



# OPENMARKOV TUTORIAL

[www.openmarkov.org](http://www.openmarkov.org)

Version 0.4.0

June 9, 2021

CISIAD

UNED



# Contents

<b>Introduction</b>	<b>1</b>
<b>1 Bayesian networks: edition and inference</b>	<b>3</b>
1.1 Overview of OpenMarkov's GUI . . . . .	3
1.2 Editing a Bayesian network . . . . .	4
1.2.1 Creation of the network . . . . .	4
1.2.2 Structure of the network (graph) . . . . .	4
1.2.3 Saving the network . . . . .	4
1.2.4 Selecting and moving nodes . . . . .	4
1.2.5 Conditional probabilities . . . . .	5
1.3 Inference . . . . .	5
1.3.1 Entering findings . . . . .	6
1.3.2 Comparing several evidence cases . . . . .	6
1.4 Canonical models . . . . .	7
<b>2 Learning Bayesian networks from data</b>	<b>9</b>
2.1 Introduction . . . . .	9
2.2 Basic learning options . . . . .	9
2.2.1 Automatic learning . . . . .	9
2.2.2 Positioning the nodes with a model network . . . . .	11
2.2.3 Discretization and selection of variables . . . . .	11
2.2.4 Treatment of missing values . . . . .	14
2.3 Interactive learning . . . . .	14
2.3.1 Learning with the hill climbing algorithm . . . . .	14
2.3.2 Imposing links with a model network . . . . .	16
2.3.3 Learning with the PC algorithm . . . . .	17
2.4 Exercises . . . . .	19
<b>3 Decision analysis networks</b>	<b>21</b>
3.1 An example with one decision . . . . .	21
3.1.1 Editing the decision analysis network . . . . .	21
3.1.2 Equivalent decision tree . . . . .	23
3.1.3 Optimal strategy . . . . .	24
3.1.4 Order of the variables in the decision tree and the strategy tree . . . . .	24
3.2 An example with two decisions . . . . .	25
<b>4 Influence diagrams</b>	<b>29</b>
4.1 An example with one decision . . . . .	29
4.1.1 Editing the ID . . . . .	29
4.1.2 Resolution, optimal strategy and maximum expected utility . . . . .	30
4.2 An example with two decisions . . . . .	31
4.2.1 Editing the influence diagram . . . . .	32
4.2.2 Resolution of the influence diagram . . . . .	33
4.3 Introducing evidence in influence diagrams* . . . . .	36
4.3.1 Post-resolution findings . . . . .	36

4.3.2	Pre-resolution findings . . . . .	36
4.4	Imposing policies* . . . . .	38
<b>5</b>	<b>Multicriteria analysis with DANs and IDs</b>	<b>41</b>
5.1	Construction of multicriteria networks . . . . .	41
5.2	Analysis of multicriteria networks . . . . .	43
5.2.1	Unicriterion analysis . . . . .	43
5.2.2	Cost-effectiveness analysis . . . . .	44
5.2.2.1	Global cost-effectiveness analysis . . . . .	44
5.2.2.2	Cost-effectiveness analysis analysis for a decision . . . . .	46
<b>6</b>	<b>Cost-effectiveness with Markov influence diagrams</b>	<b>49</b>
6.1	Construction of the MID . . . . .	50
6.1.1	Creation of the network . . . . .	50
6.1.2	Construction of the graph . . . . .	50
6.1.3	Specification of the parameters of the model . . . . .	50
6.2	Evaluation of the MID . . . . .	53
6.2.1	Configuring the evaluation options . . . . .	54
6.2.2	Expanding the model . . . . .	55
6.2.3	Temporal evolution of variables . . . . .	55
6.2.4	Cost-effectiveness analysis . . . . .	55
6.3	Probabilistic sensitivity analysis . . . . .	56
6.3.1	Introducing uncertainty in the model . . . . .	57
6.3.2	Running the analysis . . . . .	59
	<b>Bibliography</b>	<b>62</b>

# List of Figures

1.1	A Bayesian network consisting of two nodes. . . . .	3
1.2	Conditional probability for the variable <i>Disease</i> . . . . .	5
1.3	Conditional probability for the variable <i>Test</i> . . . . .	5
1.4	Prior probabilities for the network <code>BN-disease-test.pgm</code> . . . . .	6
1.5	Posterior probabilities for the evidence case $\{Test = positive\}$ . . . . .	7
1.6	Posterior probabilities for the evidence case $\{Test = negative\}$ . . . . .	7
1.7	A network containing four nodes. The probability of <i>E</i> will be specified using a noisy OR model. . . . .	8
1.8	Canonical parameters of a noisy OR model. . . . .	8
2.1	Network <i>Asia</i> proposed in [26]. . . . .	10
2.2	OpenMarkov's <b>Learning</b> dialog. . . . .	10
2.3	Network <i>Asia</i> learned automatically. . . . .	11
2.4	Selection and discretization of variables. . . . .	12
2.5	Network learned with the <i>Wisconsin Breast Cancer</i> dataset. . . . .	13
2.6	Intervals obtained for the discretized variable <i>RadiusMean</i> . . . . .	13
2.7	Network learned with the <i>Adult</i> dataset. . . . .	15
2.8	Initial state of the interactive hill climbing algorithm for the dataset <i>Asia</i> . . . . .	16
2.9	A version of the network <i>Asia</i> used to impose some links on the learning process. . . . .	16
2.10	Initial state of the interactive PC algorithm for the database <i>Asia</i> . . . . .	18
3.1	A DAN for deciding about the therapy (Example 3.1). . . . .	22
3.2	Utility table for the node <i>Effectiveness</i> . . . . .	23
3.3	Decision tree for Example 3.1. It results from the DAN in Figure 3.1. . . . .	23
3.4	Strategy tree for Example 3.1. It also results from the DAN in Figure 3.1. Compare it with the decision tree in Figure . . . . .	25
3.5	A DAN with two decisions (Example 3.2). . . . .	25
3.6	Restrictions for the link <i>Do test?</i> $\rightarrow$ <i>Result of test</i> . They mean that when the test is not done it can give neither a positive nor a negative result. . . . .	26
3.7	Conditional probability table for <i>Result of test</i> . Some cells are colored in red because of the restrictions shown in Figure 3.6 link restriction. . . . .	26
3.8	Decision tree for Example 3.2. It results from the DAN in Figure 3.5. . . . .	27
4.1	ID for Example 3.1. Compare it with the DAN in Figure 3.1. . . . .	30
4.2	Optimal policy for the decision <i>Therapy</i> . . . . .	31
4.3	Expected utility for the decision <i>Therapy</i> . . . . .	31
4.4	Evaluation of the influence diagram <code>ID-test-2therapies.pgm</code> . . . . .	32
4.5	An influence diagram with two decisions ( <code>ID-decide-test-2therapies.pgm</code> , Example 3.2). . . . .	33
4.6	States for <i>Result of test</i> in the ID for Example 3.2. We have added the dummy state <i>not done</i> . . . . .	33
4.7	Conditional probability table for <i>Result of test</i> . . . . .	34
4.8	Evaluation of the influence diagram <code>ID-decide-test-2therapies.pgm</code> . . . . .	34
4.9	Optimal policy for the node <i>Do test?</i> . . . . .	34
4.10	Expected utility for the node <i>Do test?</i> . . . . .	35

