before the decisions that may reveal some information, such as $Do\ test\ E?$ and $Do\ test\ C?$ [12].

For the same reason, the decision tree in Figure 3.9 might have a meta-decision node *OD* with two branches, $OD = Do\ test\ E?$ and $OT = Therapy$, but the utility of the second could never be higher than that of the first, and when it is clear which decision to make first, the meta-decision is unnecessary.

3.3.3 Advantages of DANs

As we have seen in Section 3.3.1, with a little practice in the use of OpenMarkov, it only takes around one minute to add a new test to the DAN and evaluate the new model. With one more click we get the decision tree. In contrast, had we built the decision tree for Example 3.1 (Fig. 3.3) directly with a software package for decision trees—instead of building the DAN—it would have taken a while to add the decision about the test E (Example 3.2, Fig. 3.9), and much more time to add the decision about the second test, C (Example 3.3, Fig. 3.3), because the decision tree

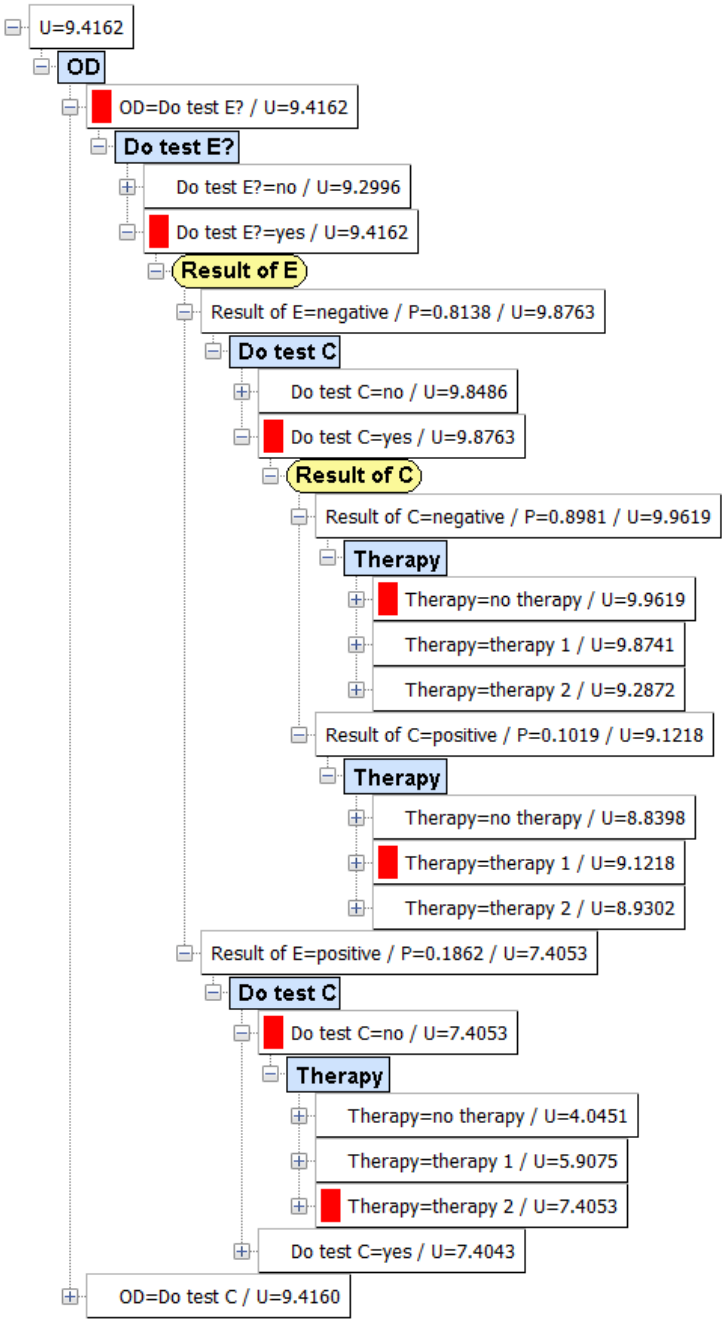


Figure 3.12: Decision tree for Example 3.3. It results from the DAN in Figure 3.10. The strategy implicit in this decision tree is the same as that in Figure 3.11.

is really big—in Fig. 3.9 many of its branches are collapse; you can appreciate its real size by expanding it again from the DAN in Fig. 3.10. Please note that the size of a DAN grows linearly with the number of variables, while the size of a tree grows exponentially. Additionally, adding new variables in a decision tree does not only require to modify its structure, but also to recompute the probabilities of all its branches, which cannot be accomplished by just cutting and pasting sub-trees.

Another advantage with respect to decision trees is that the graph of a DAN compactly represents the structure of the model and some of its structural hypotheses. For example, in Figure 3.3 we can see that there are two tests and that they assumed to be conditionally independent given

the disease but, in contrast, we are not imposing any order between them.

Furthermore, we should remark that these examples are unicriterion problems with no uncertainty in the parameters. Doing cost-effectiveness analysis and sensitivity analysis with other software tools is much easier with DANs than with decision trees, as we will show in the next chapters.

Finally, we should mention that influence diagrams also have the advantages with respect to decision trees that we have discussed in this section. However, we will see in the next chapter that the problem posed in Example 3.3 cannot be solved—at least, not easily—with influence diagrams.

Chapter 4

Influence diagrams

Influence diagram (IDs) are very similar to DANs. In fact, for every ID there is an equivalent DAN; the transformation can be done with a straightforward algorithm. However, for most DANs there is no equivalent ID. There are some problems that can be represented and solved with both types of models but for in general for each problem a DAN is easier to build than an ID. For these reasons we claim that DANs can favorably replace IDs as a decision analysis tool. However, in this chapter we will present IDs for three main reasons.

First, all decision analysts know the ID formalism and many of them have used it to solve real-world problems. In contrast, DANs are almost unknown because they have been proposed recently and no journal paper about them has been published yet.[1] If we did not describe IDs in this tutorial, those analysts would miss them. For this purpose, we will use the same examples as in the previous chapter.

Second, we wish to prove the above assertions that building DANs is easier than building IDs (at least, it is never more difficult) and that some problems solvable with DANs cannot be solved with IDs.

Third, the implementation of IDs in OpenMarkov, which started several years before the creation of DANs, offer several facilities that are not yet available for DANs, such as the explanation of reasoning, sensitivity analysis, cost-effectiveness analysis, and Markov models.

4.1 An example with one decision

4.1.1 Editing the ID

Building an ID for Example 3.1 is very similar to the construction of the equivalent DAN:

1. Open the network `BN-disease-test.pgmx`. Open its contextual menu by right-clicking on the background, click Network properties, and in the Network type field select *Influence diagram*. Observe that, again, the icons for inserting decision nodes (□) and utility nodes (◇) are now active.

2. Insert the decision node *Therapy*, with states *no therapy*, *therapy 1*, and *therapy 2*, and the utility node *Effectiveness* node, just as for the DAN. Draw the links *Disease→Effectiveness* and *Therapy→Effectiveness* and enter the effectiveness values, as in Figure 3.1.

[1] IDs were developed in the 1970s at the Stanford Research Institute. Near the end of that decade, Howard and Matheson wrote a paper describing them. Being unable to get it accepted at any journal, they decided to publish it in a book in 1984 [16]. Paradoxically, a paper widely cited in decision analysis, artificial intelligence, economics, control theory, and several other fields was not available in any periodicals library. Fortunately, in 2005 it was included in a special double issue of the *Decision Analysis* journal [17], together with several retrospective papers about IDs.

So far, the history of DANs is not very different. The paper describing them was rejected at three conferences before being accepted at a workshop [11] and three relevant AI journals have already rejected it. We hope it will not need as long as the ID paper to appear in a journal.

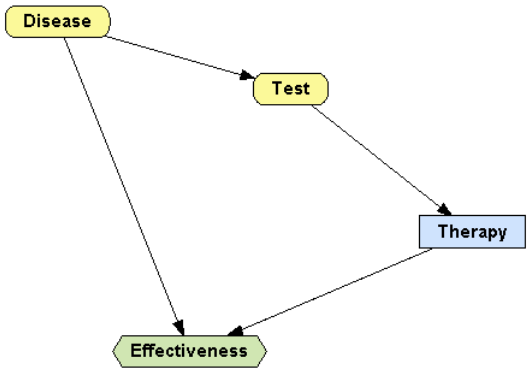


Figure 4.1: ID for Example 3.1. Compare it with the DAN in Figure 3.1.

3. The main difference with the DANs lies in the way to indicate that the result of the test is known before making the decision. Now, instead of ticking the Always observed checkbox, which is not available for IDs, we draw a link from *Result of E* to *Therapy*. A link like this, from a chance node to a decision, is called an **information link**.

4. You may rearrange the nodes to make the graph clearer, as shown in Figure 4.1.

5. Save the network as `ID-test-2therapies.pgmx`.

6. Check that the decision tree and the strategy tree are the same as for the DAN (Figs. 3.3 and 3.4).

4.1.2 Resolution, optimal strategy and maximum expected utility

We have seen that one way of evaluating a DAN or an ID consists in generating an equivalent decision tree or a strategy tree. However, IDs are more restrictive than DANs, and this, which is a disadvantage from the point of view of expressive power, makes it possible to perform several types of inference which are not possible for general DANs.[2] Let us now introduce a few concepts to illustrate it.

The values that are known making a decision are said to be its **informational predecessors**. In a DAN the informational predecessors of a decision are not always the same. Thus, for the DAN in Figure 3.5, when the decision *Do test E?* is made before the decision *Therapy* and the option chosen is to do the test, both *Do test E?* and *Result of E* are informational predecessors of *Therapy*; but if the option chosen is not to do the test, *Result of test* is not an informational predecessor of *Therapy*. In contrast, in an ID the informational predecessors of a decision are the same in all the scenarios; it cannot occur that in some cases one variable is observed when making the decision and in other cases it is not.

This property of IDs allows us to define a **policy** for a decision as a family of probability distributions such that there is one distribution for each configuration of its informational predecessors. For example, the policy shown in Figure 4.2 contains one probability distribution for *Test* = *negative* (first column) such that in this scenario the option *no therapy* will always by chosen (probability = 1.0 = 100%). The probability distribution for *Test* = *positive* (second column) states when the test gives a positive result the option *therapy 2* will always be chosen. The reason why some cells in Figure 4.2 are colored in red in will be explained below. A policy in which every probability is either 0 or 1, as in this example, is called a **deterministic policy**. It is also possible to have purely **probabilistic policies**, i.e., those in which some of the probabilities are different from 0 and 1—we will see an example in Figure 4.11.

[2]There is very restricted subset of DANs, satisfying certain conditions of symmetry, such that for every ID there is a symmetric DAN, and vice versa [12]. In future versions of OpenMarkov all the inference options that we

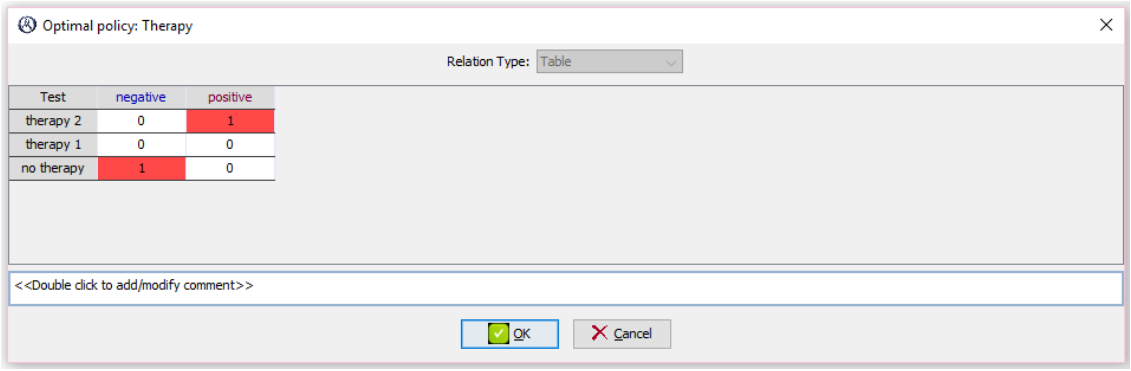


Figure 4.2: Optimal policy for the decision *Therapy*.

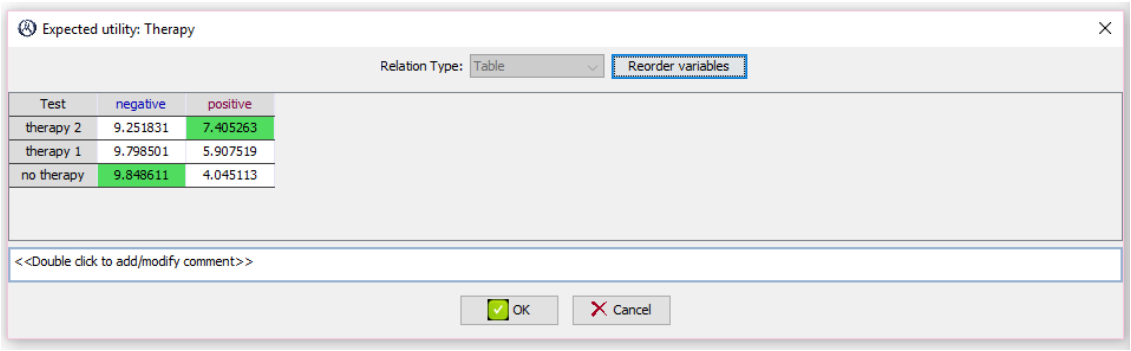


Figure 4.3: Expected utility for the decision *Therapy*.

In this context, a **strategy** is defined as a set of policies, one for each decision in the ID. Each strategy has a **expected utility**, which depends on the probabilities and utilities that define the ID and on the policies that constitute the strategy. A strategy that maximizes the expected utility is said to be **optimal**. A policy is said to be **optimal** if it makes part of an optimal strategy.

The **resolution** of an ID consists in finding an optimal strategy and its expected utility, which is the **maximum expected utility**.

In order to evaluate this ID, click the Inference button ( ) to switch from edit mode to inference mode. OpenMarkov performs two operations: the first one, called **resolution**, consists in finding the optimal strategy, as explained above. The policies that constitute the optimal strategy can be inspected at the corresponding decision nodes. For example, if you select Show optimal policy in the contextual menu of the node *Therapy*, this will open the window shown in Figure 4.2. For each column (i.e., for each configuration of the informational predecessors of this node), the cell corresponding to the optimal option is colored in red. The contextual menu of decision nodes also contains the option Show expected utility. The expected utility of *Therapy* can be observed in Figure 4.3.

The second operation performed by OpenMarkov when switching to inference mode, called **propagation**, consists in computing the posterior probability of each chance and decision node and the expected utility of each utility node. The result is shown in Figure 4.4. Please note that the probability distribution for *Test* is the same as in Figure 1.4. Given that this ID contains only one utility node, the value 9.3937 shown therein is the global maximum expected utility of the ID, which coincides with the one at the root node of the decision tree in Figure 3.3.

describe in this section will also be available for symmetric DANs, and some of them even for asymmetric DANs.

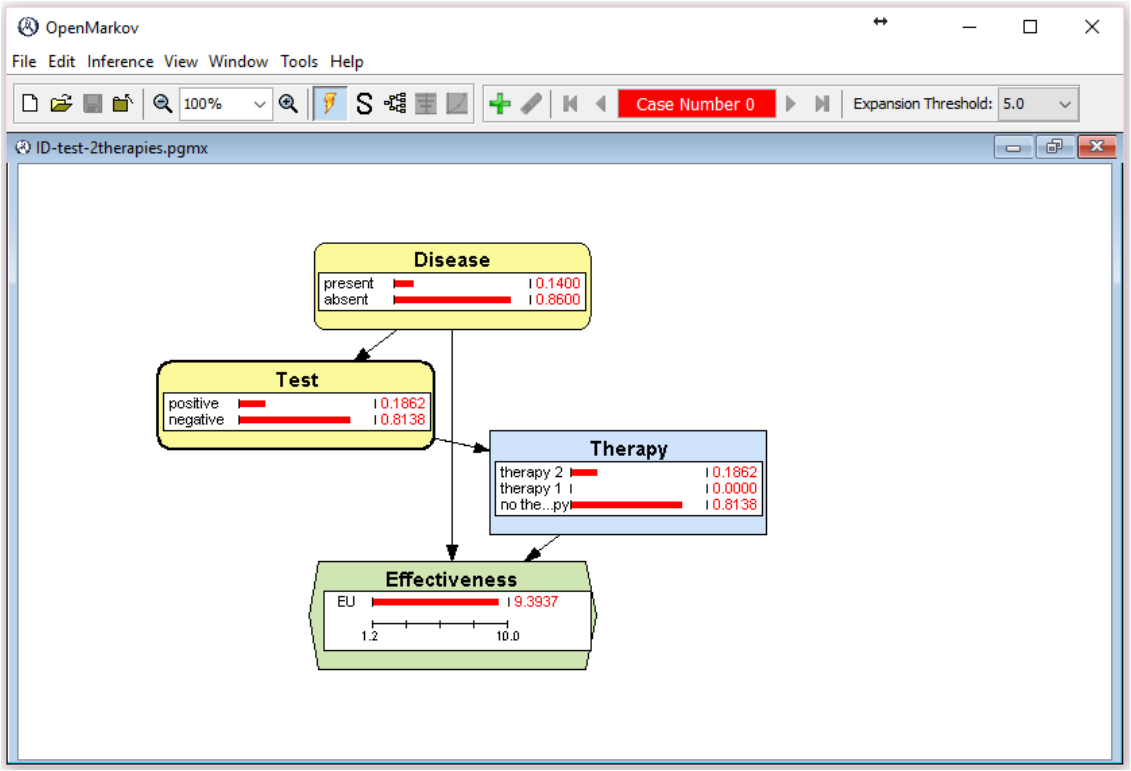


Figure 4.4: Evaluation of the ID in Figure 4.1.

4.2 An example with two decisions

We will now solve with an ID the problem stated in Example 3.2.

4.2.1 Editing the influence diagram

The steps necessary to adapt the previous ID are the following:

1. Open the network `ID-test-2therapies.pgmx`.
2. Rename the node *Test* as *Result of E*.
3. Create the decision node *Do test E?*. Keep the default states, *yes* and *no*.
4. Draw the link *Do test E?*→*Result of E*. The graph should look like the one in Figure 4.5.
5. If *Do test E?* = *no*, the result of the test can be neither *positive* nor *negative* but, unlike the case of DANs, the variable *Result of E* must take a value even when the test is not done. For this reason, we introduce the **dummy state** *not done*, as shown in Figure 4.6.
6. As the number of states for *Result of E* has changed, its conditional probability table has been lost. Open the conditional probability table *Result of E* and introduce the sensitivity and specificity of the test. Indicate that when the test is not performed, the probability of *not done* is 1 (by definition), and when the test is done, the probability of *not done* is 0. It may be necessary to use the Reorder variables option to obtain the table shown in Figure 4.7.
7. Save this ID as `ID-decide-test-2therapies.pgmx`.

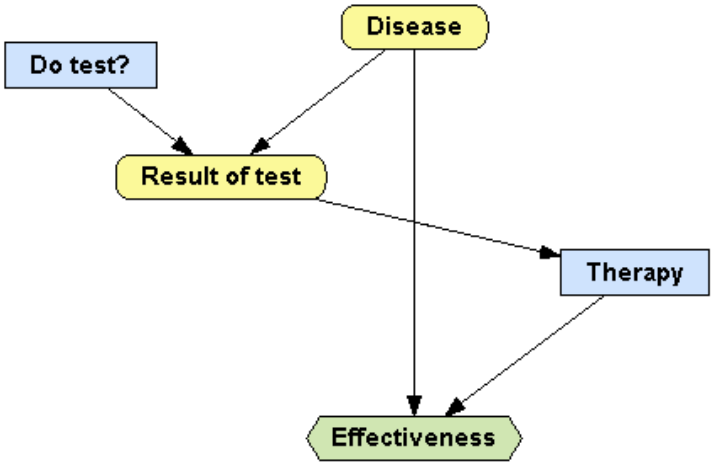


Figure 4.5: ID for Example 3.2. It contains two decisions.

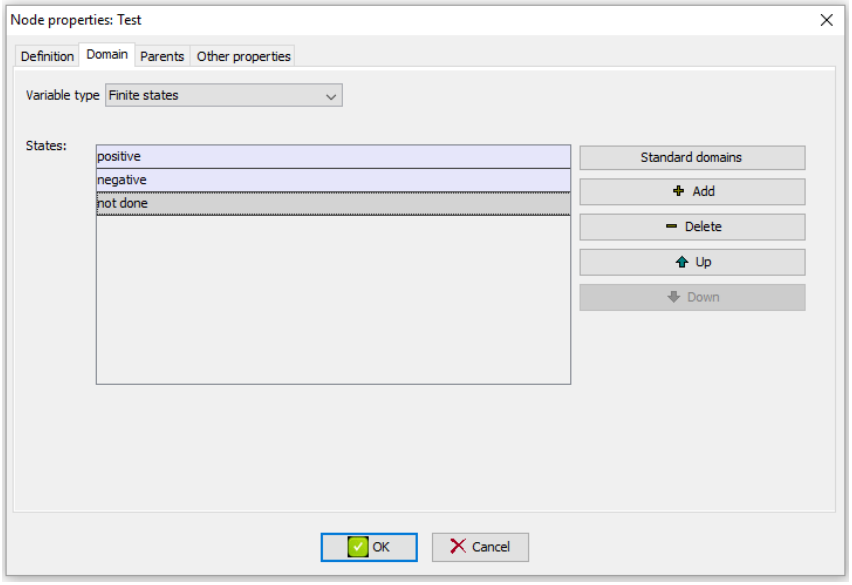


Figure 4.6: States of *Result of E* for the ID in Figure 4.1. We have added the dummy state *not done* to represent the result of the test when the decision for *Do test E?* is *no*.

4.2.2 Resolution of the influence diagram

Let us first evaluate the ID with the options that are also available for DANs:

1. Click the Optimal strategy button, S, and observe that the strategy is the same as in Figure 3.4.

2. Observe that, in contrast, the decision tree is bigger than that in Figure 3.9 because the node *Result of E* also appears in the branches for *Do test E?* = *no*, and it always has three branches, some of them with zero probability.

Then we evaluate it with the options specific for IDs:

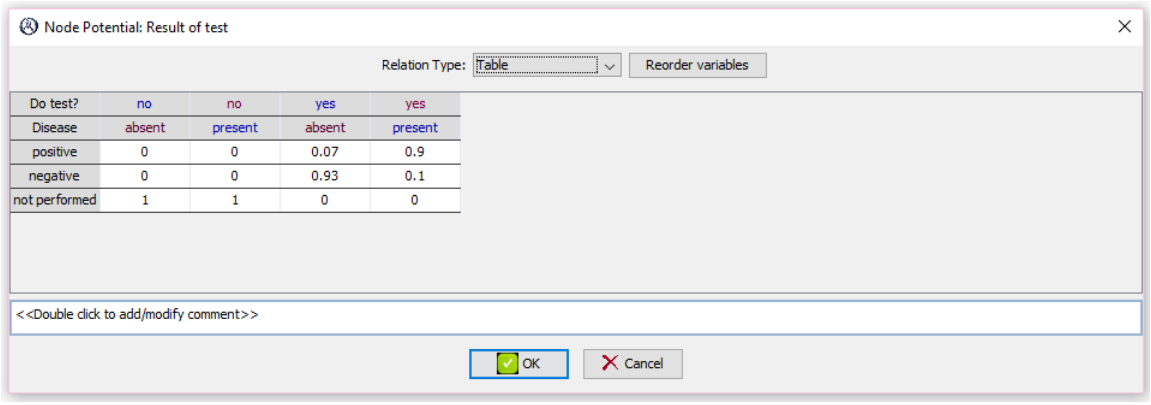


Figure 4.7: Conditional probability table for *Result of E*.

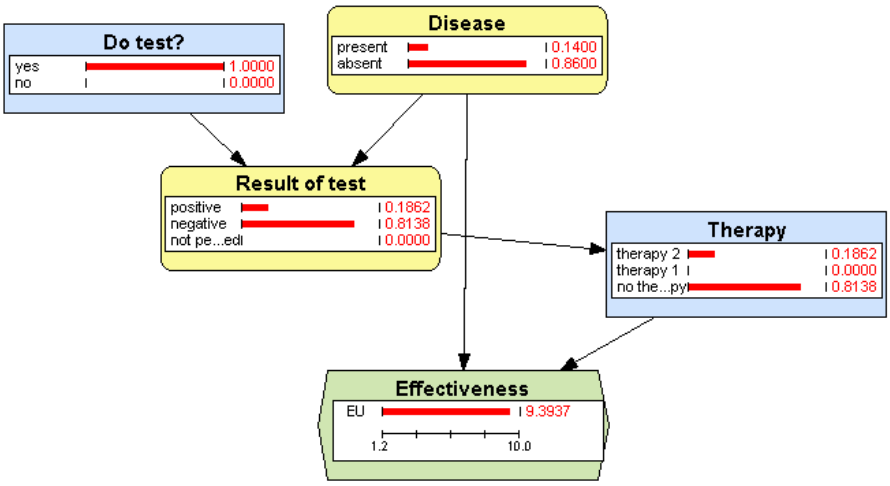


Figure 4.8: Propagation of evidence after resolving the ID in Figure 4.1.

1. Click on the Inference icon. The result is shown in Figure 4.8.

2. We examine first the policies that constitute the optimal strategy. Observe that the optimal policy for *Do test E?* is to always do the test (Figure 4.9) because the expected utility for

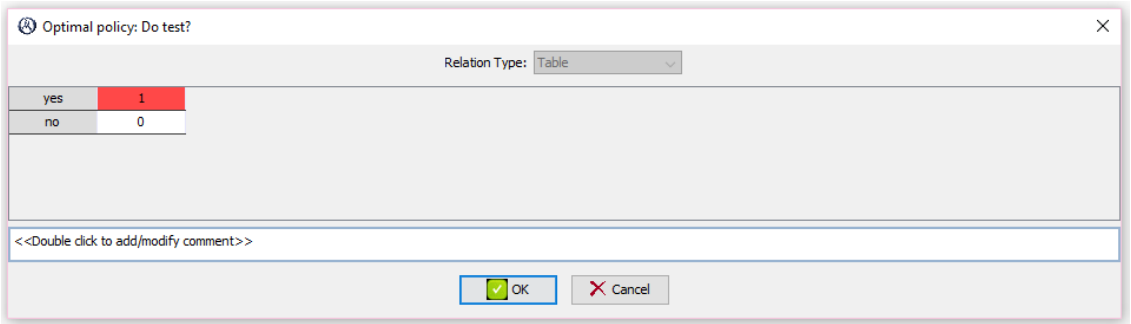


Figure 4.9: Optimal policy for the node *Do test E?*.

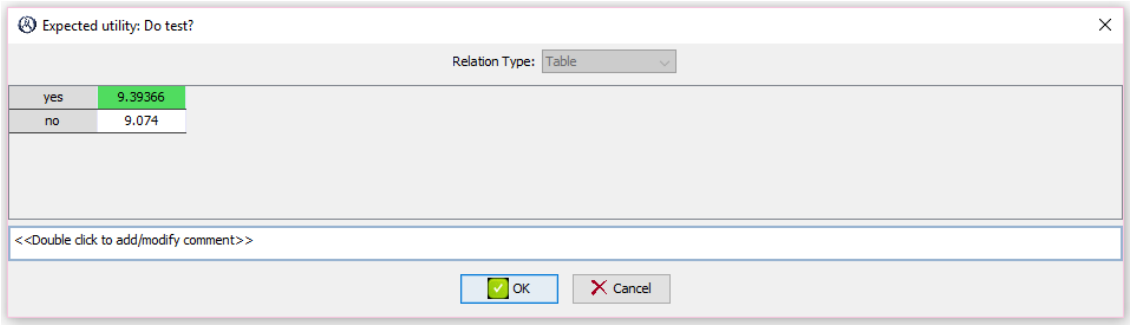


Figure 4.10: Expected utility for the node *Do test E?*.

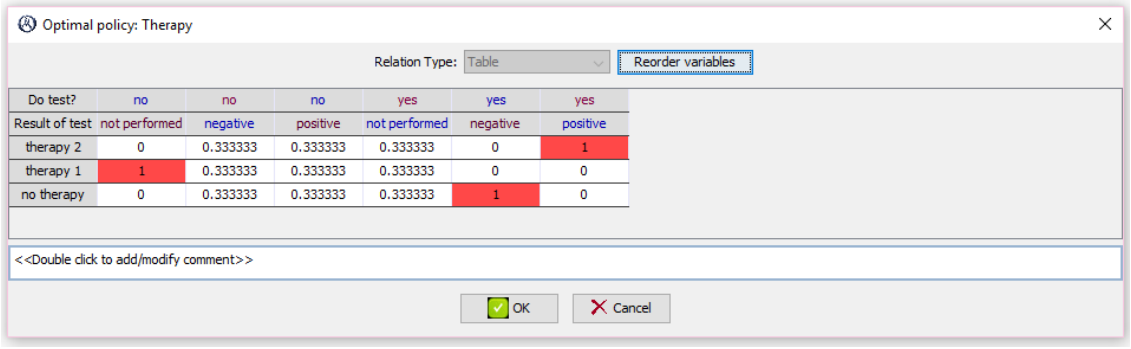


Figure 4.11: Optimal policy for the decision *Therapy* in the ID of Figure 4.1. The cells containing a probability of 1/3 correspond to impossible scenarios and so their value is irrelevant.

this option is higher (Figure 4.10).

3. Examine the optimal policy for *Therapy*, shown in Figure 4.11. In that table, the first column indicates that if the test is not performed the first therapy should be applied. The penultimate column indicates that no therapy should not be applied when the test is done and gives a negative result. The last column means that the second therapy should be applied when the test gives a positive result.

   The other three columns correspond to impossible configurations, and therefore the expected utility computed by the algorithm is 0 in all cases, as shown in Figure 4.12. Given that no option is better than the others, the algorithm has made the Solomonic decision of

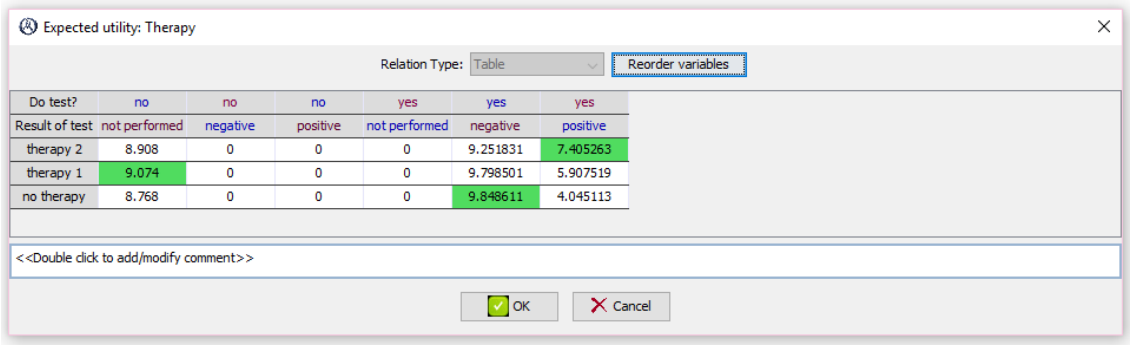


Figure 4.12: Expected utility for the decision *Therapy* in the ID of Figure 4.1. The cells with zero utility correspond to impossible scenarios.

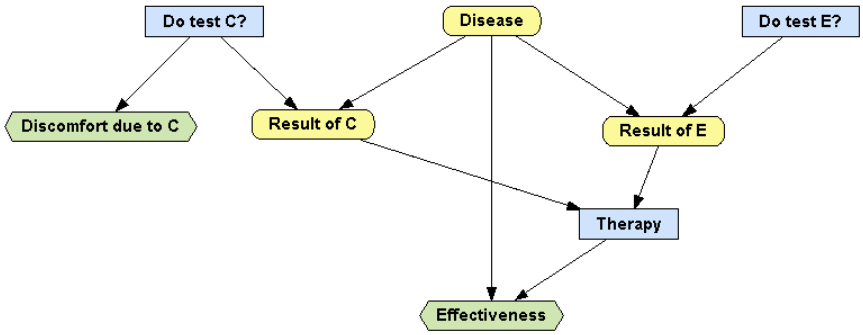


Figure 4.13: First attempt to build an ID for Example 3.3. Even after introducing the numerical parameters, this ID is incomplete because it does not contain any directed path connecting all the decisions.

not favoring any of them: it has assigned uniform probability distributions, which means that, using a random process (for example, a lottery) a third of the patients would receive therapy 2, another third would receive therapy 1, and the other third no therapy. However, the content of these columns is irrelevant because, as we said, they correspond to impossible scenarios that will never occur.[3]

4. After having examined the policies, we analyze the posterior probabilities and the expected utilities shown in Figure 4.8. Observe at the node *Do test E?* that the test will be done in 100% of cases; this is a consequence of the optimal policy for this decision (Figure 4.9).

5. Observe at the node *Therapy* that 9.68% of patients will receive the therapy, exactly the same proportion that will give a positive result in the test (see the node *Test*). This coincidence is a result of the policy for the node *Therapy*, namely to apply the therapy if and only if the test gives a positive result (see Figure 4.11).

6. The expected utility of *Effectiveness* is 9.3937, the same as in the example with only one decision (Figure 4.4) in which the test was always done.

4.3 An example with three decisions: limitations of IDs

Let us return to Example 3.3, in which there are were two tests. We may try to build the ID shown in Figure 4.13, which is similar to the DAN in Figure 3.10. After adding the nodes and drawing the links, we must introduce the conditional probability table for *Result of C*, which would be similar to that of *Result of E* (Fig. 4.7); but even then the ID is not complete, because it contains no directed path connecting all the decisions, which is, by definition, one of the requirements of IDs. The purpose of that requirement is to have a total ordering of the decisions. In this ID there is only a partial ordering, because the graph does not determine whether the first decision should be *Do test C?* or *Do test E?*. It is the decision modeler who has to establish an order between the two.

There are two possibilities that make sense. One of them is drawing a link from *Result of E* to *Do test C*, as shown in Figure 4.14 (we will save this network as `ID-decide-2tests-2therapies.pgmx`). In this case, the directed path *Do test E?* $\rightarrow$ *Result of E* $\rightarrow$ *Do test C?* $\rightarrow$ *Result of C* $\rightarrow$ *Therapy* would connect the three decisions, meaning that if decide to do the test E its result will be known when deciding about doing the other test, and the results of the tests done, if any, will be available when deciding which therapy to apply. The other possibility would be to draw a link from *Result of C* to *Do test E*. In both cases there will be a total ordering of the three decisions. Please note

[3] In future versions of OpenMarkov we will try to remove from policy tables those columns, but it is not an easy task because the probability of each scenario depends on the probabilities of its informational predecessors, so it is not in the table, which only contains utilities.

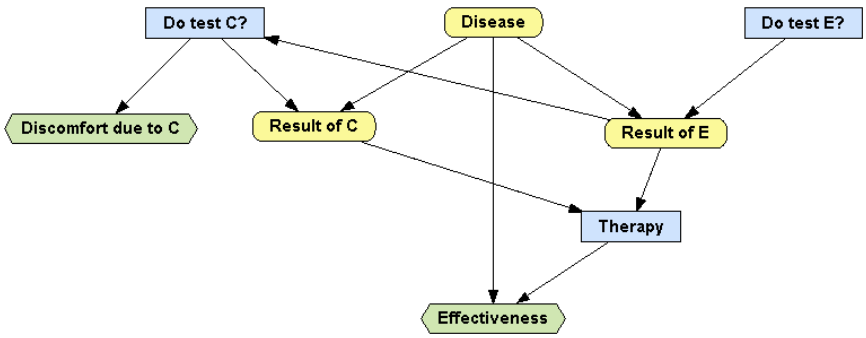


Figure 4.14: An ID for Example 3.3. Now the path *Do test E?* → *Result of E* → *Do test C?* → *Result of C* → *Therapy* connects all the decisions and establishes a total ordering among them, such that the decision about test E is done first. We should then build another ID in which we decide first about test C, and then compare the expected utilities of the two models.

that the first option corresponds to the branch $OD = Do\ test\ C?$ in Figure 3.12 and the second to the branch $OD = Do\ test\ C?$.

The main difference is that the DAN does not require a total ordering of the decisions when specifying the model; it is the inference algorithm that evaluates all the possible orderings and selects the optimal one. (As mentioned in Section 3.3.2, it even considers the possibility of making first the decision about the therapy, but discards it because it can never be better than the other orderings.) In contrast, in this example it is necessary to build one ID for each possible ordering of the decisions.

However, this "solution" might not work in other cases. For example, if there are three tests, A, B, and C, such that when A gives a positive result B should be decided before C but when it gives a negative result C should be decided first.

There is a modeling trick that in principle might be used to solve the n-test problem with one ID, but the model would be cumbersome for $n = 2$ (two tests) and virtually unfeasible for $n \geq 3$ [12]. For this reason, it is much better to use DANs, which are advantageous even for $n = 1$ because they do not need dummy states (cf. Sec. 4.2) and scale up nicely as we add more tests to the model.

4.4 Introducing evidence in influence diagrams*

This section, intended only for advanced users, describes the difference between the two kinds of findings for IDs in OpenMarkov: **pre-resolution** and **post-resolution**.

4.4.1 Post-resolution findings

Example 4.1. In the scenario of Example 3.2, what is the expected utility for the patients suffering from the disease?