4.11	Policy for the decision <i>Therapy</i> in the influence diagram <code>ID-decide-test-2therapies.pgm</code> .	35
4.12	Expected utility for the decision <i>Therapy</i> in the influence diagram <code>ID-decide-test-2therapies.pgm</code> .	35
4.13	Introduction of the post-resolution finding <i>Disease = present</i> .	37
4.14	Introduction of the post-resolution finding <i>Test = positive</i> .	37
4.15	Introduction of the pre-resolution finding <i>Disease = present</i> .	38
4.16	Optimal strategy for <code>ID-decide-test-2therapies.pgm</code> .	39
4.17	Imposed policy for the node <i>Dec:Test</i> .	39
4.18	Introduction of evidence with an imposed policy.	40
5.1	Cost-effectiveness decision criteria.	42
5.2	Cost of test table.	42
5.3	An decision analysis network, <code>DAN-decide-test-2therapies-CE.pgm</code> (left), and an influence diagram, <code>ID-decide-test-2therapies-CE.pgm</code> (right) for the problem in Example 5.1.	42
5.4	Unicriterion transformation.	43
5.5	Equivalent decision tree for <code>DAN-decide-test-2therapies-CE.pgm</code> and <code>ID-decide-test-2therapies-CE.pgm</code> .	44
5.6	Inference options for cost-effectiveness analysis.	45
5.7	Types of cost-effectiveness analysis.	45
5.8	Global cost-effectiveness analysis.	45
5.9	Optimal strategy when lambda is between 11,171 and 33,383 euros.	46
5.10	Cost-effectiveness plane when analyzing <i>Do Test?</i> .	47
6.1	MID for Chancellor's HIV model.	51
6.2	Conditional probability for <i>State [1]</i> , represented as a tree.	52
6.3	Conditional probability for <i>State [1]</i> , represented as an ADD.	53
6.4	Value of <i>Cost [0]</i> as a function of its parents.	53
6.5	The effectiveness, represented by node <i>Life years [0]</i> , depends on the patient's state, <i>State [0]</i> .	54
6.6	Cost-effectiveness analysis options.	54
6.7	MID model expanded to three time slices	55
6.8	Evolution of variable <i>State</i> over time for combined therapy.	56
6.9	Cost-effectiveness analysis for Chancellor's model.	56
6.10	Cost-effectiveness plane for Chancellor's model.	57
6.11	Frontier interventions for Chancellor's model.	57
6.12	Transition probabilities as Dirichlet distributions for $P(\text{State [1]} \mid \text{State [0]} = \text{state B})$ .	58
6.13	Chancellor's model modified for probabilistic sensitivity analysis.	58
6.14	Utility function for <i>Drug cost [0]</i> .	59
6.15	Probabilistic sensitivity analysis: cost-effectiveness plane for Chancellor's model.	60
6.16	Probabilistic sensitivity analysis: acceptability curve for Chancellor's model	60
6.17	EVPI in Chancellor's model	61

# List of Tables

3.1	Effectiveness of the therapies. . . . .	21
5.1	Cost of each intervention. . . . .	41
5.2	Optimal policy for therapy on the scenario <i>Do test?</i> = <i>yes</i> and <i>Result of test</i> = <i>positive</i> . . . . .	47
6.1	Transition probabilities for monotherapy. . . . .	49
6.2	Transition probabilities for the first three years of combined therapy. . . . .	49
6.3	Annual costs per states and therapy type. . . . .	50
6.4	withDirichlet distribution parameters for transition probabilities . . . . .	57
6.5	Direct medical and community care cost means associated with each state . . . . .	58





# 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 [29], are purely probabilistic, while others, such as influence diagrams [20], include decisions and utilities.

OpenMarkov (see [www.openmarkov.org](http://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 [28]; other excellent books are [23, 8, 22, 24, 27]. 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.pgm`, `.csv`, [www.openmarkov.org](http://www.openmarkov.org)...).

Finally, we should mention that the sections marked with an asterisk (for example, Sec. 4.3) 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):

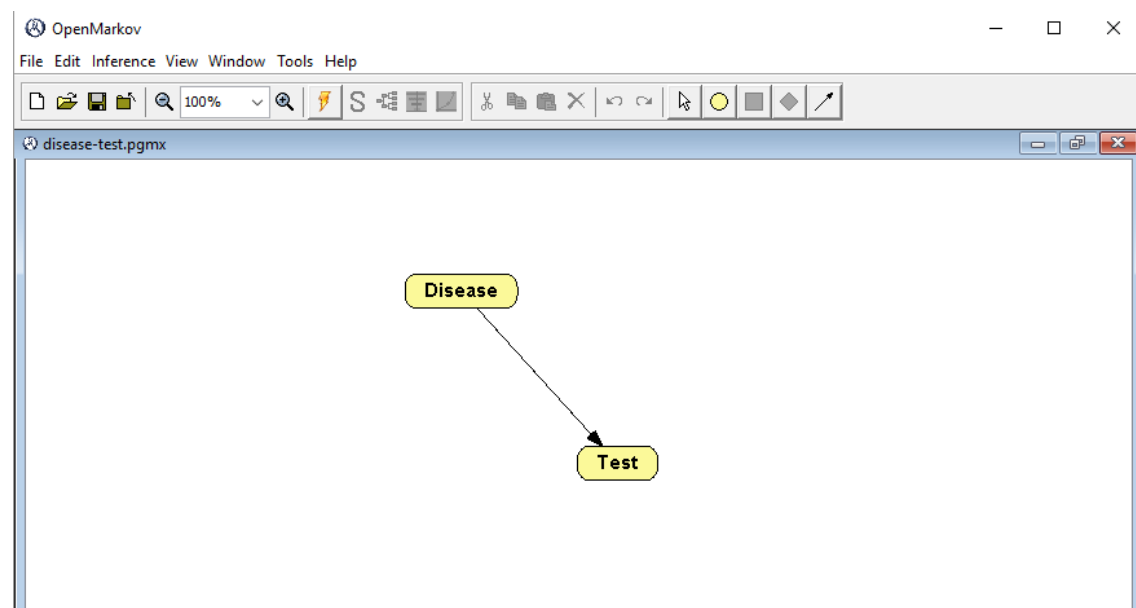







Figure 1.1: A Bayesian network consisting of two nodes.

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 (S), 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.pgm`, 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 90% and a specificity of 93%. 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  $Disease \rightarrow Test$ .

### 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 where 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. 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 `.pgm` 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](http://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  $P(\text{Disease}=\text{present}) = 0.14$ . 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).

present	0.14
absent	0.86

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):

Disease	absent	present
positive	0.07	0.9
negative	0.93	0.1

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.

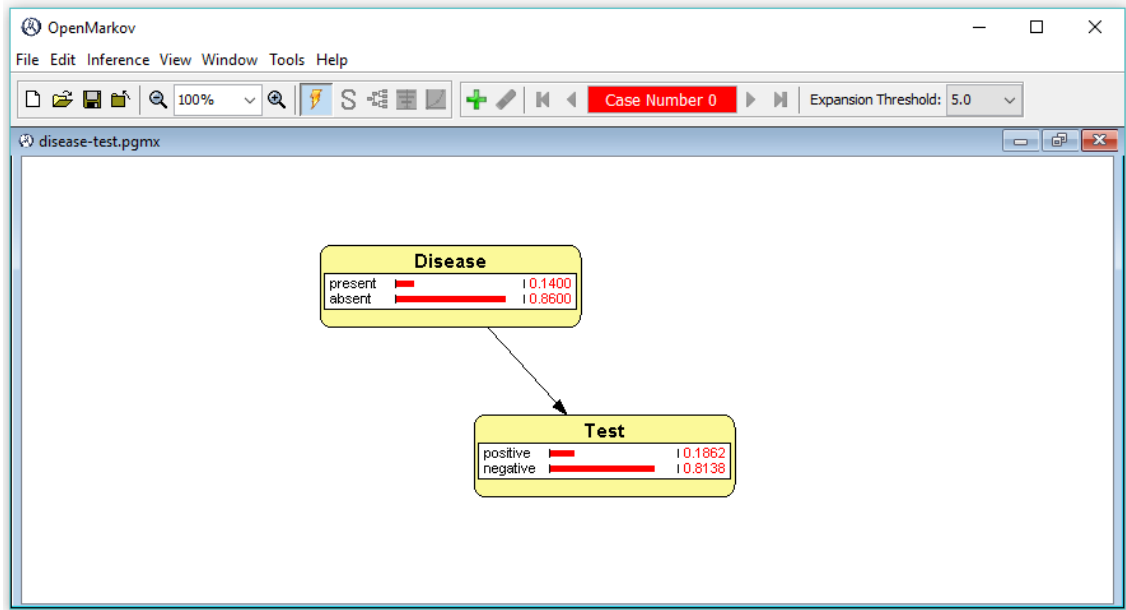


Figure 1.4: Prior probabilities for the network BN-disease-test.pgm.

### 1.3.1 Entering findings

A **finding** consists of 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  $P(Disease=absent | Test=negative) = 0.9828$ .

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.

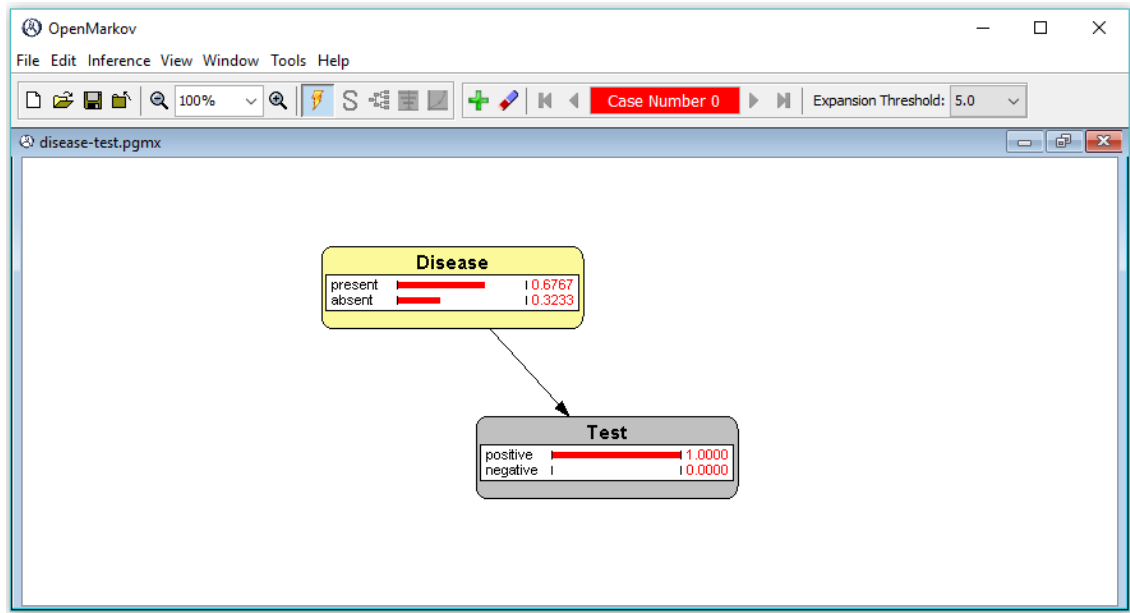


Figure 1.5: Posterior probabilities for the evidence case  $\{Test = positive\}$ .