This question can be answered with the ID in Figure 4.1 by *first* resolving the network (OpenMarkov does it when we switch to inference mode) and *then* introducing the finding *Disease = present*, as if it were a Bayesian network (cf. Sec. 1.3.1). The result, shown in Figure 4.15, tells us that the test will give a positive result in 90% of cases, which is the sensitivity of the test stated in Example 1.1. We can also see that 90% of those patients will receive the second therapy, as expected from the policy for *Therapy* obtained when resolving the ID (Fig. 4.11). For the same reason, the 10% of patients with a negative test result will receive no therapy. The expected utility, displayed at node *Effectiveness* in Figure 4.1, results from a weighted average: according to Table 3.1, the effectiveness is 6.5 for those having a positive test and 1.2 for those having a

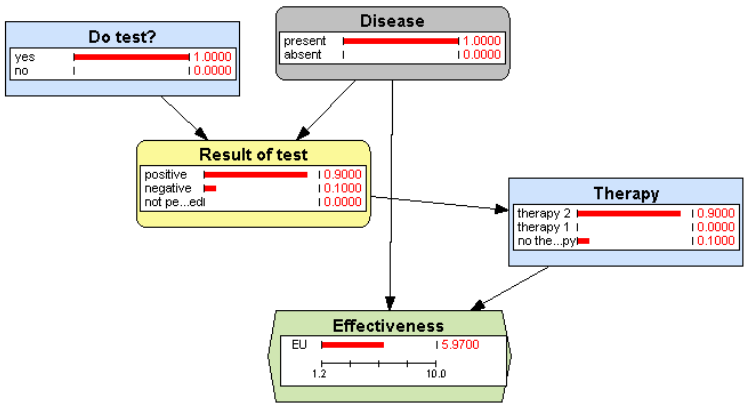


Figure 4.15: Introduction of the **post-resolution** finding *Disease* = *present*.

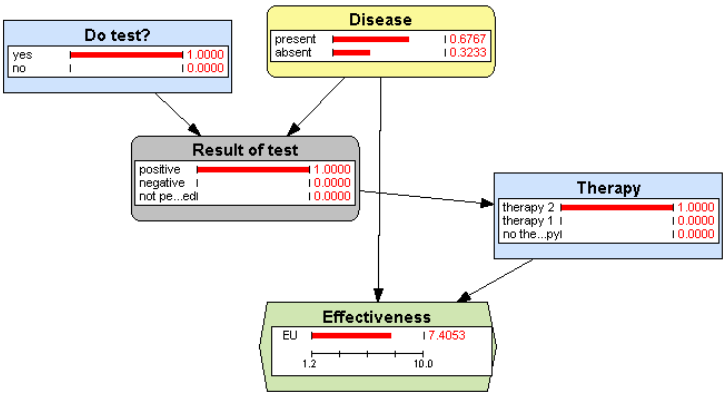


Figure 4.16: Introduction of the **post-resolution** finding *Test* = *positive*.

negative test; therefore,$6.5 \times 0.90 + 1.2 \times 0.10 = 5.97$. This answers the question posed in the example.

Example 4.2. In the scenario of Example 3.2, what is the probability that of suffering from the disease when the test gives a positive result?

This question can also be answered with the ID in Figure 4.1 in a similar way, i.e., introducing the post-resolution finding *Result of E* = *positive*, as in Figure 4.16, where we can see that the probability of suffering from the disease when the test gives a positive result (a true positive) is 67.67%. This answers the question under consideration. We can further analyze that figure and observe that 100% of the patients with a positive test will receive the second therapy and that their expected effectiveness is 7.4053, which is the weighted average of true positives and false positives (applying the second therapy): $6.5 \times 0.6767 + 9.3 \times 0.3233 = 7.4053$.

4.4.2 Pre-resolution findings

Example 4.3. Given the problem described in Example 3.2, what would be the optimal strategy if the decision maker—the doctor, in this case—knew with certainty that some patient has the disease?

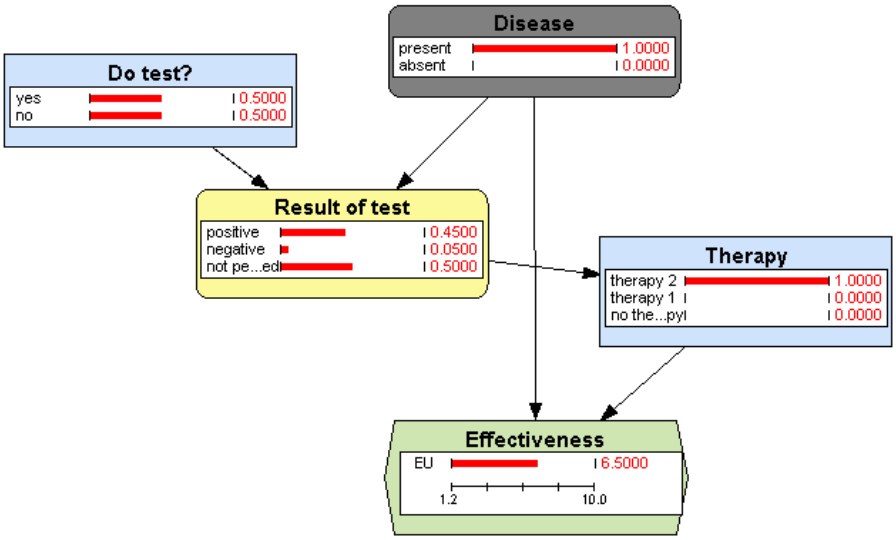


Figure 4.17: Introduction of the **pre-resolution** finding *Disease* = *present*.

One way to solve this problem with OpenMarkov is to use the ID built above, but introducing the finding *Disease* = *present* before resolving the ID; for this reason it is called a **pre-resolution finding**. This can be accomplished by using the Add finding option of the contextual menu of the node *Disease* while still in edit mode and then switching to inference mode; the result is shown in Figure 4.17. The optimal strategy computed by OpenMarkov is "apply the second therapy", obviously, because now we know with certainty that the disease is present.[4]

Another difference with the example in the previous section is that the node *Disease* in Figure 4.17 is colored in dark gray to indicate that it has a pre-resolution finding, while the lighter gray in Figure 4.15 denotes a post-resolution finding.[5]

The main difference between the Examples 4.1 and 4.3 is that in the latter the decision maker (the doctor) knows that the patient has the disease before making the decisions about the test and the therapy—this is what a pre-resolution finding means—and for this reason he/she does not need to do the test, while if in the Example 4.3 the doctor has decided to do the test to all patients because he/she did not know with certainty who has the disease. The question posed in that example takes the perspective of an external observer who, knowing the decision maker's strategy, i.e., how he/she will behave, uses the ID—the model—to compute the posterior probabilities and the expected utilities for different subpopulations. Therefore the strategy does not depend on the information known by the external observer, which is unknown to the decision maker.

Pre-resolution findings, which are known to the decision maker since the beginning, i.e., before making any decision, correspond to concept of evidence defined in [14], while post-resolution findings correspond to the concept of evidence proposed in [21].

[4]In this situation it is irrelevant whether we do the test or not (there is a tie between the expected utilities of both states, because we have not yet considered the cost of the test), and for this reason the algorithm has assigned a uniform policy to the decision *Do test E?*; it implies that the test will be done in 50% of cases, and in 90% of them will give a positive result, i.e., in 45% of the total of patients. But, we insist, the policy for *Therapy* is to always apply the second therapy, regardless of the result of the test.

[5]As a mnemonic, remember that a *darker* gray denotes a *stronger* impact on the ID, because a pre-resolution finding affects both the optimal strategy and the posterior probabilities and utilities, while a post-resolution finding only affects the posterior probabilities and utilities. Another mnemonic is that a *lighter* gray denotes a finding that arrives *later*, i.e., after the resolution.

4.5 Imposing policies for what-if reasoning*

OpenMarkov allows the user to impose policies on decision nodes. The decisions that have **imposed policies** are treated as if they were chance nodes, i.e., when the evaluation algorithm looks for the optimal strategy it only returns **optimal policies** for those decisions that do not have imposed policies, and these returned policies, which are optimal in the context of the policies imposed by the user, may differ from the absolutely optimal policies.

The purpose of imposing policies is to analyze the behavior of the ID for scenarios that can never occur if the decision maker applies the optimal strategy.

Example 4.4. Let us consider again the problem defined in the Example 3.2, which can be solved with the ID shown in Figure 4.14. We saw that the optimal strategy is: "do the test E; it give a positive result, apply the second therapy; if it is negative, do test C and if it gives a negative result apply the first therapy; otherwise, do not apply any therapy". We may wonder why it is not worth doing test C when E has given a positive result. According to our intuition, a negative result in C might lower the probability of the disease and in that case it would be better to apply the first therapy or no therapy. We are therefore interested in the probability $P(Disease = present \mid Result\ of\ E = positive, Result\ of\ C = negative)$.

In order to solve this problem, we try the following:

1. Open again the network `ID-decide-test-2therapies.pgmx`.
2. Switch to Inference mode.
3. Enter the findings *Result of* E = *positive* and *Result of* C = *negative*.

You will get an Incompatible evidence error, according with the optimal strategy, when E has given a positive result the decision maker will never to test C, so the second finding is incompatible with the first. One way to solve this problem is to **impose a policy**—in this case, a suboptimal policy—as follows:

1. Come back to edit mode because policies can only be imposed before resolving the ID.
2. In the contextual menu of *Do test E?*, select Impose policy. In the field Relation type select *Delta* (because the option chosen will be the same in all cases, regardless of the values taken by the informational predecessors of this decision) and in the state Field select *yes* (because we will always do the test).

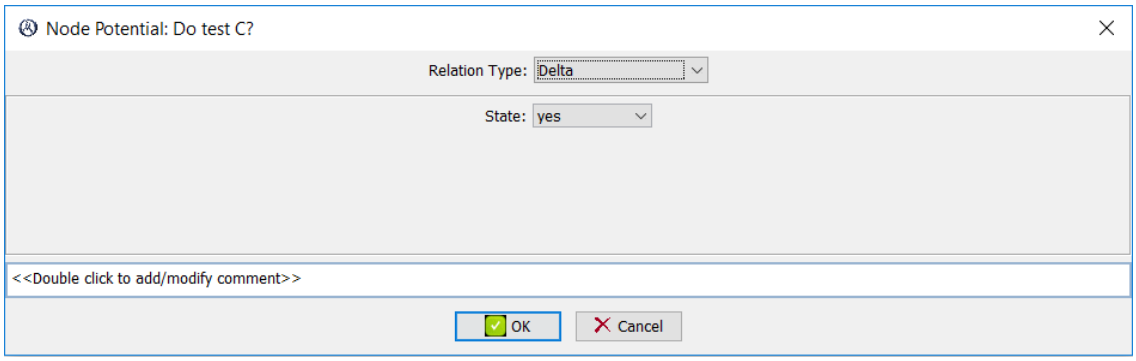


Figure 4.18: Imposed policy for the decision *Do test E?*.

3. Click OK and observe that the node *Do test E?* is now colored in a darker blue to indicate that it has an imposed policy.
4. Evaluate the ID. Observe that now the probability of doing this test is 100%, as we wished.

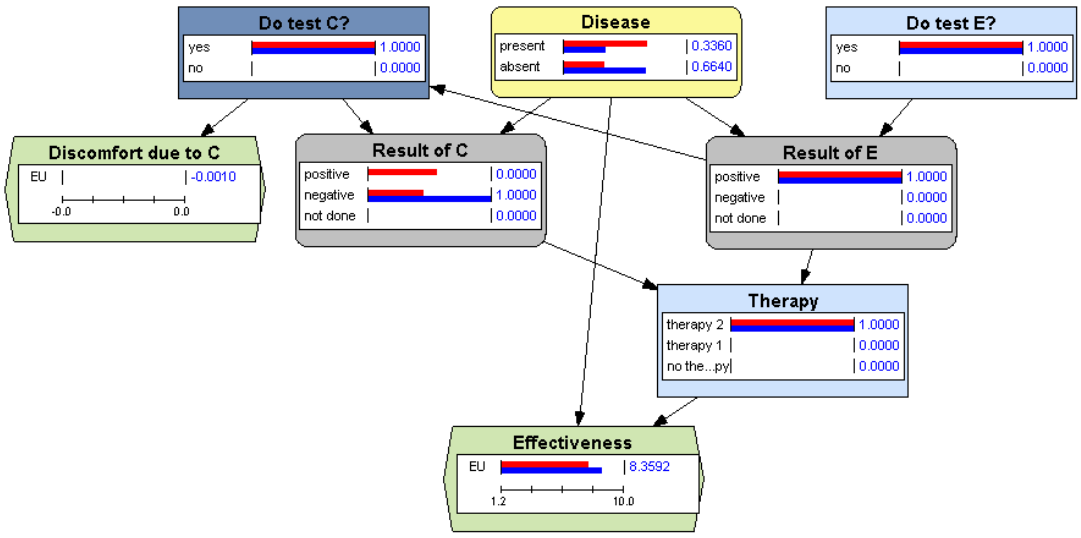


Figure 4.19: Introduction of evidence with an imposed policy.

5. Also observe that when we introduce the finding *Result of E = positive* the probability of *Disease = present* increases to 0.6767, but after introducing the finding *Result of C = negative* it only decreases to 0.3360 (cf. Fig. 4.19).

In a sensitivity analysis (which we will explain in a future version of this tutorial) we would be able to see that when the probability of the disease is above 0.195, the optimal policy for *Therapy* is to apply the second therapy. Therefore, when test E is positive, C can never lower the probability *Disease* below that threshold and since the result of test C cannot change the therapeutic policy, it is not worth doing it.

Chapter 5

Multicriteria decision analysis

The examples we have considered so far only had one decision criterion, the effectiveness. In this chapter we will consider examples involving two criteria, effectiveness and cost, and explain how to encode networks containing any number of criteria.

Example 5.1. The costs of the two tests and the two therapies introduced in the previous examples are shown in Table 5.1. We will assume from now on that the effectiveness indicated in Example 3.1 is measured in quality-adjusted life years (QALYs). What is the optimal strategy?

Intervention	**Cost**
Therapy 1	€ 20,000
Therapy 2	€ 70,000
Test C	€ 18
Test E	€ 150

Table 5.1: Cost of each intervention.

The problem we face now is that the problem involves two criteria, measured in different units: € and QALYs. We will first represent the problem and then solve it.

5.1 Construction of multicriteria networks

The representation of the problem can be done as follows:

1. Open the influence diagram `ID-decide-2tests-2therapies.pgmx` built in Sec. 4.3.
2. Right-click on a blank zone of the network panel to open the Network properties dialog. Select the tab Advanced, click the buttons Decision criteria and Standard criteria, and select Cost-effectiveness (€/QALY). The result is shown in Figure 5.1.
3. Open the Node properties dialog for the node *Effectiveness* and set its Decision criterion to *Effectiveness*.
4. Do the same for *Discomfort due to C*.
5. Create the utility node *Cost of* therapy, as shown in Figure 5.2. Check that its Decision criterion is *Cost*, because by default the decision criterion for value nodes is the first of the decision criteria of the network.
6. Create the utility nodes *Cost of C* and *Cost of E* , checking that their Decision criterion is *Cost*.

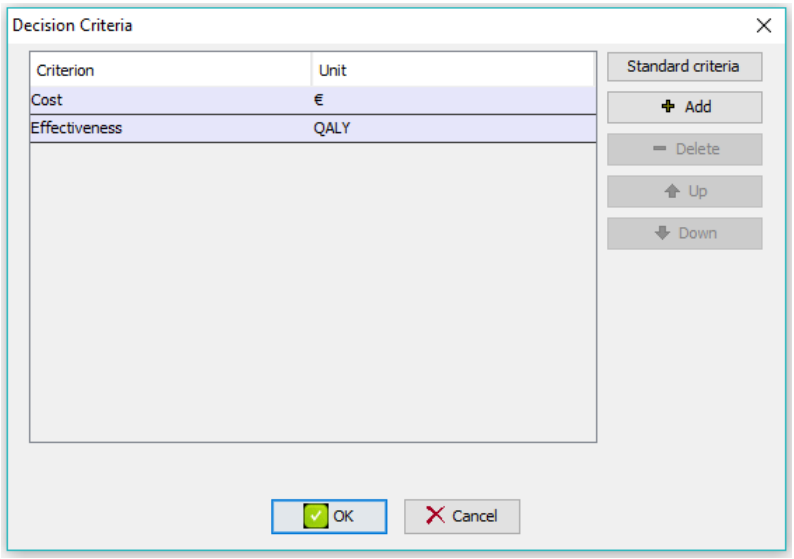


Figure 5.1: Cost-effectiveness decision criteria.

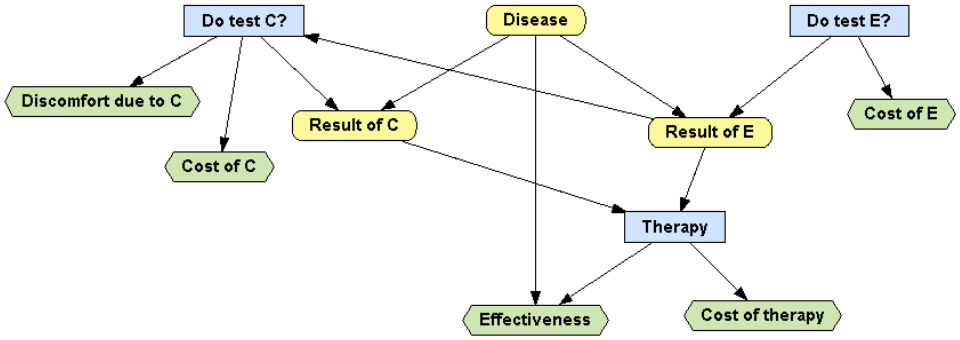


Figure 5.2: An ID with two criteria: cost and effectiveness.

7. Draw the links *Do test C?*→*Cost of C*, *Do test E?*→*Cost of E*, and *Therapy*→*Cost of therapy,*

8. Open the potential for the node *Cost of therapy*. In the Relation type field select *Table* and introduce the cost of each therapy, as shown in Figure 5.3.

9. Open the potential for *Cost of C* and introduce the cost of the test, € 18.

10. Open the potential for *Cost of E* and introduce the cost of the test, € 150.

11. Save the network as `ID-decide-2tests-2therapies-CE.pgmx`.

5.2 Analysis of multicriteria networks

OpenMarkov offers two ways to solve multicriteria problems. The first one consists in mapping all the criteria into a single one. The second one is cost-effectiveness analysis. We describe them in the next sections.

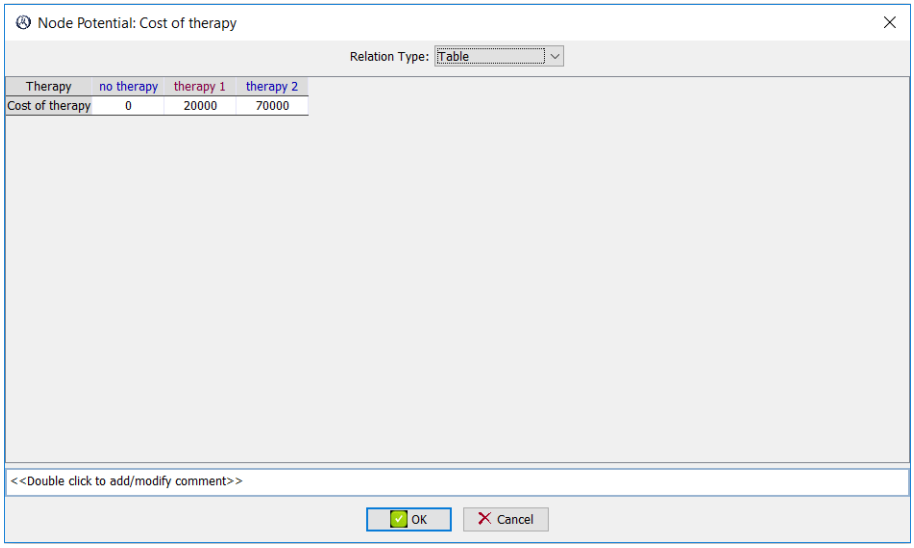


Figure 5.3: Cost of test E.