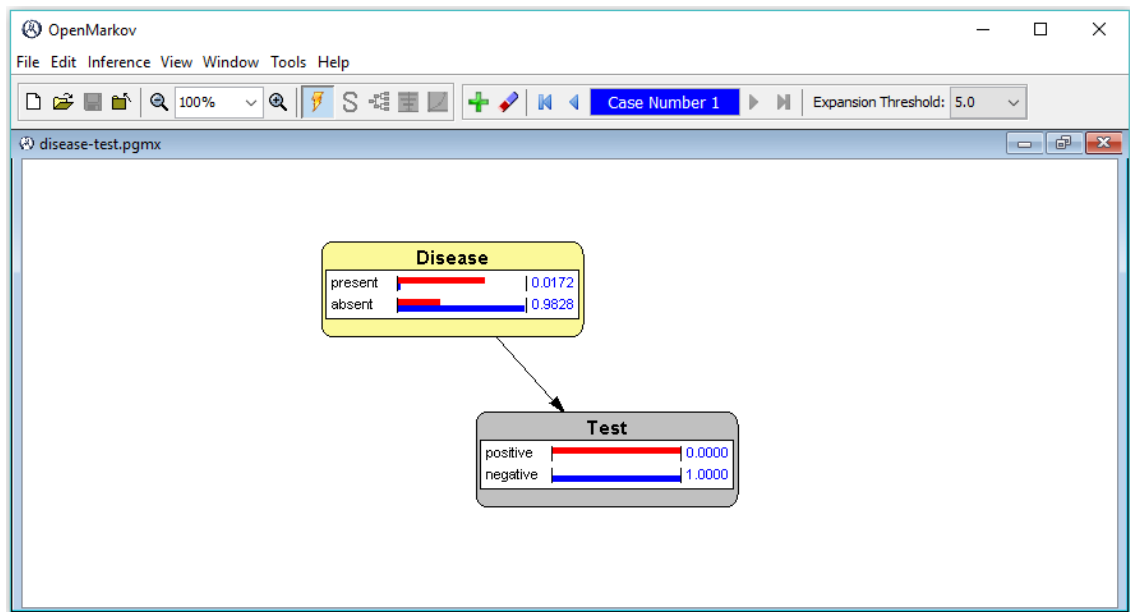


Figure 1.6: Posterior probabilities for the evidence case  $\{Test = negative\}$ .

## 1.4 Canonical models

Some probabilistic models are called **canonical** because they can be used as elementary blocks for building more complex models [29]. 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 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

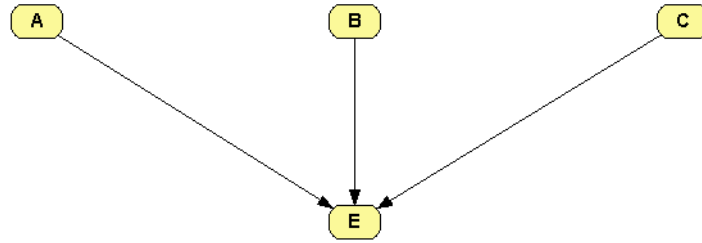


Figure 1.7: A network containing four nodes. The probability of  $E$  will be specified using a noisy OR model.

Node Potential: E

Relation Type: **OR / MAX** Reorder variables

☐ Whole table  
☒ Canonical parameters

☐ Net  
☐ Compound

☐ Show as probabilities  
☐ Show as values

☐ All parameters  
☐ Independent parameters

	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

<<Double click to add/modify comment>>

OK Cancel

Figure 1.8: Canonical parameters of a noisy OR model.

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 from data

### 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 [26] with the hill climbing algorithm, also known as search-and-score [19]. As a dataset we will use the file `asia10K.csv`, which contains 10,000 cases randomly generated from the Bayesian network `BN-asia.pgm` (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.

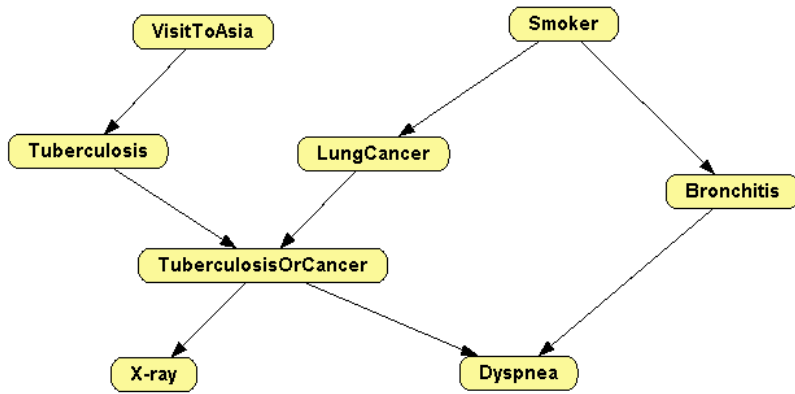


Figure 2.1: Network *Asia* proposed in [26].

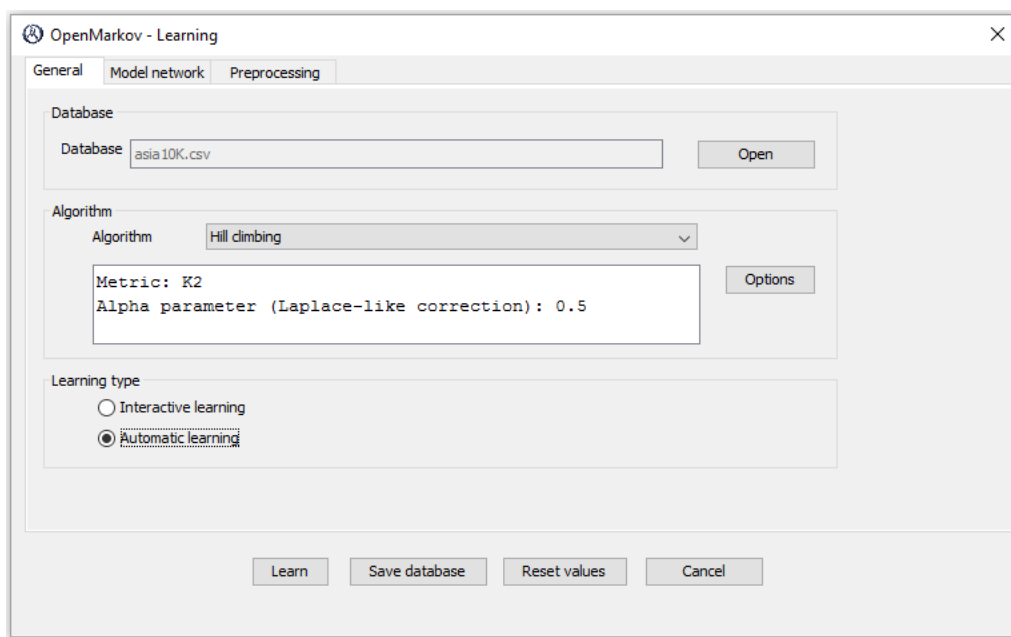


Figure 2.2: OpenMarkov’s Learning dialog.

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  $\alpha$  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.

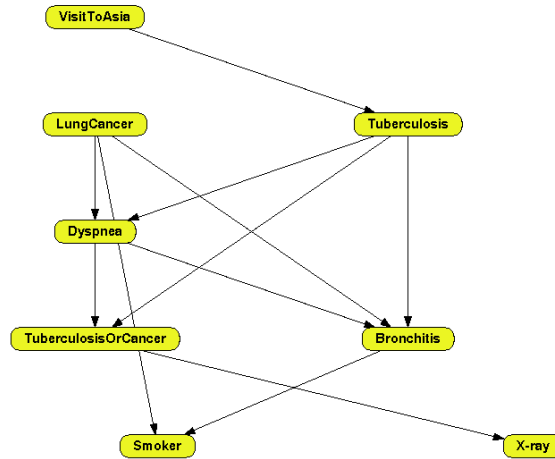


Figure 2.3: Network *Asia* learned automatically.

## 2.2.2 Positioning the nodes with a model network

For those who are familiar with the *Asia* network as presented in the literature [26], 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.pgm](http://www.cisiad.uned.es/ProbModelXML/examples/bn/BN-asia.pgm).
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.pgm`.
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.

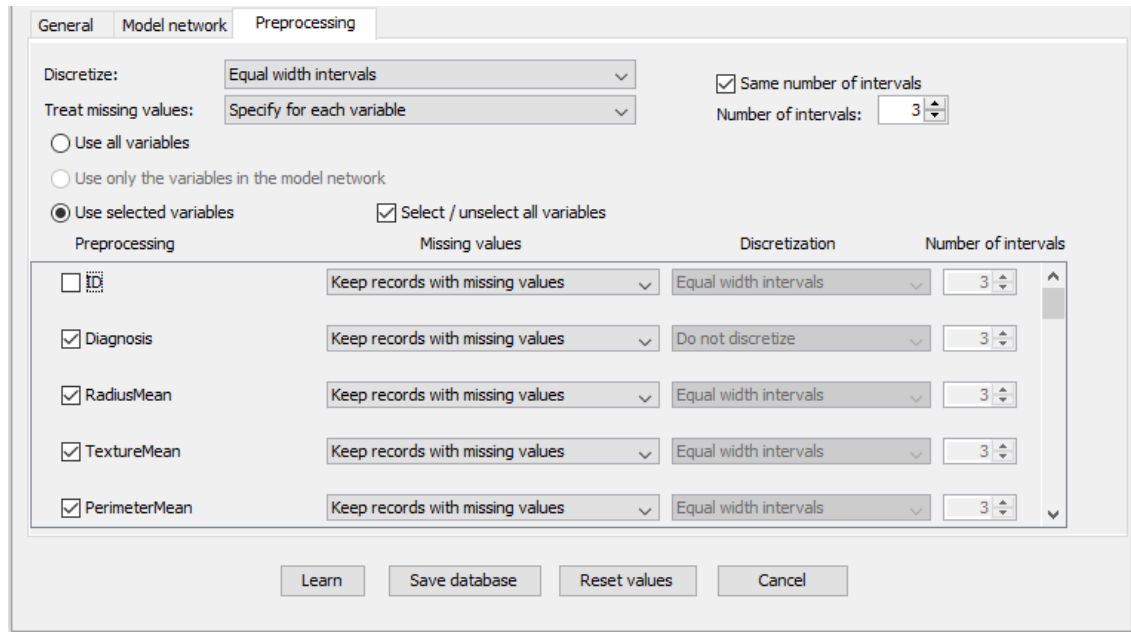


Figure 2.4: Selection and discretization of variables.

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*,<sup>1</sup> check the box **Same number of intervals**<sup>2</sup> 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,

<sup>1</sup>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.

<sup>2</sup>Leaving this option unchecked will allow you to choose a different number of intervals for each variable.

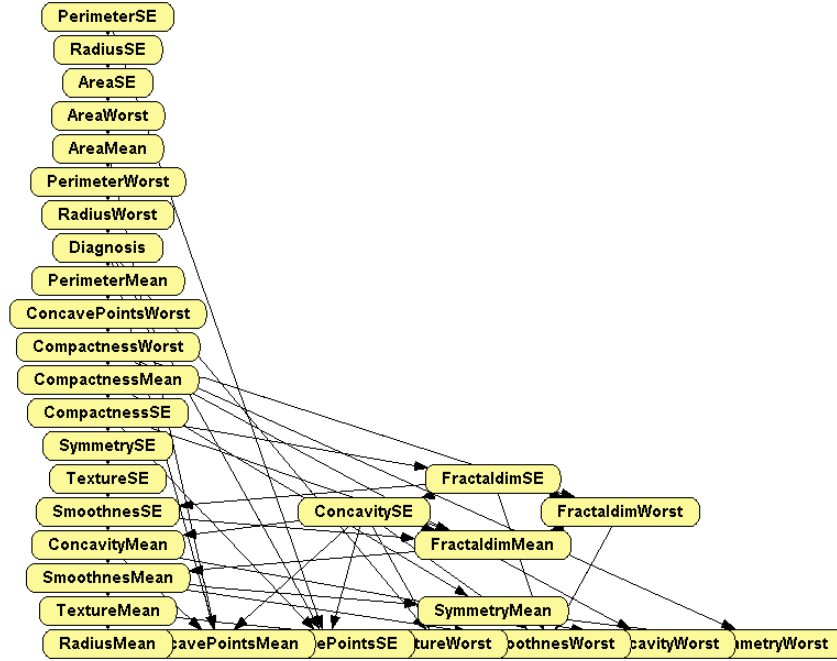


Figure 2.5: Network learned with the *Wisconsin Breast Cancer* dataset.