5.2.1 Unicriterion analysis

Let us assume that a model has n criteria, $\{C_1, \ldots, C_n\}$. They can be combined into a single criterion, U, by assigning a weight α_i (also called scale factor) to each C_i:

$$U = \alpha_1 C_1 + \ldots + \alpha_n C_n \ .$$

In particular cost and effectiveness can be combined into a single criterion, the net monetary benefit (NMB) [30], as follows:

$$NMB \ = \lambda E - C \ ,$$

where E is the effectiveness, C is the cost, and λ is a parameter that transforms the effectiveness into monetary units. Please note that this equation is a particular case of the previous one, with λ as the scale factor for the effectiveness and -1 for the cost.

The parameter λ is called *willingness to pay* (WTP) or *cost-effectiveness threshold*. It is different from each decision maker. In health economics, when the effectiveness is measured in QALYsλ usually denotes the WTP of the health care system for which the analysis in conducted. For instance, the WTP of the Spanish public health system agreed as a consensus among experts is €30,000/QALY.

For this WTP, the above ID can be evaluated as follows:

1. Select Inference ▷ Inference options.
2. Observe that by default the type of Analysis is Unicriterion.
3. Set the Unit to €.
4. Set the scales for cost and effectiveness to -1 and €30,000/QALY respectively.

Once we have a unicriterion ID, we can perform the same type of analyses described in Chapter 4: resolve the ID, show the expected utilities, the optimal policies and the optimal strategy (which is shown in Fig. 5.5), introduce pre- and post-resolution evidence, impose policies, etc.

5.2.2 Cost-effectiveness analysis

In many fields it is difficult to assign a monetary value to effectiveness. In medicine, in particular, it is very hard to put a price to human life. Therefore, the value of λ is often unknown, or at

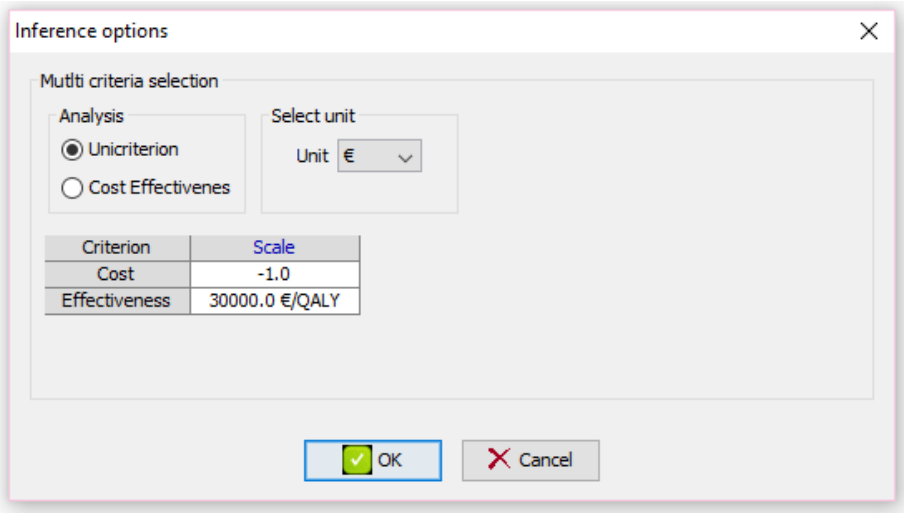


Figure 5.4: Options for converting cost and effectiveness into a single criterion, the net monetary benefit (NMB), when the willingness to pay (WTP) is $\lambda = €30,000/\text{QALY}$.

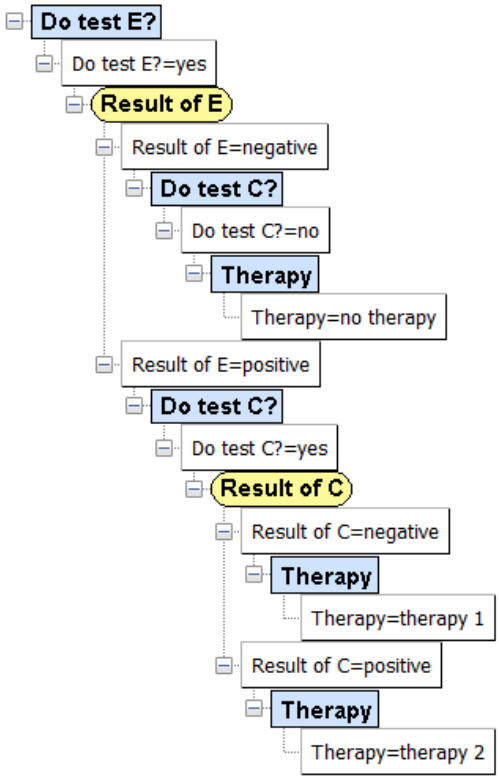


Figure 5.5: Optimal strategy resulting from a unicriterion analysis of the ID in Figure 5.2 with $\lambda = €30,000/\text{QALY}$.

least very imprecise, which makes it impossible to join the two criteria into a single one. Cost-effectiveness analysis (CEA) is implicitly based on the concept of NMB introduced above, but it consists just in expressing the results as a function of λ, so that each decision maker can take the strategy corresponding to their own WTP.

In OpenMarkov CEA can be accessed by selecting Tools ▷ Cost-effectiveness analysis ▷ Deterministic analysis, by clicking the Cost-effectiveness analysis icon (), or by pressing Ctrl+T.

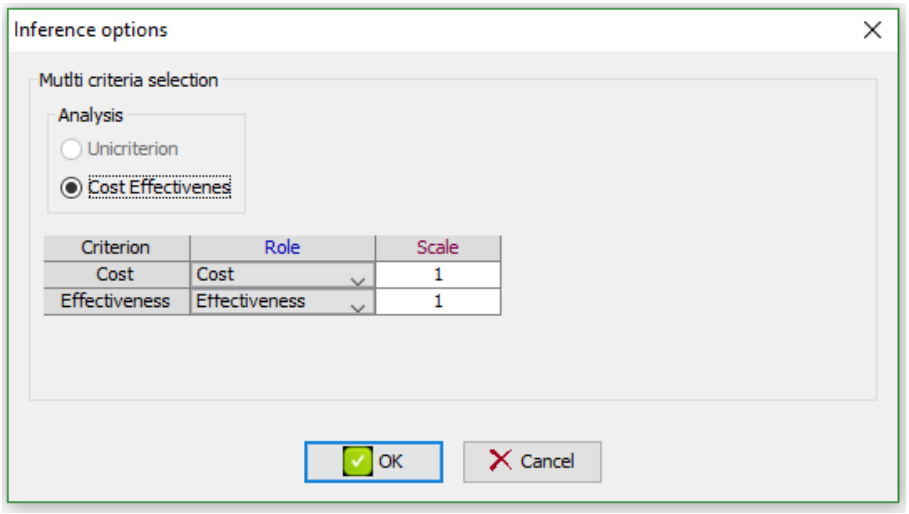


Figure 5.6: Inference options for cost-effectiveness analysis.

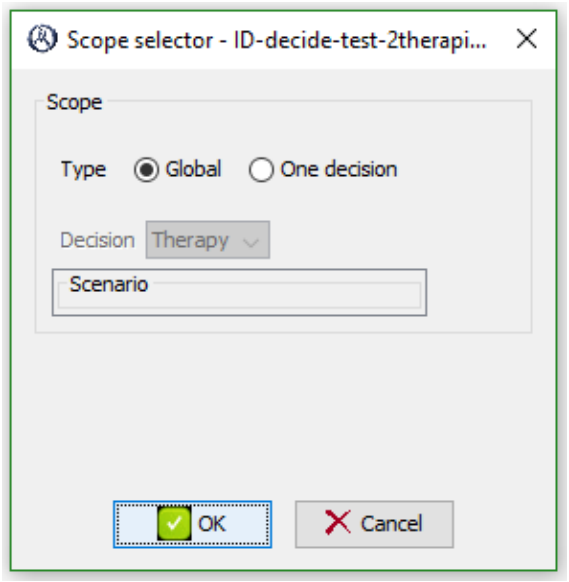


Figure 5.7: Types of cost-effectiveness analysis.

The first dialog launched, shown in Figure 5.6, allows us to select the criteria to be used in the analysis. In our example this dialog may seem unnecessary but in other models having more criteria it may be useful; for example, if we have defined several types of cost (cost for the health care system, costs covered by the patient, etc.), we can perform the CEA from different perspectives by including some criteria and excluding others.

The next dialog (cf. Fig. 5.7) allows us to choose how to perform the CEA: globally or for one decision in one scenario.

5.2.2.1 Global cost-effectiveness analysis

If we select Global, we obtain the table shown in Figure 5.8. We observe that the cost, the effectiveness, and the optimal strategy depend on the value of λ, and the six intervals shown in this table cover all the values of λ, from 0 to $+\infty$. For example, when $0 < \lambda < €8,115.37/\text{QALY}$, the cost is 0, the expected effectiveness (i.e., the one expected on average for every patient) is

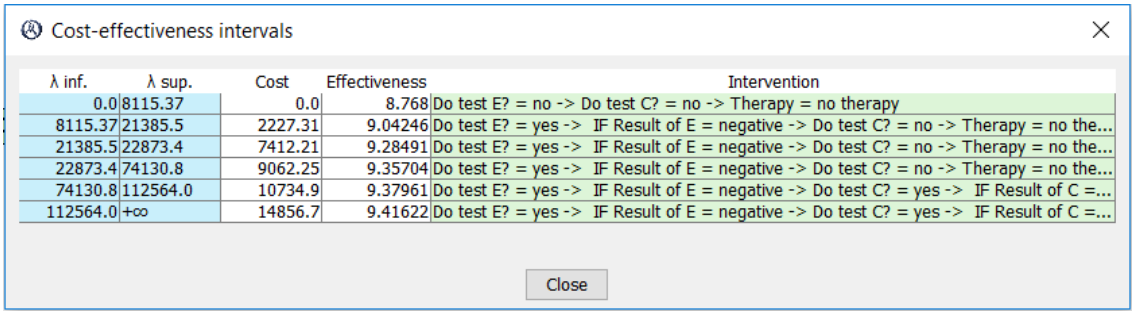


Figure 5.8: Global cost-effectiveness analysis.

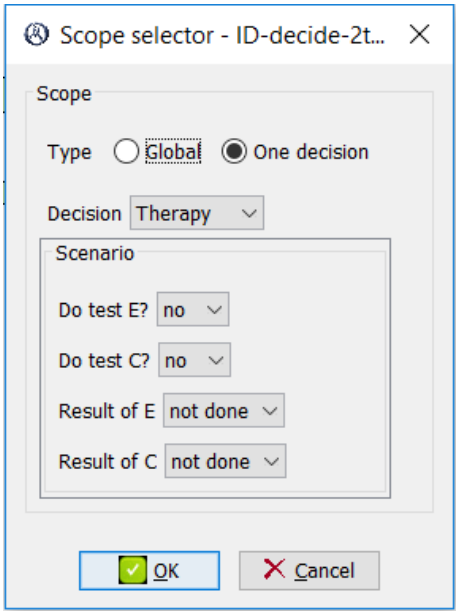


Figure 5.9: Selection of a decision and a scenario for CEA.

8.768, and the optimal strategy is not to do any test and not to apply any therapy. The value $\lambda = €30000/\text{QALY}$ falls within the fourth interval; if you click on the intervention for this row (the green cell), you will obtain the same strategy as in our unicriterion analysis (cf. Fig. 5.5), as expected.

The algorithm that OpenMarkov uses to perform this analysis is described in [1].

5.2.2.2 Cost-effectiveness analysis for a decision

This type of analysis refers to a decision in a scenario. It is performed by selecting One decision in the scope selector (cf. Fig. 5.7). We must select first the decision and then the scenario is determined by assigning one value to each informational predecessor of the decision, as shown in Figure 5.9; please note that this choice of decision and scenario corresponds to the first *Therapy* node in the decision tree.

Then OpenMarkov shows a dialog with tree tabs. In the first one, Analysis (cf. Fig. 5.10), we can observe that the option *therapy 1* is cheaper and more effective than *therapy 2*; this implies that the NMB of the first therapy is always higher than that of the second, regardless of the value of λ. We then say that *therapy 1* dominates *therapy 2*. In contrast, *therapy 1* is more expensive than *no therapy*, but also more effective; in this case λ determines which option has a higher NMB.

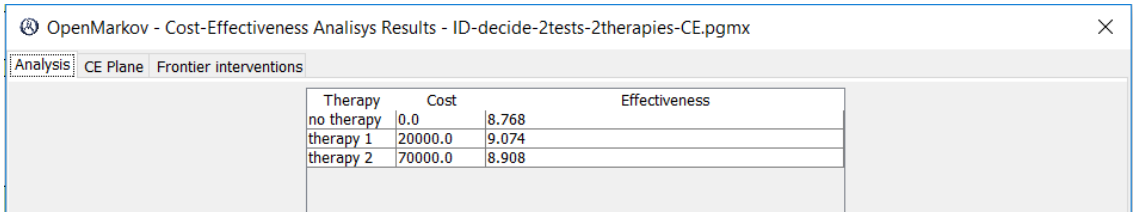


Figure 5.10: Results of the CEA for the decision *Therapy* when no test has been performed.

When we have two interventions (two options) A and B such that B is more effective and more effective, we can define the incremental cost-effectiveness ratio (ICER) as follows:

$$ICER_{B,A} = \frac{C_B - C_A}{E_B - E_A} \ .$$

It is easy to prove that

$$NMB_B > NMB_A \iff \lambda > ICER_{B,A} \ .$$

According to the values shown in 5.10, the ICER for *therapy 1* with respect to *no therapy* is

$$ICER \ = \frac{€20,000 - €0}{9.074 \text{ QALY} - 8.768 \text{ QALY}} = €65,359/\text{QALY} \ .$$

This is the value that OpenMarkov shows in the tab Frontier interventions of the same dialog. In that dialog *therapy 2* does not appear because it is dominated. We can also observe the plot in the CE plane tab. The horizontal axis represents the effectiveness and the vertical one the cost. The slope of line that connects the points for *no therapy* and *therapy 1* is determined by their ICER. We can appreciate it more clearly if instead of showing the absolute values of cost and effectiveness we tell OpenMarkov to show the values relative to *no therapy*; it is done by selecting Relative to in the Display area at the top right corner of that tab—see Figure 5.11.

Similarly, we can analyze the cost-effectiveness of *Therapy* for the scenario in which both tests are done and both return a positive result. The CE plane for this case is shown in Figure 5.12.

Please note that it is possible to perform a CEA for any decision, not necessarily the last one. For example, we can analyze the cost-effectiveness of test E. As the decision *Do test E?* has no informational predecessors, there is only one possible scenario. We can observe in Figure 5.13 that the test is cost-effective when cost-effective when $\lambda > 8,115.40$—the same conclusion shown in Figure 5.8—but the cost and the effectiveness depend on the optimal policies for the two subsequent decisions (*Do test E?* and *Therapy*), which in turn on the WTP. For this reason in that figure there are 9 intervals for λ and the CE plane is different from each of them.

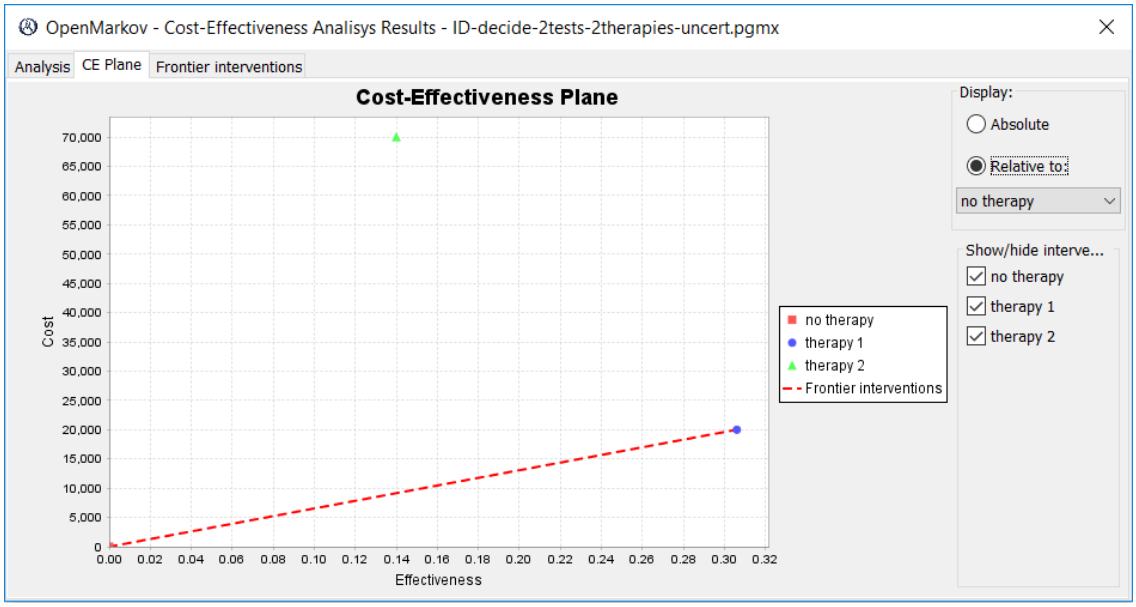


Figure 5.11: Results of the CEA for the decision *Therapy* when no test has been performed. It is obtained from the numerical results shown in Figure 5.10, making them relative to the values for *no therapy*. *Therapy 2* is dominated by *Therapy 1* because the latter is cheaper and more effective. The slope of the line that connects the points for *no therapy* and *therapy 1* determines the incremental cost-effectiveness ratio (ICER) for these two interventions.