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.

States:	Interval	Min	Max
[6.981, 14.024]	6.981	14.024	
(14.024, 21.067]	14.024	21.067	
(21.067, 28.11]	21.067	28.110	

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

- 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**.

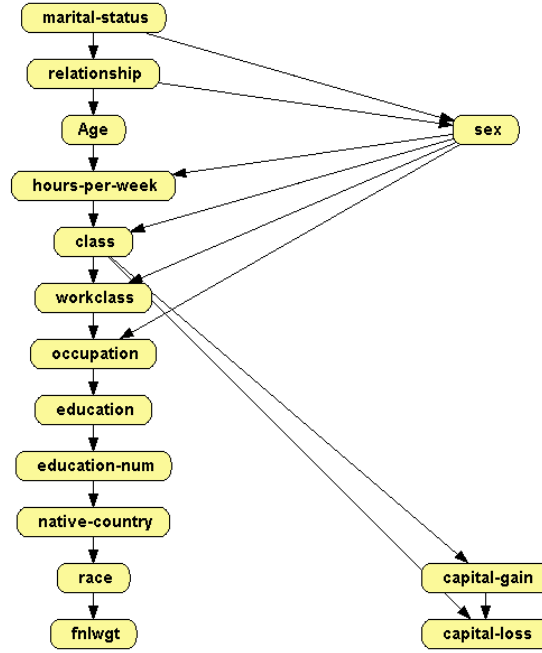


Figure 2.7: Network learned with the *Adult* dataset.

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 learning 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.

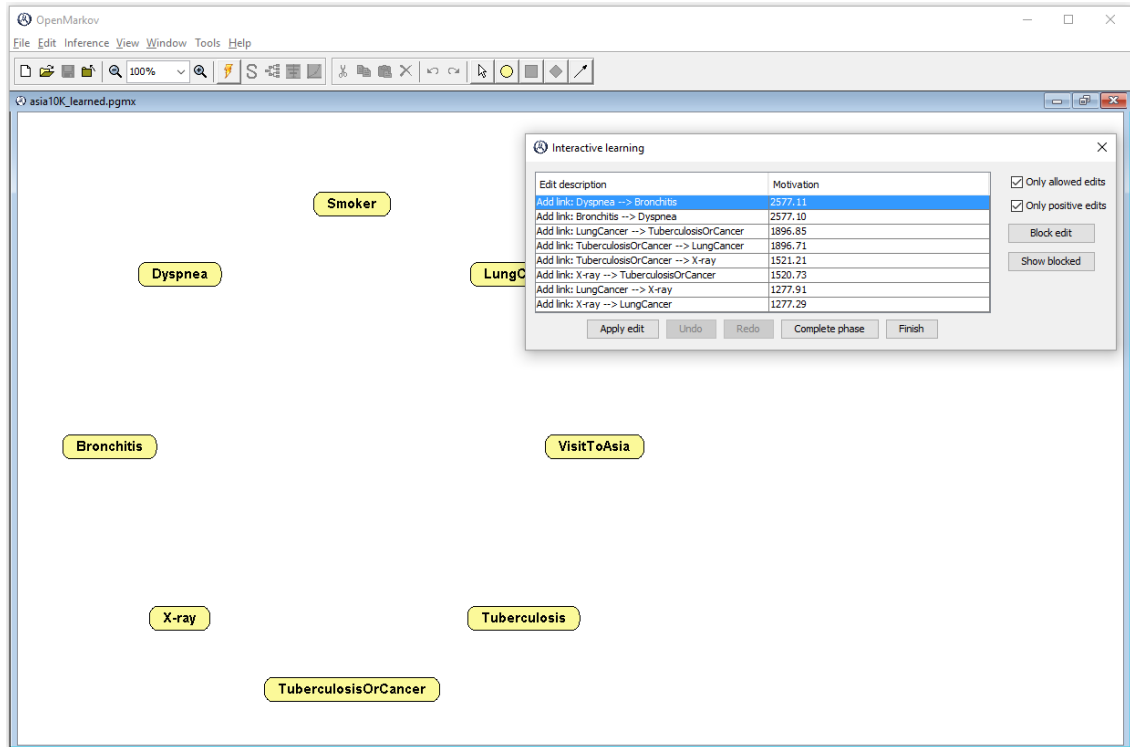


Figure 2.8: Initial state of the interactive hill climbing algorithm for the dataset *Asia*.

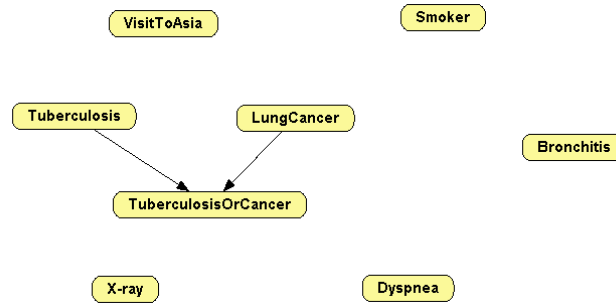


Figure 2.9: A version of the network *Asia* used to impose some links on the learning process.

### 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 [32] 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  $H_0$ .<sup>3</sup> When  $p$  is smaller than a certain value, called **significance level** and denoted by  $\alpha$ , we reject  $H_0$ , i.e., we conclude that the correlation is authentic and keep the link  $X$ – $Y$ .<sup>4</sup> In contrast, when  $p$  is above the threshold  $\alpha$  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 of 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.<sup>5</sup> In the first phase of the algorithm, each edit shown in the **Interactive learning** window consists of 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

---

<sup>3</sup>More precisely,  $p$  is the probability of finding the observed correlation (or a bigger correlation) conditioned on  $H_0$  being true.

<sup>4</sup>Please note that in Section 2.2.1, item 5, we used the symbol  $\alpha$  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  $\alpha$  in the literature, used by the PC algorithm.

<sup>5</sup>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.