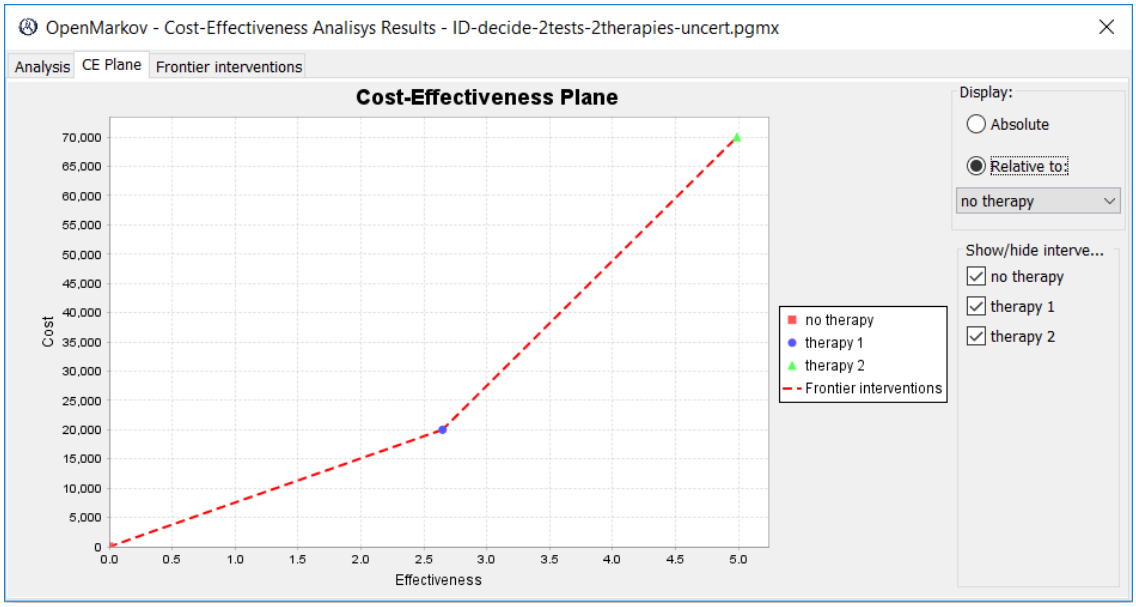


Figure 5.12: Results of the CEA for the decision *Therapy* when both tests have been positive. The optimal intervention is *no therapy*, *therapy 1*, or *therapy 2*, depending on the value of λ, the WTP.

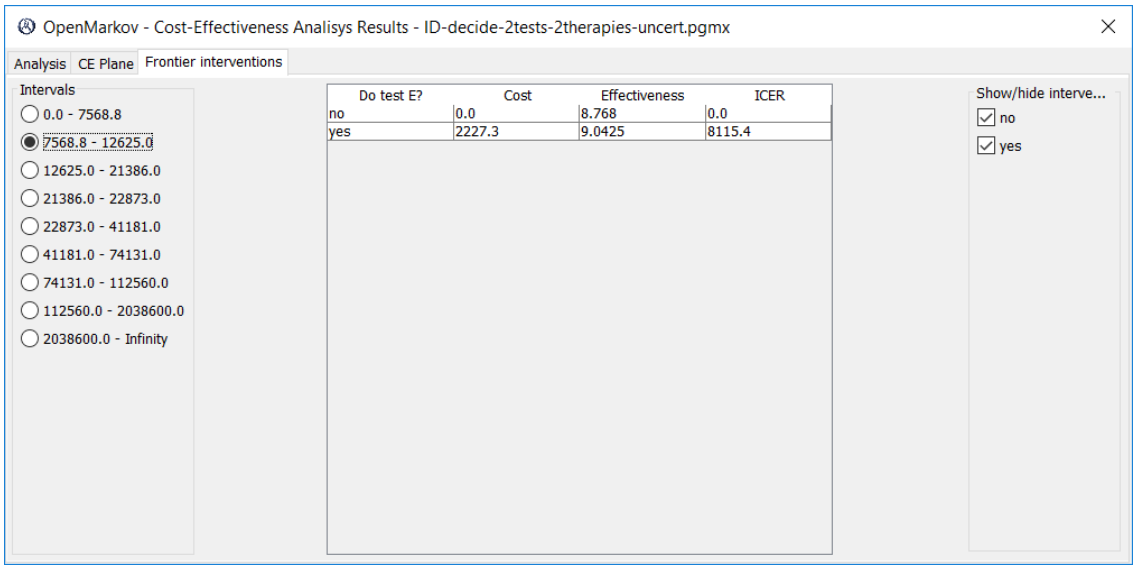


Figure 5.13: Results of the CEA for the decision *Do test E?* The test is cost-effective when $\lambda > 8,115.40$, but the cost and the effectiveness depend on the WTP because λ affects the optimal policies for the two subsequent decisions (*Do test E?* and *Therapy*).

Chapter 6

Sensitivity analysis

When building a model there are usually several sources of uncertainty. In this chapter we will focus on one type of uncertainty: the imprecision about the numerical parameters, which we will encode by assigning a **second-order probability distribution** to each parameter. This name is introduced to distinguish them from the probability distributions studied so far, such as the prevalence of the disease and the sensitivity specificity of each test, which are said to be **first-order**. We explain first how to encode the second-order uncertainty and then how to perform several types of deterministic and probabilistic sensitivity analyses.

6.1 Encoding the uncertainty

Before you start introducing uncertainty in OpenMarkov, we strongly recommend you to read the sections of technical report [2] that specifies how to encode second-order uncertainty in the format ProbModelXML; it is available at `http://www.cisiad.uned.es/techreports/ProbModelXML.php`. This is the native format used by OpenMarkov to store probabilistic graphical models. In version 0.2.0 of that technical report the explanation of how to encode second order uncertainty in probability tables is given in Section 4.4.2.b (page 26).

1. Open the network `ID-decide-2tests-2therapies-CE.pgmx`, which we built in Sec. 5.1.
2. Open the Edit probability dialog for node *Disease*. It still contains the table we built for the Bayesian network (Fig. 1.2).
3. Right-click on any of the two numeric cells and select Assign uncertainty.
4. Click on the Distribution cell for the state *present* and change its content from *Exact* to *Beta*. Set the parameter alpha to 28 and beta to 172.[1]
5. Click on the cell Distribution cell for the state *absent* and change its value to *Complement*. Set the parameter nu to any positive integer; for instance, 1. The result should look like in Figure 6.1.
6. Click OK and observe that the values of the cells has not changed—because the mean value for a Beta(28,172) is 0.14, the same prevalence we had before introducing evidence—but know there is a red triangle in each cell, meaning that there is uncertainty about the parameters. If you wish to change the uncertainty, right-click on any of those cells.
7. Insert the uncertainty for *Result of E* as shown in Figure 6.2. The probability table for this node will look like in Figure 6.3

[1] This information might stem, for example, from an experiment that involved 200 people from the population of interest; 28 of them had the disease and 172 did not.

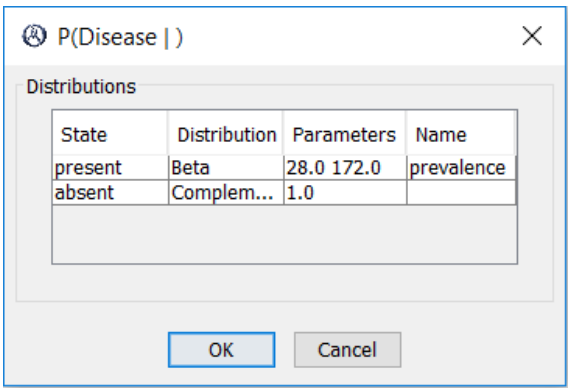


Figure 6.1: Uncertainty for the probability distribution $P(Disease)$.

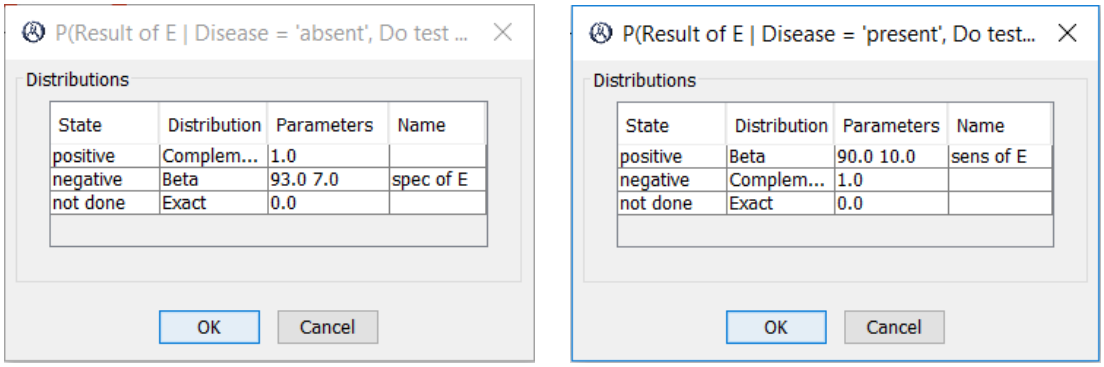


Figure 6.2: Uncertainty for the probability distribution $P(Result\ of\ E\ |Disease, Do\ test\ E?)$. Both tables correspond to the configuration *Do test E?* = *yes*. The table on the left corresponds to *Disease* = *absent* and the one on the right to *Disease* = *present*. They contain the specificity and the sensitivity of the test, respectively.

8. Save the network as `ID-decide-2tests-2therapies-uncert.pgmx`.

We will now encode in the same network the uncertainty about costs. We will model the uncertainty about each cost with a Gamma distribution with the same mean as in Table 5.1 and a standard deviation of 20% [5].[2] In OpenMarkov we can encode the Gamma using the parameters k (shape) and θ (scale), but it is also possible to use the mean and the variance.

1. Open the Edit utility dialog for node *Cost of therapy*. Right-click on the cost of *therapy 1*, select Assign uncertainty and set the type of distribution to *Gamma-mv*, where "mv" stands

[2]Instead of a Gamma we might use a log-normal distribution. It would not be correct to assign a normal distribution (a Gaussian) because it would lead to a non-null probability of the cost being negative.

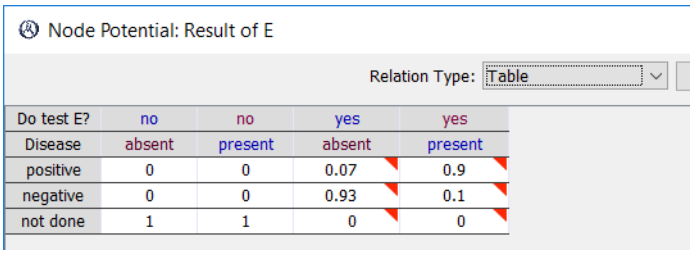


Figure 6.3: Conditional probability table for *Result of E*, with uncertainty about the sensitivity and specificity of the test. However, when the decision for *Do test E?* is no, we have absolute certainty that *Result of E* is *not done*.

for "mean and variance". Set the mean to € 20.000 and the standard deviation to € 4.000 (20% of the mean). Set the name of this parameter to "cost of therapy 1"; don't forget to press Tab or click outside this cell so that the name you have introduced is not lost (in future versions of OpenMarkov this will not be necessary).

2. For the cost of therapy 2, assign a Gamma with a mean of € 70.000 and a standard deviation of € 14.000. Set the name for this parameter to "cost of therapy 2".

3. For the cost of test E, assign a Gamma with a mean of € 150 and the standard deviation to € 30. The name must be "cost of E".

4. Save the network again.

If you wish, you can now assign uncertainty to the values of test C (sensitivity, specificity, and cost) and to each of the four values of effectiveness. Don't forget to set a name to each parameters so that you can use them in the analyses.

6.2 Sensitivity analyses

There are two main types of sensitivity analysis (SA) on the parameters of the model: deterministic and probabilistic. In general deterministic SA is based on assigning intervals to one or several parameters. In contrast, probabilistic SA is based on the second-order distributions assigned to some parameters, and is usually carried out by means of stochastic simulations (Monte Carlo techniques).

Additionally, decision analyses can be unicriterion or multicriteria; as mentioned above, a particular case of multicriteria analysis is cost-effectiveness analysis (CEA). Therefore, there are at least 4 types of SA: deterministic unicriterion, probabilistic unicriterion, deterministic for CEA, and probabilistic for CEA. However, it is common in practice to consider only the first type and the fourth, and these are types currently implemented in OpenMarkov. However, future versions of this software will offer facilities for all the types of SA.

6.2.1 Deterministic unicriterion sensitivity analysis

Currently OpenMarkov offers three facilities of unicriterion deterministic analysis: tornado diagrams, spider diagrams, and plots (one-way sensitivity analysis). In the future it will offer another facility: maps (two-way sensitivity analysis).

6.2.1.1 Tornado and spider diagrams

Tornado diagrams and spider diagrams contain essentially the same information, but it is presented in different ways.

1. Select Tools ▷ Sensitivity Analysis or click on the sensitivity analysis icon () and accept the default values of the Inference options dialog. This will open the Deterministic sensitivity analysis dialog. Observe that the default value for Analysis type is Tornado/spider diagram, with 50 points per parameter, and all the parameters of the model for which there is uncertainty have been selected.

2. In order to generate a tornado diagram or a spider diagram it is necessary to specify an interval for each parameter. In OpenMarkov there are four ways of specifying the intervals for the variation of the Horizontal axis parameter.

   (a) By default, OpenMarkov takes for each parameter an interval that encloses a Percentage of the 2nd order probability. In our example, the second-order probability for the prevalence of *Disease* is a Beta with $\alpha = 70$ and $\beta = 430$. The interval $[0.1205, 0.1602]$ contains 80% of the probability—in fact, 10% of the second-order probability is below 0.1205 and the other 10% is above 0.1602.

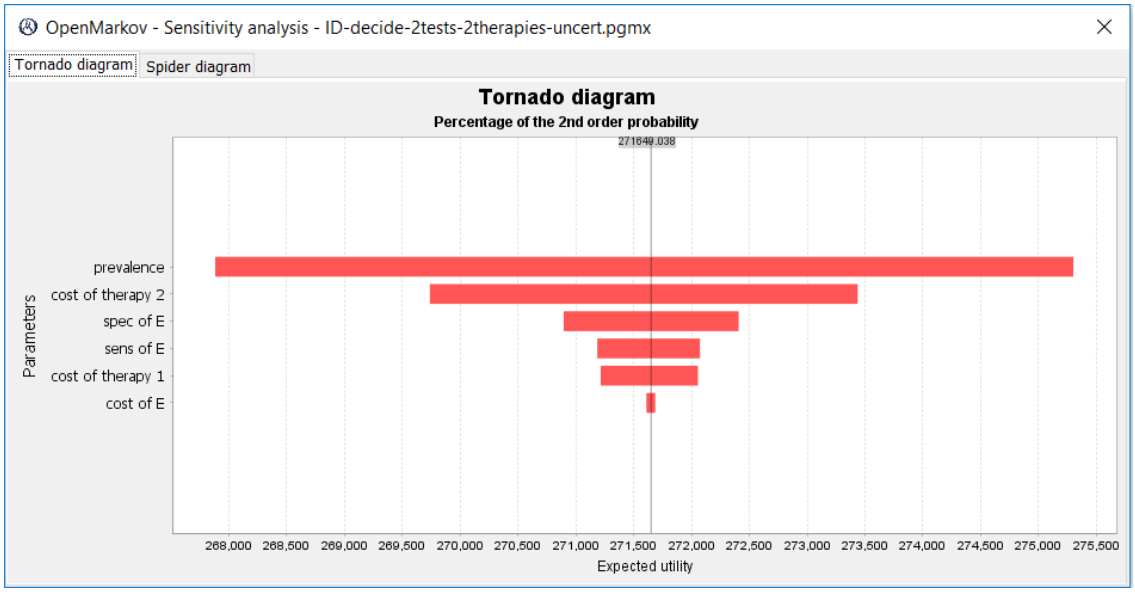


Figure 6.4: A tornado diagram for the ID with uncertainty. The horizontal axis represents the variation in the expected utility for each parameter.

(b) An alternative way of specifying each interval is to take a Percentage over reference value—for example, 25%. Thus, if the reference value for the prevalence is 0.14, 25% of this value is 0.035, which leads to the interval $[0.105, 0.175]$. Using this method the value for a parameter that represents a probability might be bigger than 1. In this situation, OpenMarkov can do two different things: Ignore the case (i.e., instead of evaluating the ID for $n+1$ values of the prevalence, the number of iterations would be smaller) or Throw an error message.

(c) The third way of specifying each interval is to take a Ratio of the reference value. Thus, for a ratio of 1.1 the limits of the interval for the prevalence would be $0.14/1.1 = 0.1273$ and $0.14 \times 1.1 = 0.154$.

(d) Finally, it is possible to have a User defined interval. For example, we can tell OpenMarkov to vary the prevalence from 0 to 1.

(e) If you have tried any of the alternative options, return to the default: Percentage of the 2nd order probability, with a percentage of 80%.

3. Once OpenMarkov has computed the interval of each parameter, it computes the expected utility for $n+1$ values (of that parameter) evenly distributed in the interval; by default $n = 50$ points. Having m parameters, this requires $m \cdot n$ iterations.

4. The tornado diagram for this example is shown in Figure 6.4. We can see that the parameter having the prevalence of the disease has the highest impact on the expected utility whereas the cost of test E is the parameter having the smallest impact.

5. This dialog also contains the tab Spider diagram—see Figure 6.5. In this diagram the variation for each parameter is shown in the vertical axis; it is the same variation shown in the horizontal axis of the tornado diagram. The advantage of the spider diagram is that it not only shows the absolute variation, but also the sign of the variable. For example, an increase in the prevalence of the disease or in any of the costs leads to a decrease of the expected utility (as expected) while increases in the sensitivity or in the specificity of the test make the expected utility decrease (also as expected; however, for other parameters it may be more difficult to predict the chance of the utility, and in those cases the spider diagram can be more useful than the tornado diagram).

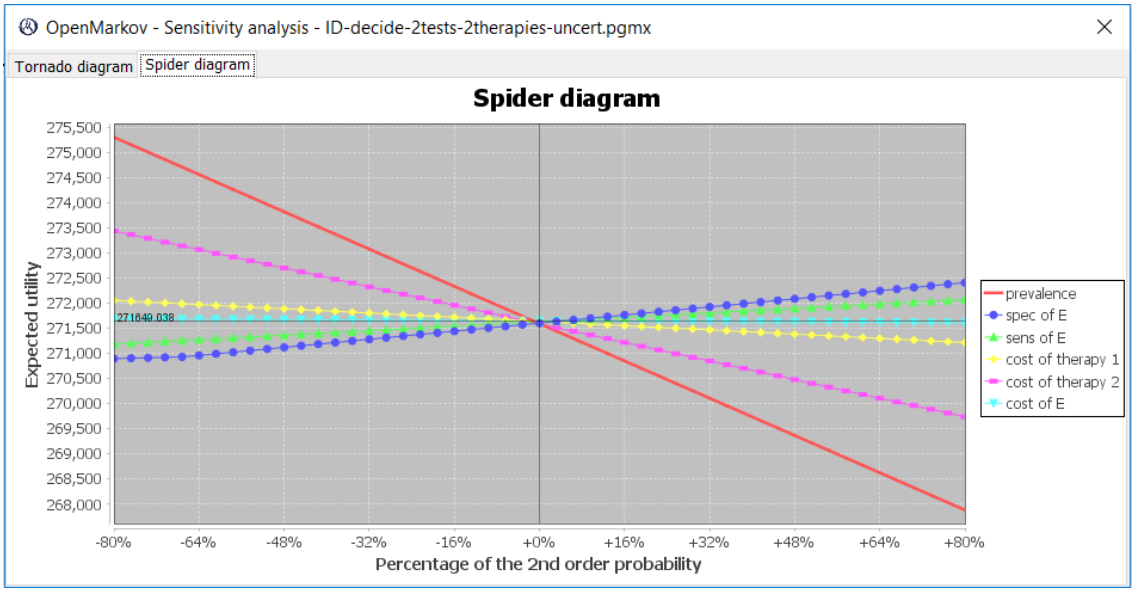


Figure 6.5: A spider diagram for the ID with uncertainty. The vertical axis represents the variation in the expected utility for each parameter (the same variations shown horizontally in Fig. 6.4).

6.2.1.2 Plot (one-way sensitivity analysis)

Another form of deterministic unicriterion analysis consists in showing how the expected utility varies with one of the parameters. For this reason this is sometimes called **one-way sensitivity analysis**.

1. Open again the network `ID-decide-2tests-2therapies-uncert.pgmx`.
2. Click on the Sensitivity analysis icon ( ).
3. In the Inference options dialog, set the Unit to *QALY*, the scale for *cost* to 0 and the scale for *effectiveness* to 1. This means that we will identify the net monetary benefit (NMB) with the effectiveness, disregarding the costs.
4. In the Sensitivity analysis dialog, set the Analysis type to Plot (one-way sensitivity analysis).
5. Check that the parameter selected is *prevalence*.
6. Set the variation in the Horizontal axis parameter to User defined interval, with limits 0 and 1.
7. Set the Scope to *One decision*. Select the Decision *Therapy*. Check that OpenMarkov has selected the scenario in which no test is done.
8. Click OK. You will obtain the plot shown in Figure 6.6.

The interpretation of this plot is as follows. We are considering the scenario in which there is no evidence from the tests because they have not been done; therefore, the probability of the disease coincides with the prevalence. The black horizontal line denotes the expected utility in the reference case, 9.074.

When the prevalence of the disease is 0 we are sure that the disease is absent. In this case, the effectiveness of *no therapy* (red line) is 10, in accordance with Table 3.1; the effectiveness of *therapy 1* (blue line) is 9.9 and that of *therapy 2* (green line) is 9.3. When the prevalence is 1, we are sure that the patient has the disease. Then the effectiveness is 1.2 for *no therapy*, 4.0 for *therapy 1*, and 6.5 for *therapy 2*; the latter is the best choice.

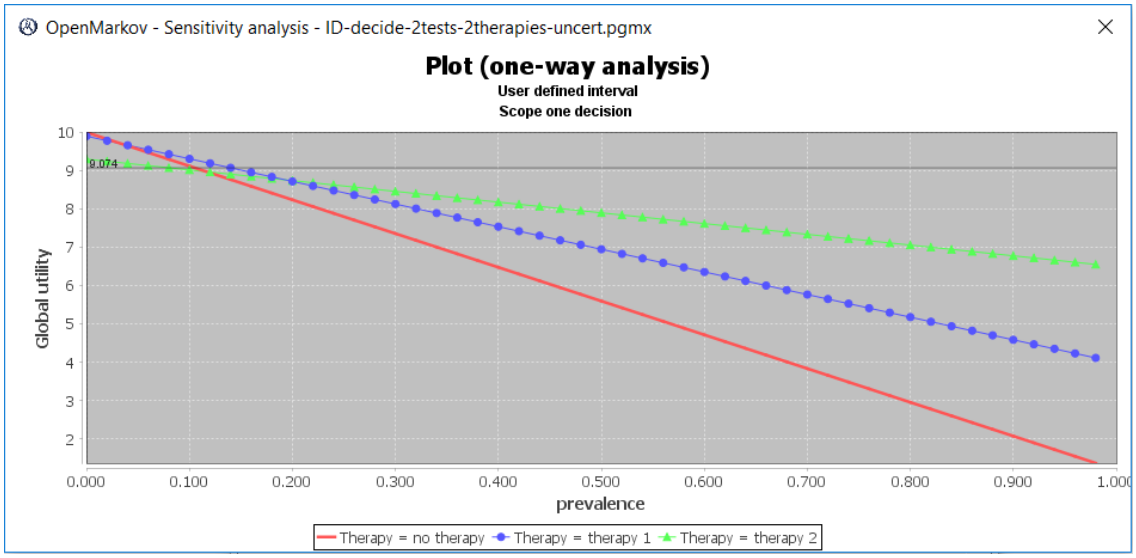


Figure 6.6: A plot (one-way sensitivity analysis) for the decision *Therapy* in the ID with uncertainty when no test has been done.

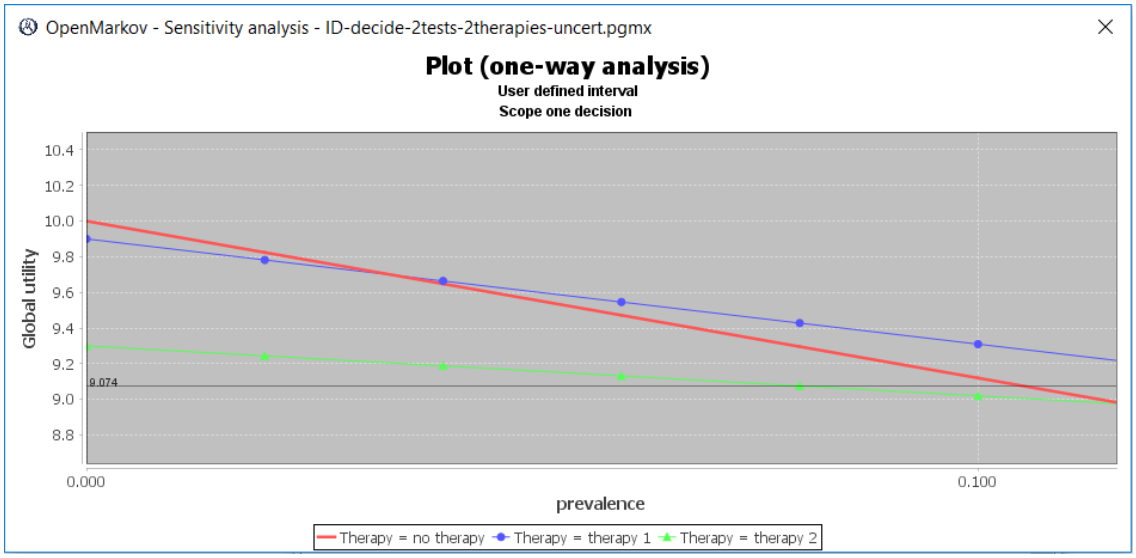


Figure 6.7: A zoom of the above plot around the intersection of the curves for *no therapy* and *therapy 1*.

When the prevalence (or the posterior probability of the disease given the results of the tests) is below 0.345, the optimal policy is not to apply any therapy. OpenMarkov is not yet able to calculate and show this threshold, but we can observe it with more precision by dragging with the mouse a small rectangle around the point of intersection of the red and the blue lines—see Figure 6.7.[3] When the prevalence (or the posterior probability) is above 0.1935, the optimal policy is to apply *therapy 2*. When the prevalence lies between these two thresholds, the optimal policy is *therapy 1*.

After doing this analysis, close the network without saving the changes.

[3]You can fine-tune the information shown in the plot by right-clicking on it and selecting Properties...

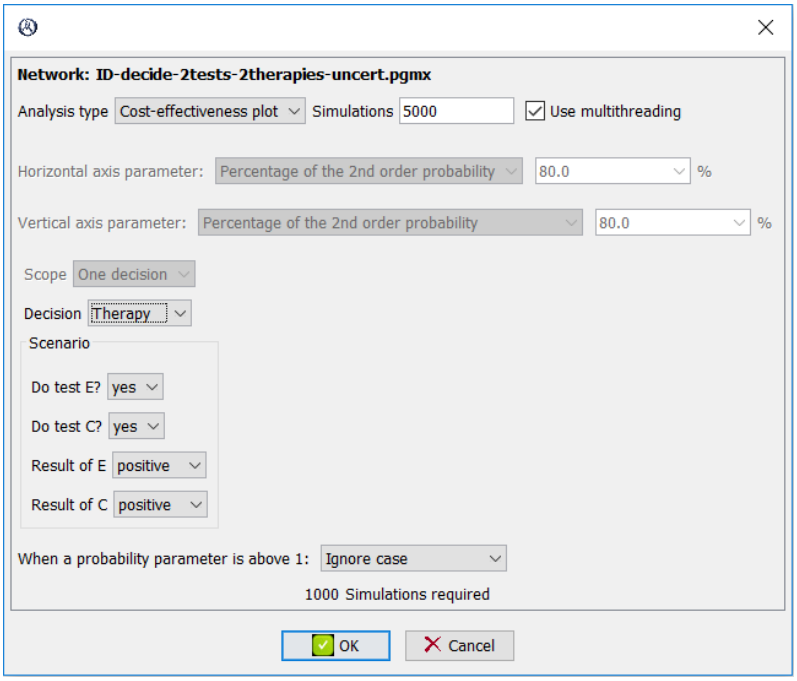


Figure 6.8: Sensitivity analysis dialog for cost-effectiveness analysis. We are telling OpenMarkov to build the scatter plot and the acceptability curve for the scenario in which both tests are positive.

6.2.2 Probabilistic cost-effectiveness sensitivity analysis

OpenMarkov offers two options for probabilistic CEA SA: scatter plots and the acceptability curves. Both of them are based on the same computations, but the presentation of the results is very different.

6.2.2.1 Scatter plot

In order to obtain the scatter plot, follow this steps:

1. Open again the network `ID-decide-2tests-2therapies-uncert.pgmx`.
2. Click on the Sensitivity analysis icon ( ).
3. In the Inference options dialog, select Cost Effectiveness.
4. In the Sensitivity analysis dialog, set the Analysis type to Cost-effectiveness plot.
5. Increase the number of Simulations to 5,000.
6. Select the Decision *Therapy* and the Scenario in which both tests are positive, as shown in Figure 6.8.
7. When OpenMarkov displays the scatter plot, make it relative to *no therapy*, as in Figure 6.9.

The process that OpenMarkov has applied to obtain this plot is the following:

1. Obtain a value for each parameter by sampling it from the second-order distribution of that parameter. If the "distribution" for a parameter is of type *Complement*, adjust it as appropriate. The result is a new ID, slightly different from the original one (which is said to be the *reference case*).

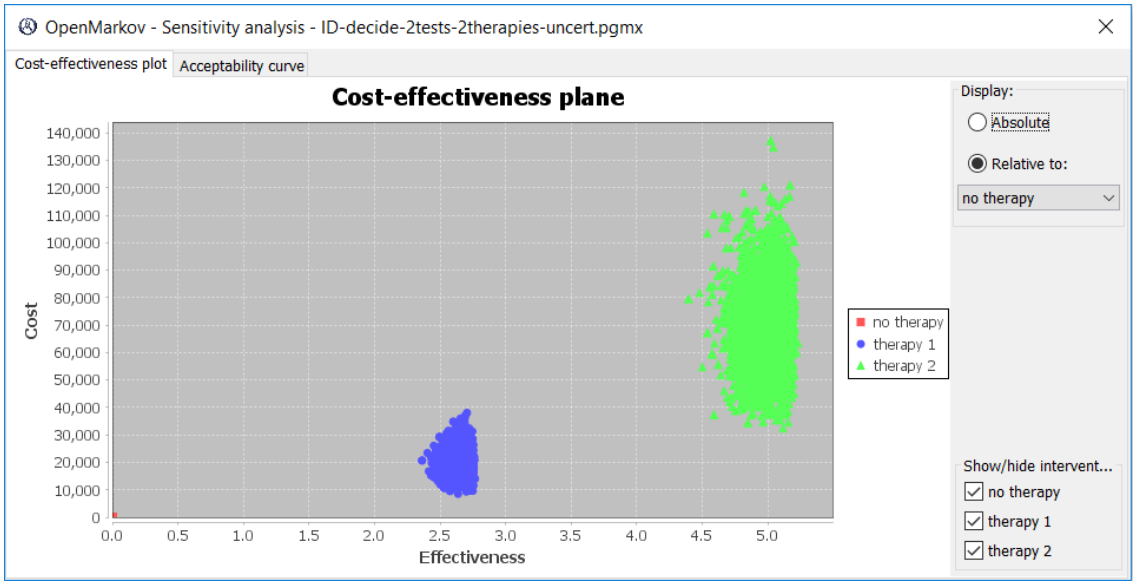


Figure 6.9: A scatter plot for the decision *Therapy* in the ID with uncertainty when both tests are positive. It is essentially the same result as in Figure 5.12 but now there are clouds instead of single points due to the uncertainty.

2. For the scenario under consideration, compute the cost and the effectiveness for each of the states of *Therapy*.

3. Repeat the process 5,000 times, the number of Simulations we indicated.

6.2.2.2 Acceptability curve

Drawing the acceptability curve does not require repeating the above simulations—it suffices to post-process the results as follows:

1. Set the value of λ to 0.

2. For each of the 5,000 simulations,

   (a) compute the NMB of each therapy with the equation given in Section 5.2.1 (remember that we already have the cost and the effectiveness of each therapy in this scenario for the ID sampled).

   (b) find out the optimal therapy, i.e., the one having the higher NMB.

3. Count in how many of the simulations each therapy has been the optimal choice. This way we obtain (an estimate of) the **probability** of each therapy being optimal for this value of λ.

4. Repeat the above steps by increasing the value of λ iteratively.

The acceptability curve built this way is shown in Figure 6.10. Observe that when λ is less than around €7,500/QALY, the optimal intervention is—probably—not to apply any therapy. When λ is above €21,000/QALY (approximately), *therapy 2* is probably the more beneficial. Between these thresholds, the optimal intervention is likely *therapy 1*.

We can also interpret this curves as follows: for a WTP of €30,000/QALY (marked with a vertical black line in Fig. 6.10 because it is our reference value for λ), there is 90% probability of *therapy 2* being the optimal treatment; the probability of *therapy 1* being optimal is only 10%; the probability of *no therapy* being optimal is virtually 0. The reference value for λ can be changed

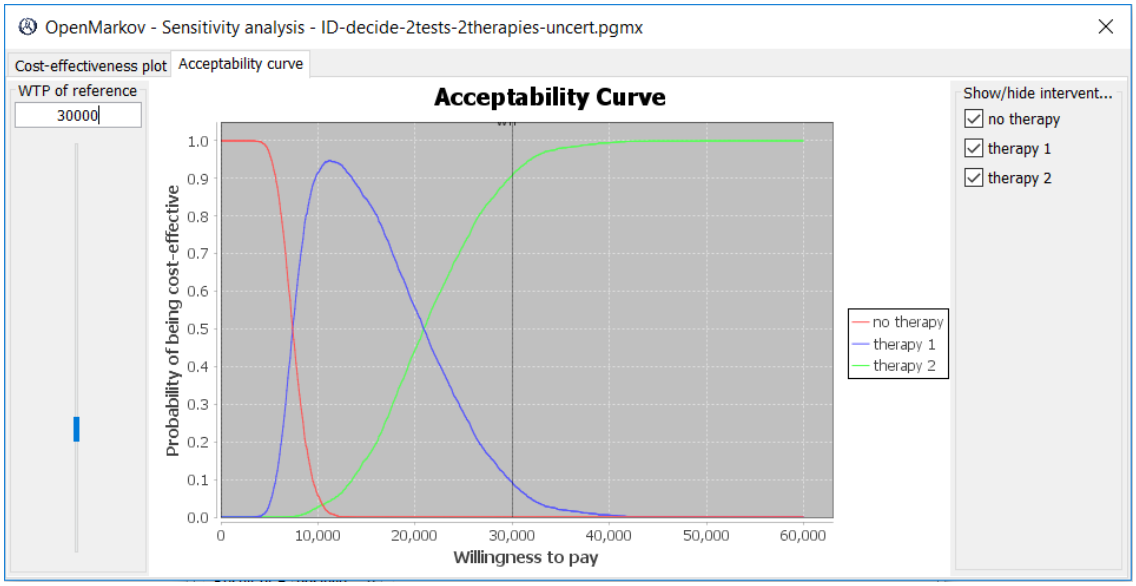


Figure 6.10: Acceptability curve for the decision *Therapy* in the ID with uncertainty when both tests are positive. It is built with the same simulations as the scatter plot (Fig. 6.9).

with the box and the slide bar at the left of the dialog; this moves the vertical bar and the changes the limits of the horizontal axis, but the curve itself is the same.

Chapter 7

Markov influence diagrams

In many real-world problems it is necessary to model the evolution of the system over time. In this chapter we introduce Markov influence diagrams (MIDs) as a formalism for building temporal models [13]. In MIDs time is discretized into cycles, i.e., intervals of equal duration; therefore the **cycle length** is one of the basic properties of a Markov model. The main difference with IDs is that MIDs may contain two types of nodes: temporal and atemporal. The latter represent properties that do not change over time, such as the sex of the patient, while temporal nodes represent the properties that evolve (for instance, the state of the patient, the presence of symptoms, or the result of a test performed in every cycle) and the events that may occur, such as a medical complication of the patient's death.

In this chapter we will explain how to build MIDs and how to evaluate them, more specifically, how to perform cost-effectiveness analysis and sensitivity analysis. We will use as an example the well-known model by Chancellor et al. [7] for HIV/AIDS. This model has become is obsolete for clinical practice, but is still very useful for didactic purposes. In this chapter we implement it as a MID, following the Excel version of this model available at the web page for the book by Briggs et al. [5].[1]

Example 7.1. The purpose of this model is to compare two treatments for HIV infection. One of them (monotherapy) administers zidovudine all the time; the other (combined therapy) initially applies both zidovudine (AZT) and lamivudine but after two years, when the latter becomes ineffective, continues with zidovudine alone. The clinical condition of the patient is characterized by four states, *A*, *B*, *C*, and *D*, of increasing severity, where *state D* represents the death.

The model used a **cycle length** of one year. The **transition probabilities**, assumed to be time-independent, were obtained from a cohort of patients, shown in Table 7.1. Notice that, according to this table, no patient regresses to less severe states, and that a patient who dies remains in state D for ever. The effect of lamivudine consists in reducing the transition probabilities; the relative risk (with respect to AZT alone) is 0.509.

The annual cost of the drugs is £ 2,278 for AZT and £ 2,086.50 for lamivudine; these values are known with precision. Additionally, there are other costs of care, depending on the the patient's state, as shown in Table 7.2; in the sensitivity analysis we will assign a Gamma with $k = 1$ to each of these parameters.

In [7] and in [5] the model was evaluated for a **time horizon** of 20 cycles. Costs were discounted at 6% annually but effectiveness was not discounted.

[1] `https://www.herc.ox.ac.uk/downloads/supporting-material-for-decision-modelling-for-health-economic-evaluation/ex25sol.xls`.

↓posterior | origin →	state A	state B	state C
state A	1,251	—	—
state B	350	731	—
state C	116	512	1,312
death	17	15	437
Total	1,734	1,258	1,749

Table 7.1: Cohort of patients from which the annual transition probabilities were obtained.