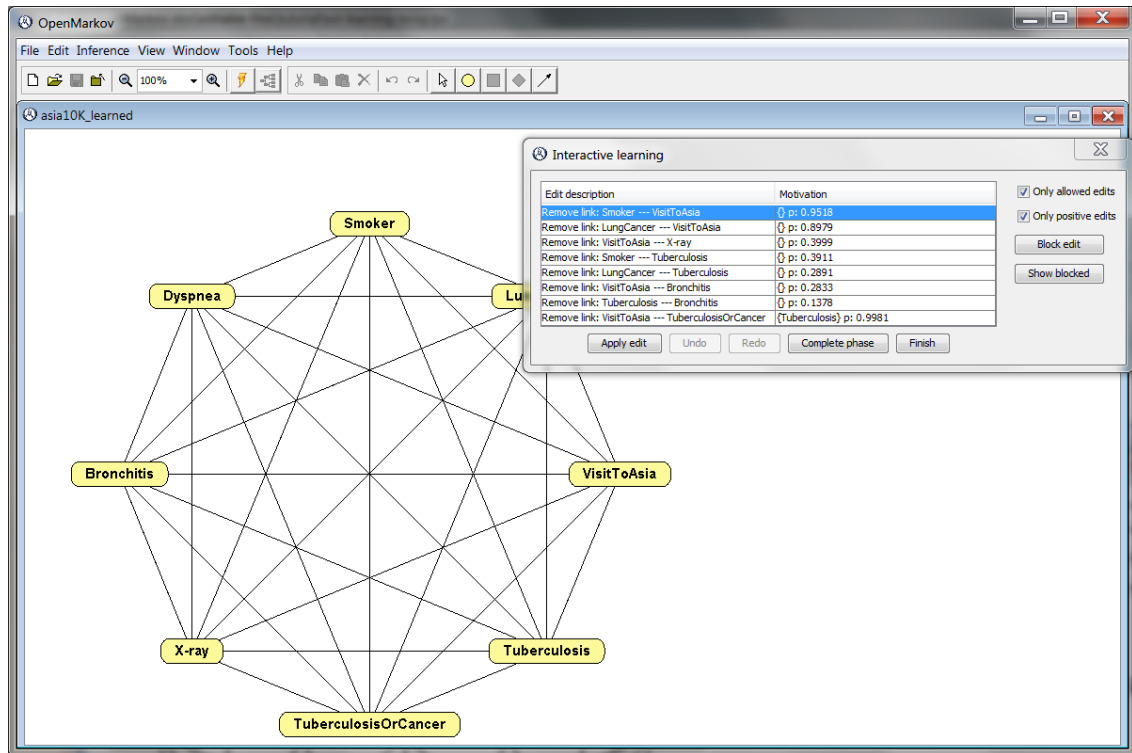


Figure 2.10: Initial state of the interactive PC algorithm for the database *Asia*.

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* “ $\{Tuberculosis\}$   $p$ : 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 of orienting some pairs of links head-to-head based on the tests of independence carried out in the first phase [29, 32]. 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](http://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) [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.

### 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.<sup>1</sup> 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?

	<i>Disease = present</i>	<i>Disease = absent</i>
<i>no therapy</i>	1.2	10.0
<i>therapy 1</i>	4.0	9.9
<i>therapy 2</i>	6.5	9.3

Table 3.1: Effectiveness of the therapies.

We observe in Table 3.1 that the best scenario occurs when 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.

#### 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

---

<sup>1</sup>At this moment we do not care about the scale used to measure effectiveness. We will discuss it in Chapter 5, when speaking about multicriteria decision making.

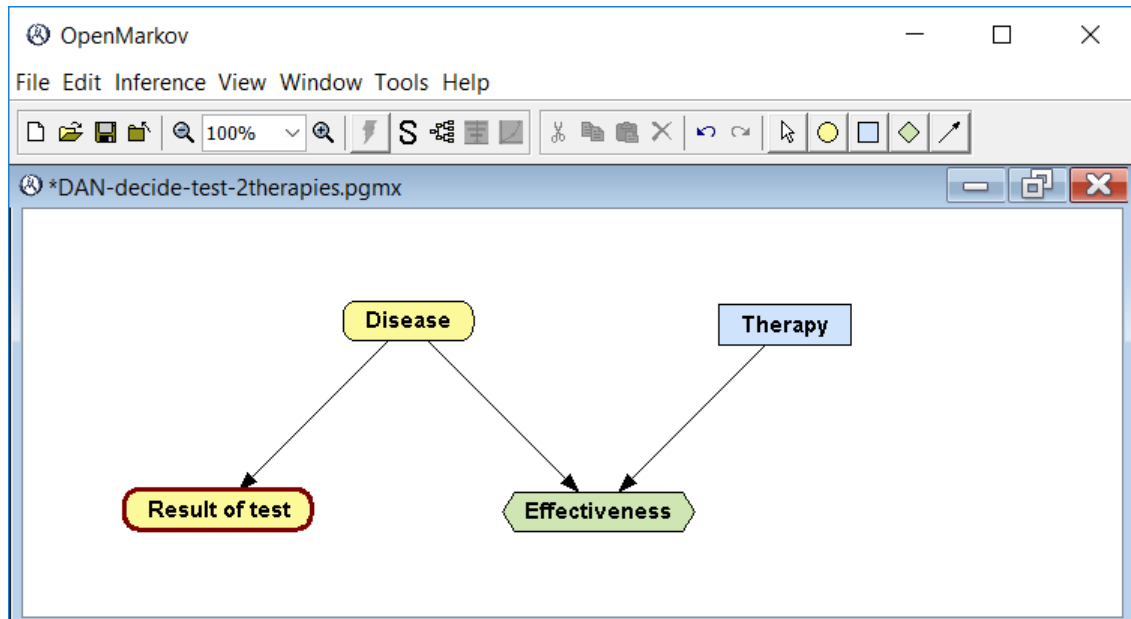




Figure 3.1: A DAN for deciding about the therapy (Example 3.1).

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.pgm`. 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.<sup>2</sup>
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.

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.pgm`.

<sup>2</sup>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.