Cost	state A	state B	state C
direct medical costs	£ 1,701	£ 1.774	£ 6,948
community care costs	£ 1,055	£ 1,278	£ 2,059

Table 7.2: Annual costs per states (in addition to the drugs).

7.1 Construction of the MID

7.1.1 Creation of the network

1. Select File ▷ New or click on the icon Create a new network (🗋) in the first toolbar.

2. Right-click on the network panel, which is still empty, and in the Network properties dialog set the Network type to *MID*. Click OK.

3. Open again the Network properties dialog go the Advanced tab, click the buttons Decision criteria and Standard criteria, and select *Cost-effectiveness (£/QALY)*, but then change the Unit for *Effectiveness* from *QALY* (quality-adjusted life year) to *LY* (life year) because in this model we only take into account life duration, not quality of life. Click OK.

7.1.2 Construction of the graph

In order to obtain a graph like the one in Figure 7.1, follow these steps.

1. Click on the icon Insert decision nodes (□). Double-click on the network panel to create a new node; in the Node properties dialog, set the Name to *Therapy choice*. Go to the Domain tab and set the states of the variable to *combined therapy* and *monotherapy*. Click OK.

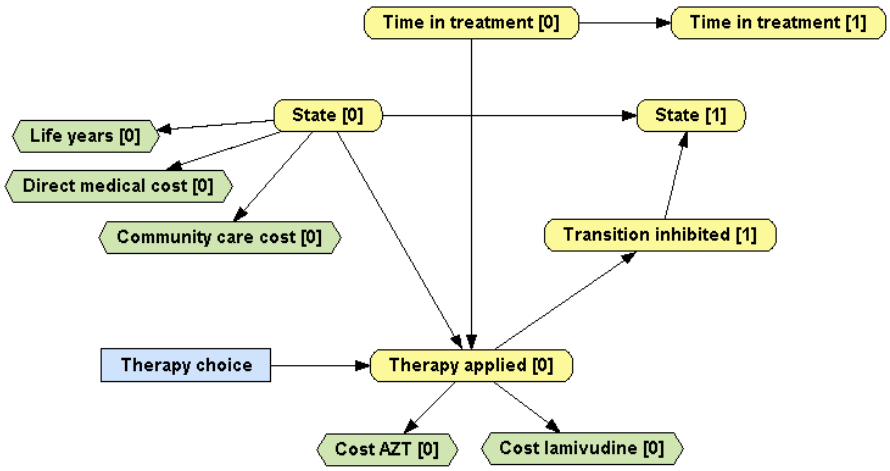


Figure 7.1: MID for Chancellor's HIV model

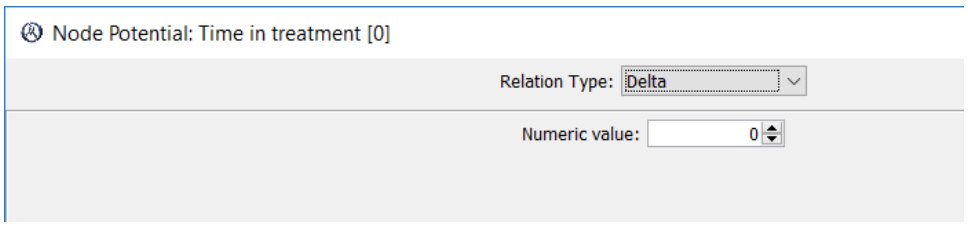


Figure 7.2: Probability distribution for *State [0]*.

2. Click on the icon Insert chance node (). Double-click on the network panel to create a new node. Name it *Time in treatment* and set its Time slice to *0*. In the Domain tab set the Variable type to *Numeric*, the Precision to *1*, the Unit to *year*, and the interval to $[0, \infty)$. Click OK.

3. Right-click on this node and select Create node in next slice to create the node *Time in treatment [1]*. Select the Insert link tool () and drag the mouse from *Time in treatment [0]* to *Time in treatment [1]* to create a link between these two nodes.

4. Create the chance node *State [0]*, i.e., create a node named *State* and set its Time slice to *0*. Go to the Domain tab and set its states to *state A* (upper position), *state B*, *state C*, and *death* (lower position). Click OK.

5. Right-click on this node and select Create node in next slice to create the node *State [1]*.

6. Create the chance node *Therapy applied [0]* and set its states to *combined therapy*, *monotherapy*, and *none*. Draw links from *Therapy choice*, *State [0]*, and *Time in treatment [1]* to *Therapy applied [0]*.

7. Create the chance node *Transition inhibited [1]*, with states *yes* and *no* (you can use the Standard domains button). Draw a link from *Therapy applied [0]* to *Transition inhibited [1]*, and from this node to *State [1]*.

8. Click on the icon Insert value node () and create the node *Life years [0]* and set its Decision criterion to *Effectiveness*. Draw a link from *State [0]* to *Life years [0]* because the amount of life accrued in a cycle depends (only) on the patient's state.

9. Create the nodes *Direct medical costs [0]*, *Community care cost [0]*, *Cost AZT [0]*, and *Cost lamivudine [0]*. Before closing the Node properties dialog for each node, check that the Decision criterion assigned to is *Cost*. Draw the links shown in Figure 7.1.

10. Save the network as `MID-Chancellor.pgmx`.

7.1.3 Specification of the parameters

After introducing the structural information of the MID, it is time to set the numerical parameters. We might introduce first the parameters without uncertainty and then specify the second-order distributions. However, in order to save effort, we will introduce the uncertainty from the beginning.

7.1.3.1 Encoding the probabilities

1. When the patient enters the model, the value of *Time in treatment [0]* is 0, by definition. Therefore, the probability distribution for this node must be a Dirac delta centered at 0, as shown in Figure 7.2.

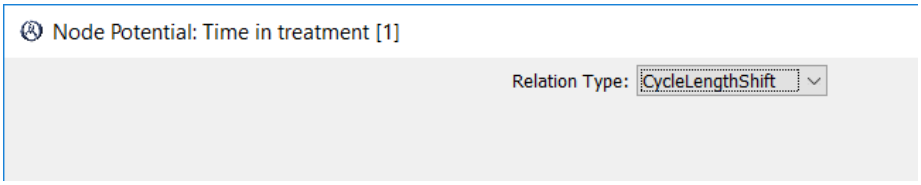


Figure 7.3: Probability distribution for *State [0]*.

Node Potential: State [0]

Relation Type: Table

state A	1
state B	0
state C	0
dead	0

Figure 7.4: Probability distribution for *State [0]*.

2. The value of the variable *Time in treatment [i+1]* will be the value of *Time in treatment [i]* plus the cycle length. During the evaluation of the model, the value of *Time in treatment [0]* will 0 years, the value *Time in treatment [1]* 1 year, and so forth. If the cycle length were half a year, the increase would be 0.5, etc.We encode this information by setting the Relation type of the probability of this node to *CycleLengthShift*, as shown in Figure 7.3.

3. We assume that initially all patients are in *state A*. Therefore, the conditional probability table for *State [0]* must be as in Figure 7.4.

4. We will encode the conditional probability for *State [1]* as the table shown in Figure 7.5. You may need to reorder the parents of this node with the Reorder variables button.

   When *Transition inhibited [1] = yes*, the patient remains in the same state as in the previous cycle, i.e., *State [0]*. For this reason there is a diagonal of 1's in the right part of the table.

   Let us now focus on the columns for *Transition inhibited [1] = no*. When the patient is dead, he remains in the same state even if the treatment would have not inhibited the transition; this explains the values of the first column.

   We can see in Table 7.1 that there were 1,734 patients in state A. One year later 17 of them have died, 350 were in state B, 116 in state C, and 1,251 had remained in state A. This

Node Potential: State [1]

Relation Type: Table Reorder variables

Transition inhibited [1]	no	no	no	no	yes	yes	yes	yes
State [0]	death	state C	state B	state A	death	state C	state B	state A
state A	0	0	0	0.721453	0	0	0	1
state B	0	0	0.581081	0.201845	0	0	1	0
state C	0	0.750143	0.406995	0.066897	0	1	0	0
death	1	0.249857	0.011924	0.009804	1	0	0	0

Figure 7.5: Conditional probability for *State [1]*, represented as a tree.

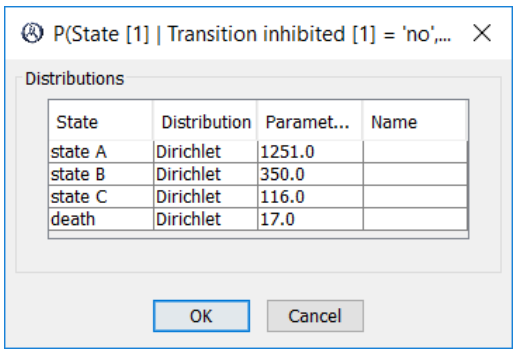


Figure 7.6: Second-order probability distribution for *State [1]* when *Transition inhibited [1]* = *no* and *State [0]* = *state A*.

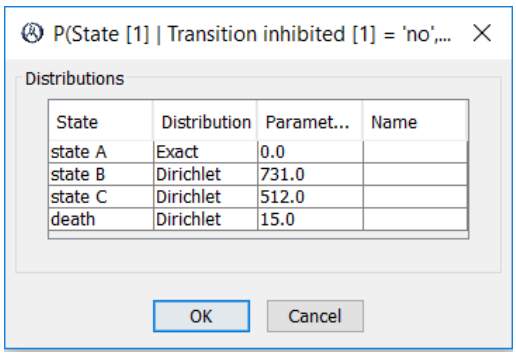


Figure 7.7: Second-order probability distribution for *State [1]* when *Transition inhibited [1]* = *no* and *State [0]* = *state B*.

information can be introduced in the potential for *State [1]* by right-clicking on any cell in the column for *state A*, selecting Assign uncertainty, and setting the second-order Distribution of each parameter to *Dirichlet*, with the values shown in Figure 7.6.

When the patient was in state B in cycle 0, the second-order probability distribution is the one shown in Figure 7.7, whose parameters were obtained from Table 7.1. Please note that in this case there is no uncertainty for *State [1]* = *state A*.

We can also observe in Table 7.1 that 1,312 out of the 1,749 patients in state C remained in that state one year later, while the rest have died. We can encode this information using a Dirichlet distribution with two parameters, but we can also use a Beta (which is a particular case of the Dirichlet), as shown in in Figure 7.8.

We must introduce now the conditional probability for the node *Transition inhibited [1]*, which represents the probability that lamivudine prevents the patient from transitioning to a more severe state. Chancellor et al. [7], following [29], considered that the relative risk of combined therapy with respect to monotherapy was 0.509. This figure can be taken as the probability that the transition occurs even when lamivudine is applied. In order to obtain approximately the same confidence intervals as in [29], we assigned to this parameter the Beta distribution shown in Figure 7.9. Don't forget to assign a name to this parameter if you wish to be able to perform sensitivity analysis on it. [2]

[2]If the relative risk were 1, it would mean that lamivudine would be unable to inhibit the transition; i.e., the probability of inhibition would be 0. If the relative risk were 0, it would mean that the probability of inhibition would be 1. This interpretation of the relative risk as a probability clearly shows that the uncertainty about

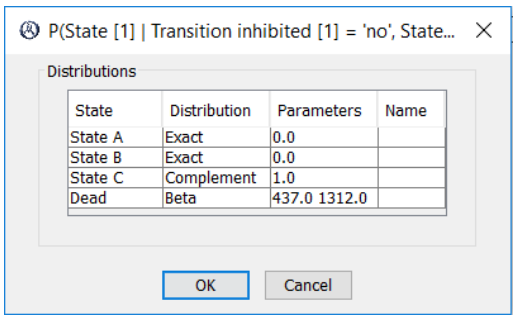


Figure 7.8: Second-order probability distribution for *State [1]* when *Transition inhibited [1]* = *no* and *State [0]* = *state C*.

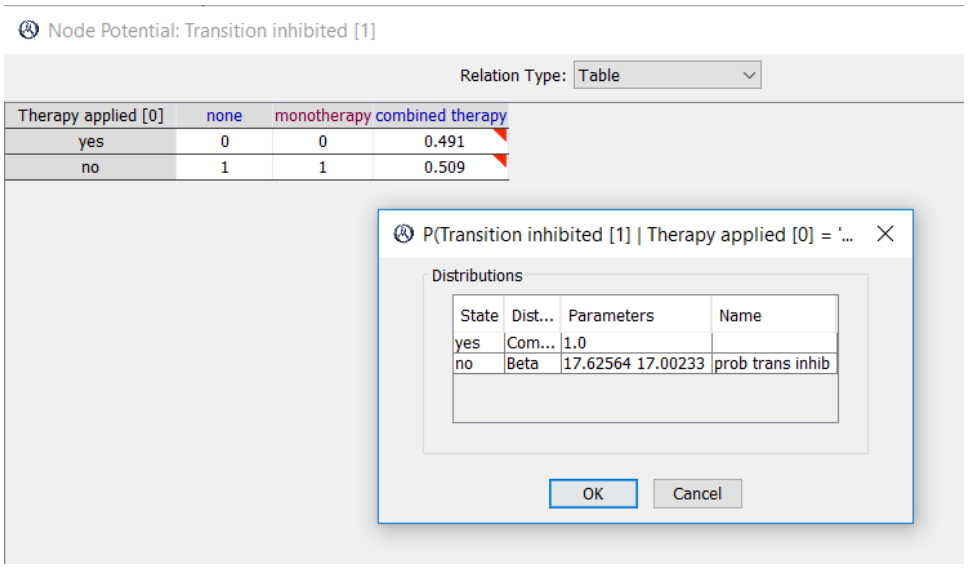


Figure 7.9: Conditional probability for *Transition inhibited [1]*.

5. Finally, edit the conditional probability of *Therapy applied [0]*.[3] Set its Relation type to *Tree/ADD*. If the root of the tree is not *State [0]*, right-click on that node, select Change variable, and choose *State [0]*, as in Figure 7.10.

   Click on the node *State [0]* = *state A*, select Add states to branch, and check the boxes for *state B* and *state C*. Right-click again on that node, choose Add subtree, and select *Therapy choice*.

   Right-click on the node *Therapy choice* = *combined therapy*, select Add subtree, and choose *Time in treatment [0]*. It will create a subtree with a single branch that covers the whole interval $[0, +\infty)$. Right-click on the box containing that interval, select Split interval and set the Threshold value to *2*; notice that the radio button Included in the first interval is checked, which means that the threshold will be included in the first sub-interval, i.e., the interval $[0, +\infty)$ is partitioned into $[0, 2]$ and $(2, +\infty)$. If we had selected the option Included in the second interval for the threshold, the partition would consist of $[0, 2)$ and $[2, +\infty)$.

   Right-click on the node $P($*Therapy applied [0]*$)$ for the interval $[0, 2]$ and select Edit potential. Set the Relation type to *Delta* and the State to *combined therapy*. Click OK.

this parameter should not be modeled with a log-normal distribution—as it was done in the Excel version cited above—because it would lead to transition probabilities below 0 or above 1. For this reason we decided to model the uncertain about this parameters with a Beta distribution.

[3]Given that it is common to make irreversible mistakes during the edition of trees, we recommend you to save the network now, so that you don't lose the changes made so far.

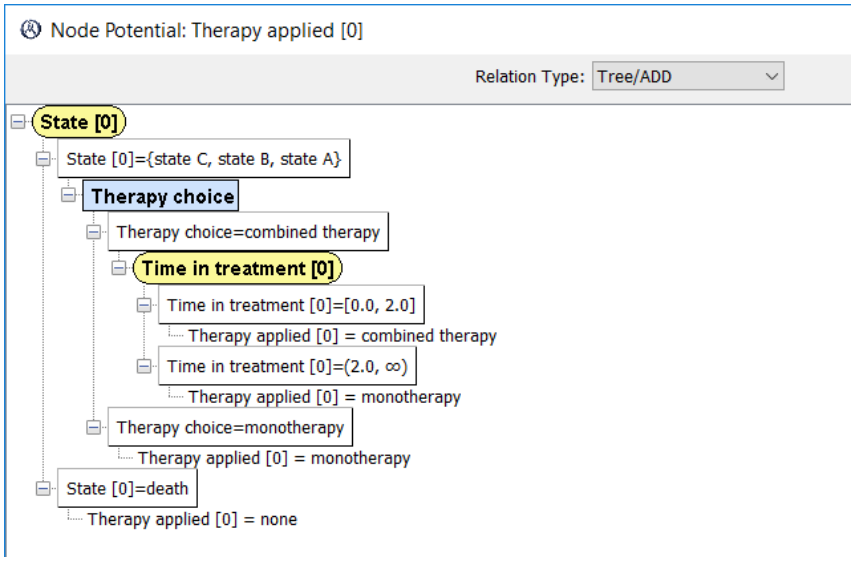


Figure 7.10: Conditional probability for *Therapy applied [0]*.

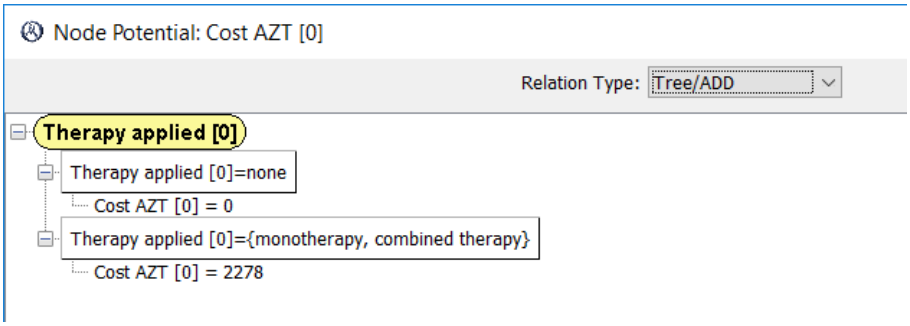


Figure 7.11: Utility function for *Cost AZT [0]*.

Analogously, set the value of *Therapy applied [0]* for the other branches to *monotherapy* or *none*, as in Figure 7.10.

6. Click OK and save the network again.

Notice that the potentials for *Time in treatment [0]*, *State [0]*, *State [1]* and *Therapy applied [0]* are deterministic, which means that the values are known with certainty—at least when knowing the values (the configuration) of its parents; put another way, the conditional probability distributions for these nodes are degenerate, i.e., the probability is 0 for all the values except for one. Therefore, it does not make sense to assign uncertainty to them. In contrast, the potentials for *Transition inhibited [1]* and *State [1]* strictly probabilistic, i.e., some of their probabilities lie inside the interval $(0, 1)$, and it does make sense to assign them second-order probability distributions to represent the uncertainty.

7.1.3.2 Encoding the utilities

We must introduce now the utilities, first the cost and then the effectiveness.

1. The cost of AZT can be encoded as in Figure 7.11. In this case, given that we know with precision the cost, we might have used a table, writing the value *2,278* twice—for *monotherapy* and for *combined therapy*—but it is better to use a tree so that this parameter is encoded only once in the model.

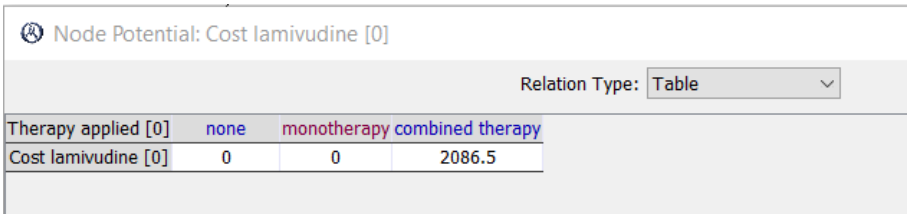


Figure 7.12: Utility function for *Cost lamivudine [0]*.

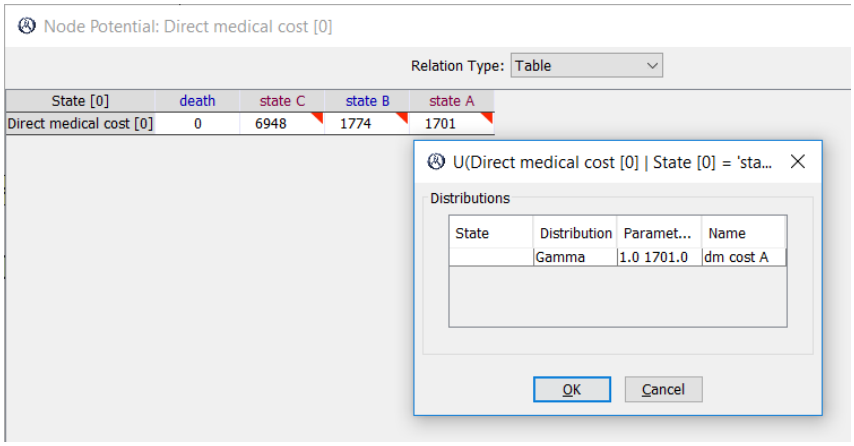


Figure 7.13: Utility function for *Direct medical costs [0]*.

2. The cost of lamivudine can be encoded as in Figure 7.12. In this case it is not worth using a tree.

3. The utility for *Direct medical costs [0]* can be encoded as in Figure 7.13, taking the values from Table 7.2 and assigning a Gamma distribution with $k = 1$ to each state.[4]

4. The utility for *Community care cost [0]* is entered in the same way.

We introduce now the effectiveness, which in this model is identified with life duration. When the patient is alive in one cycle, he/she accrues one life year. Therefore the effectiveness can be easily encoded as in Figure 7.14

7.2 Evaluation of the MID

7.2.1 Inference options

Once we have built the model, we evaluate it. First, we have to set up the inference options, as shown in Figure 7.15, where we can observe new options specific for Markov models:

- the number of cycles, i.e., **the time horizon**;
- whether transitions occur at the beginning of the cycle or at the end of the cycle, or whether a **half-cycle correction** must be applied (this is likely to change in future versions of OpenMarkov);

[4]In a Gamma distribution, the mean is $k\theta$ and the standard deviation $\sqrt{k}\theta$. Therefore, when $k = 1$ the mean and the standard deviation coincide with θ. We have taken this value of k in order to reproduce as closely as possible the Excel model. However, we think it would be more reasonable to make k at least equal to 2. Please note that if the standard deviation is assumed to be 20% of the mean (as we did in Section 6.1, following what is common in the literature), then $k = 25$.

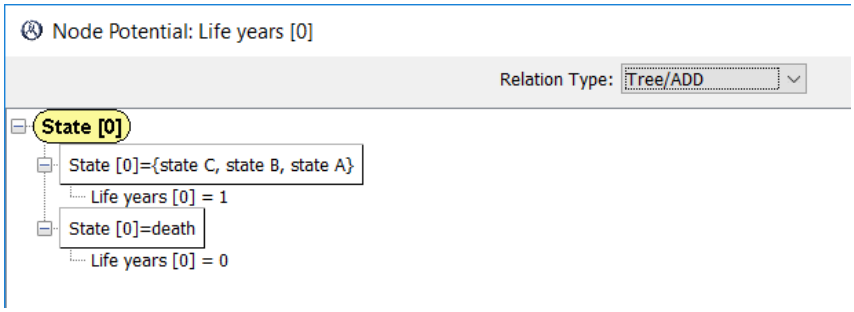


Figure 7.14: Utility function for the effectiveness, *Life years [0]*.

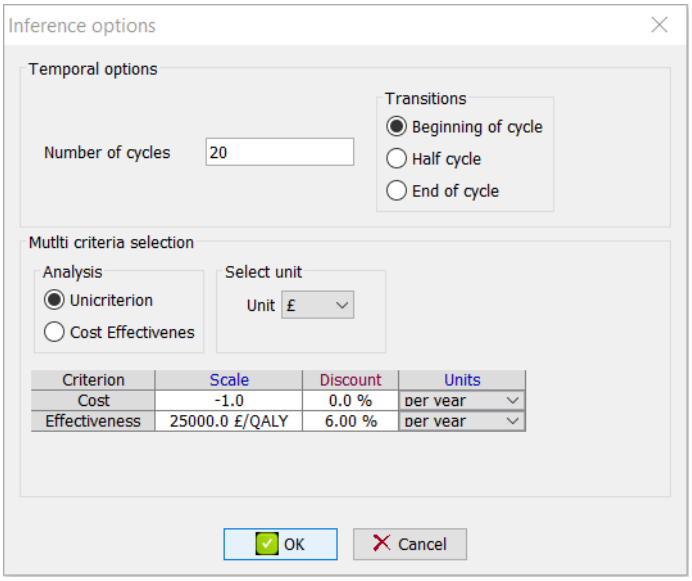


Figure 7.15: Inference options for Markov models.

- a **discount** for each criterion; in this model it is indifferent to apply the discount *per cycle* or *per year* because the cycle length is one year.

7.2.2 Evaluation facilities common to non-temporal models

OpenMarkov evaluates a Markov model by expanding the network to the desired horizon and applying the discounts, as specified in the inference options dialog. Most options for non-temporal influence diagrams are also available for MIDs:

- finding the optimal strategy,
- doing cost-effectiveness analysis, both globally and for one decision,
- performing several types of sensitivity analysis: tornado and spider diagrams, plots (one-way sensitivity analysis), scatter plots, acceptability curves, etc.

We recommend you to try each one of them for the above MID.

Other options, such as introducing evidence, imposing policies, showing the expected utilities, propagating evidence, etc., will be available in future versions of OpenMarkov, but only for non-temporal variables.

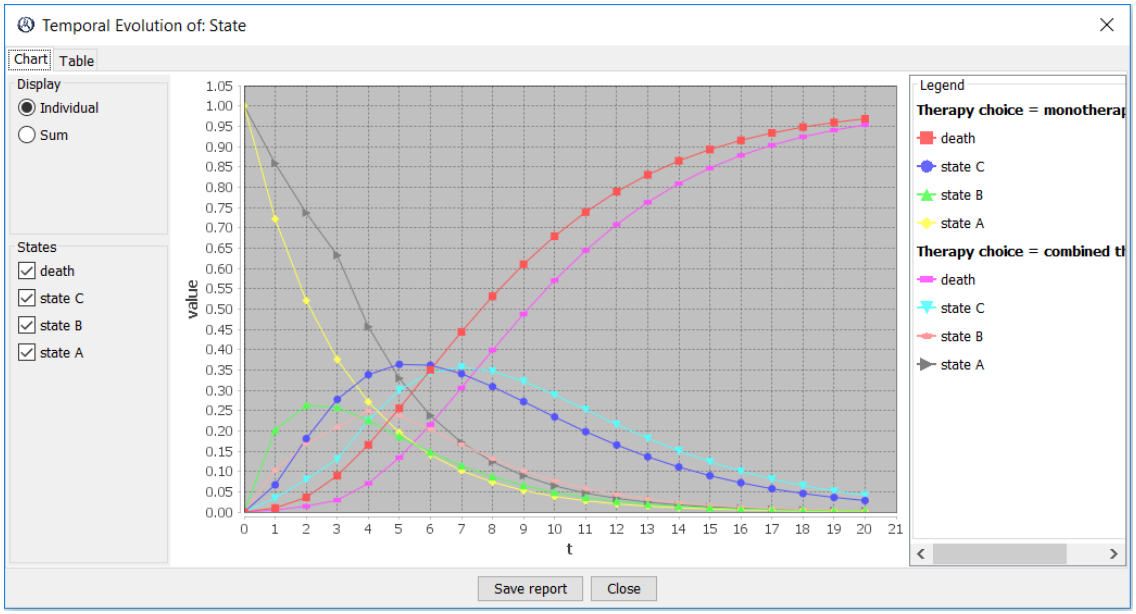


Figure 7.16: Evolution of the probability of the states of variable *State*. It is possible to uncheck some boxes to see the curves for only some states.

7.2.3 Temporal evolution of variables

In OpenMarkov it is possible to show the evolution of temporal variables.

1. Right-click on *State [0]* and select Temporal evolution.
2. In the next dialog, accept the default values.
3. You will obtain the plot in Figure 7.16. The Table tab shows the results in tabular format and the Save report button allows you to export them to Excel.
4. Uncheck the box for *death* and select the radio button *Sum*. This will show the sum of the probabilities for states A, B, and C, i.e., the probability that the patient is alive at each cycle—Fig. 7.17.

A similar analysis can be performed for the nodes that represent cost or effectiveness.

1. Right-click now on *Cost AZT [0]*, select Temporal evolution, and accept the default values of the next dialog.
2. Click the *Cumulative* radio button. The result will be as in Figure 7.18.

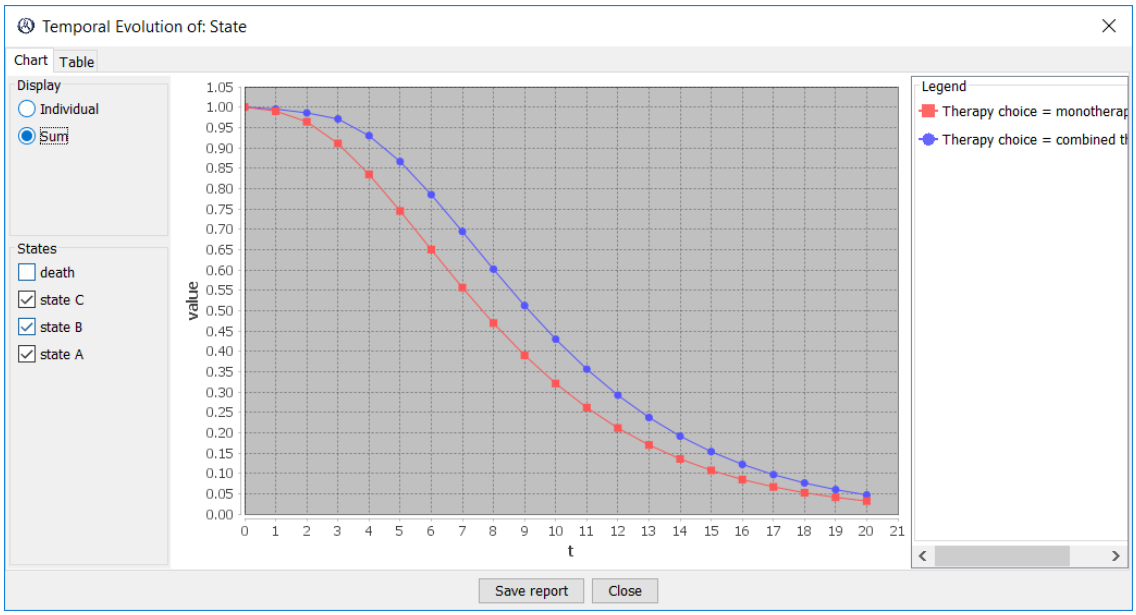


Figure 7.17: Probability of the patient being alive at each cycle. It is the sum of the probabilities for all the states except *death*.

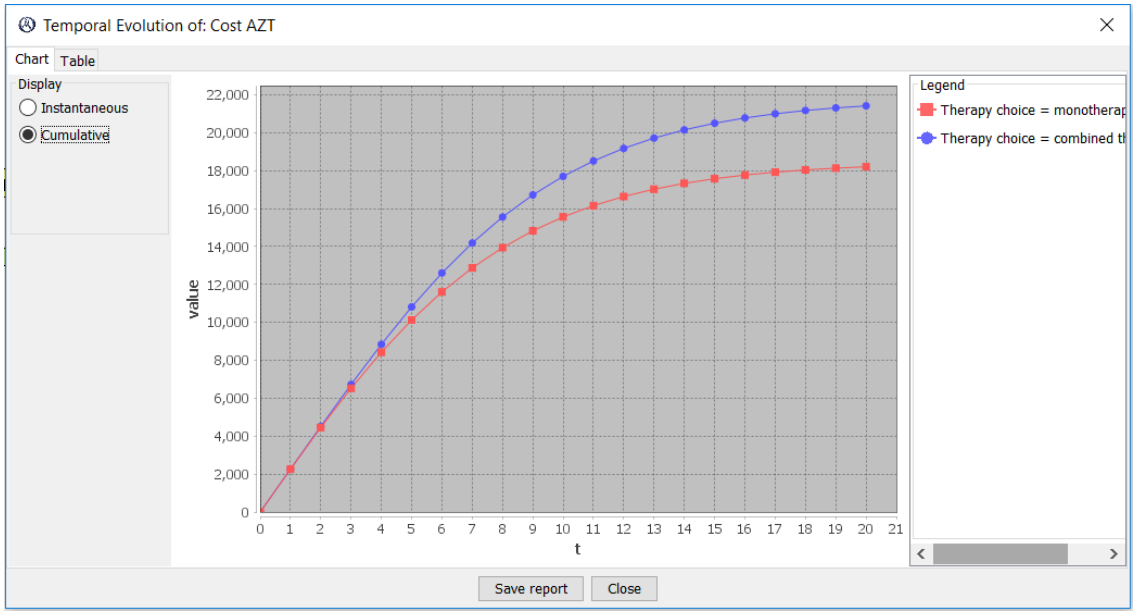


Figure 7.18: Cumulative cost of AZT. The curve grows more and more slowly because patients die progressively, as shown in Figure 7.17.

Acknowledgment

This work has been supported by the Spanish Ministry of Education and Science under grants TIN2006-11152 and TIN2009-09158, the Health Institute Carlos III of the Spanish Government under grant PI13/02446. and the Ministry of Economy under grant TIN2016-77206-R, and the European Regional Development Fund (ERDF).

Bibliography

[1] Arias, M. and Díez, F. J. (2015). Cost-effectiveness analysis with influence diagrams. *Methods of Information in Medicine*, 54:353–358.

[2] Arias, M., Díez, F. J., and Palacios, M. P. (2011). ProbModelXML. a format for encoding probabilistic graphical models. Technical Report CISIAD-11-02, UNED, Madrid, Spain.

[3] Beinlich, I. A., Suermondt, H. J., Chávez, R. M., and Cooper, G. F. (1989). The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. In *Proceedings of the 2nd European Conference on AI and Medicine*, pages 247–256, London. Springer-Verlag, Berlin.

[4] Bermejo, I., Oliva, J., Díez, F. J., and Arias, M. (2012). Interactive learning of Bayesian networks with OpenMarkov. In [6], pages 27–34.

[5] Briggs, A., Claxton, K., and Sculpher, M. (2006). *Decision Modelling for Health Economic Evaluation*. Oxford University Press, New York.

[6] Cano, A., Gómez, M., and Nielsen, T. D., editors (2012). *Proceedings of the Sixth European Workshop on Probabilistic Graphical Models (PGM'12)*, Granada, Spain.

[7] Chancellor, J. V., Hill, A. M., Sabin, C. A., Simpson, K. N., and Youle, M. (1997). Modelling the cost effectiveness of lamivudine/zidovudine combination therapy in hiv infection. *Pharmacoeconomics*, 12:54–66.

[8] Darwiche, A. (2009). *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, New York.

[9] Díez, F. J. (2007). *Introducción a los modelos gráficos probabilistas*. UNED, Madrid.

[10] Díez, F. J. and Druzdzel, M. J. (2006). Canonical probabilistic models for knowledge engineering. Technical Report CISIAD-06-01, UNED, Madrid, Spain.

[11] Díez, F. J., Luque, M., and König, C. (2012). Decision analysis networks. In [6], pages 83–90.

[12] Díez, F. J., Luque, M., König, C., and Bermejo, I. (2014). Decision analysis networks. Technical Report CISIAD-14-01, UNED, Madrid, Spain.

[13] Díez, F. J., Yebra, M., Bermejo, I., Palacios-Alonso, M. A., Arias, M., Luque, M., and Pérez-Martín, J. (2017). Markov influence diagrams: A graphical tool for cost-effectiveness analysis. *Medical Decision Making*, 37:183–195.

[14] Ezawa, K. (1998). Evidence propagation and value of evidence on influence diagrams. *Operations Research*, 46:73–83.

[15] Herskovits, E. (1990). *Computer-based probabilistic-network construction*. PhD thesis, Dept. Computer Science, Stanford University, STAN–CS–90–1367.

[16] Howard, R. A. and Matheson, J. E. (1984). Influence diagrams. In Howard, R. A. and Matheson, J. E., editors, *Readings on the Principles and Applications of Decision Analysis*, pages 719–762. Strategic Decisions Group, Menlo Park, CA.

[17] Howard, R. A. and Matheson, J. E. (2005). Influence diagrams. *Decision Analysis*, 2:127–143.

[18] Kjærulff, U. and Madsen, A. L. (2010). *Bayesian Networks and Influence Diagrams: A Guide to Construction and Analysis*. Springer, New York.

[19] Koller, D. and Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, Cambridge, MA.

[20] Korb, K. B. and Nicholson, A. E. (2010). *Bayesian Artificial Intelligence*. CRC Press, second edition.

[21] Lacave, C., Luque, M., and Díez, F. J. (2007). Explanation of Bayesian networks and influence diagrams in Elvira. *IEEE Transactions on Systems, Man and Cybernetics—Part B: Cybernetics*, 37:952–965.

[22] Lauritzen, S. L. and Spiegelhalter, D. J. (1988). Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society, Series B*, 50:157–224.

[23] Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press.

[24] Neapolitan, R. E. (2004). *Learning Bayesian Networks*. Prentice-Hall, Upper Saddle River, NJ.

[25] Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA.

[26] Raiffa, H. (1968). *Decision Analysis. Introductory Lectures on Choices under Uncertainty*. Addison-Wesley, Reading, MA.

[27] Raiffa, H. and Schlaifer, R. (1961). *Applied Statistical Decision Theory*. John Wiley & Sons, Cambridge, MA.

[28] Spirtes, P., Glymour, C., and Scheines, R. (2000). *Causation, Prediction and Search*. The MIT Press, Cambridge, Massachusetts, second edition.

[29] Staszewski, S., Hill, A. M., Bartlett, J., Eron, J. J., et al. (1997). Reductions in HIV-1 disease progression for zidovudine/lamivudine relative to control treatments: a meta-analysis of controlled trials. *AIDS*, 11:477–483.

[30] Stinnett, A. A. and Mullahy, J. (1998). Net health benefit: A new framework for the analysis of uncertainty in cost-effectiveness analysis. *Medical Decision Making*, 18:S68–S80.