Node Potential: Effectiveness						
Relation Type: <span>Exact</span> <span>Reorder variables</span>						
Disease	absent	absent	absent	present	present	present
Therapy	none	therapy 1	therapy 2	none	therapy 1	therapy 2
Effectiveness	10	9.9	9.3	1.2	4	6.5

<<Double click to add/modify comment>>

OK Cancel

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

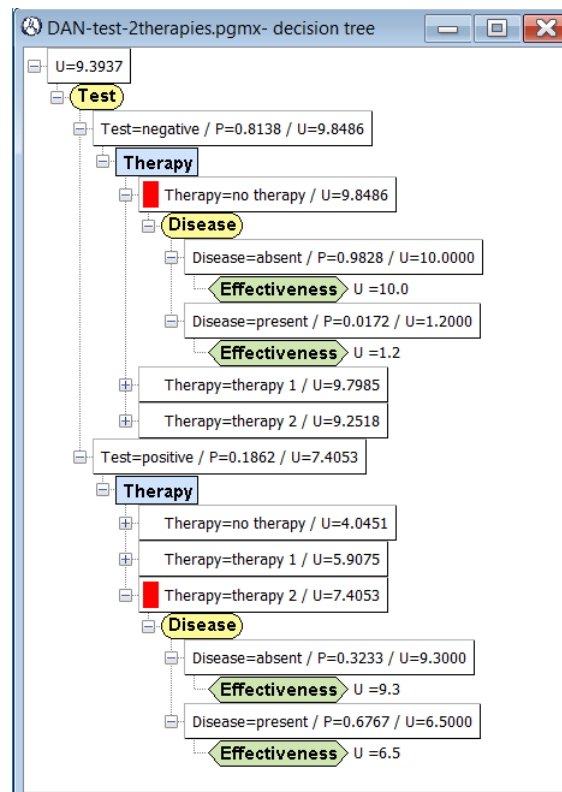



Figure 3.3: Decision tree for Example 3.1. It results from the DAN in Figure 3.1.


### 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 [31, 30].
  - (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 () 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 *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.



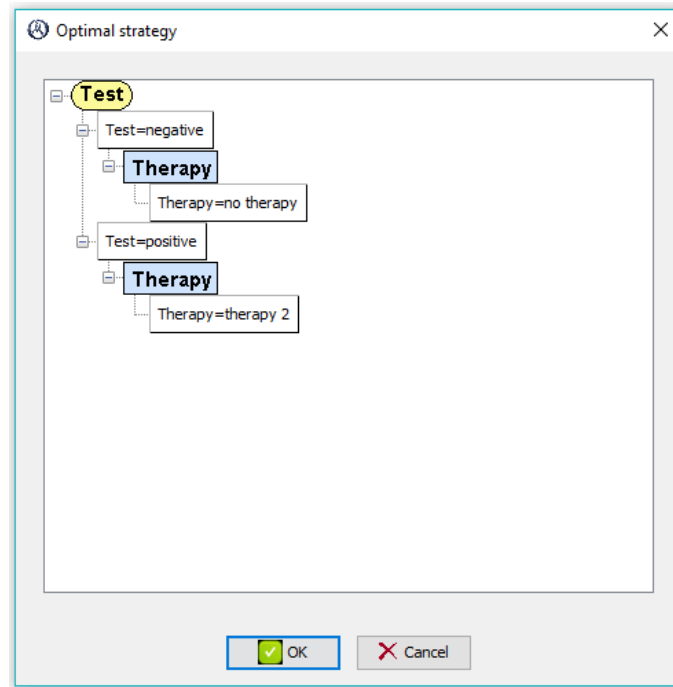


Figure 3.4: Strategy tree for Example 3.1. It also results from the DAN in Figure 3.1. Compare it with the decision tree in Figure

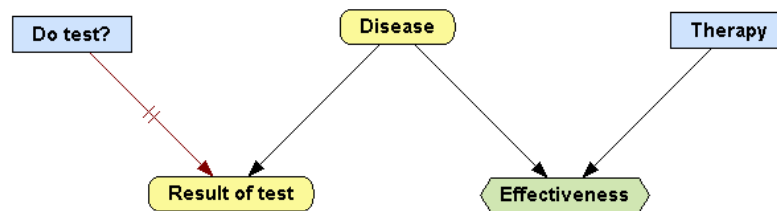


Figure 3.5: A DAN with two decisions (Example 3.2).


## 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.

1. Open the network `DAN-test-2therapies.pgm`.
2. Rename the node *Test* as *Result of test*, and uncheck the box *Always observed*.
3. Add the decision node *Do Test?*. Keep the default states, *yes* and *no*.
4. Draw the link  $Do\ Test? \rightarrow Result\ of\ test$ .
5. If  $Do\ test? = no$ , the result of the test can be neither *positive* nor *negative*. We encode this information by right-clicking on the link  $Do\ Test? \rightarrow Result\ of\ test$  and selecting **Add**

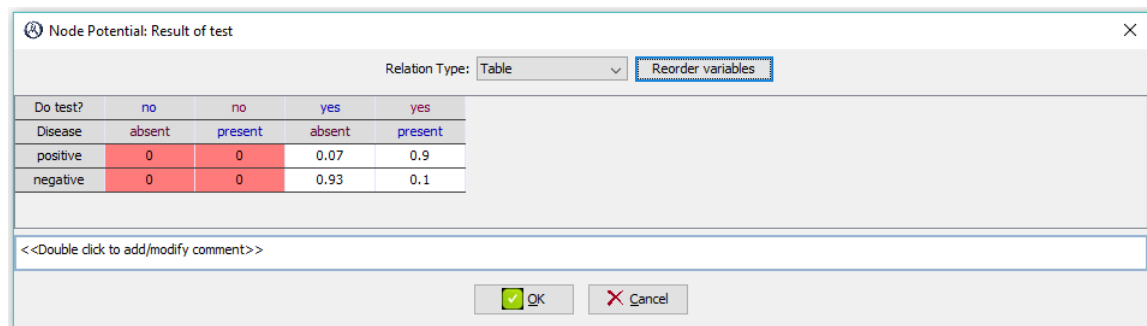


Link Restriction: Link between Do test? and Result of test

Do test?	no	yes
positive	0	1
negative	0	1

OK Cancel

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



Node Potential: Result of test

Relation Type: Table Reorder variables

Do test?	no	no	yes	yes
Disease	absent	present	absent	present
positive	0	0	0.07	0.9
negative	0	0	0.93	0.1

<<Double click to add/modify comment>>

OK Cancel

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

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  $Result\ of\ test = yes$  and  $Result\ of\ test = yes$ . Press OK.

- 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,  $Result\ of\ test$ . (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.)
- Open the probability table for  $Result\ of\ test$  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.
- If the test is performed ( $Do\ Test? = yes$ ), the result of test is known. In order to establish this **revelation condition**, right-click on link  $Do\ Test? \rightarrow Result\ of\ test$ , select **Edit revelation conditions** and check the box for **yes**. The link will be colored in dark red.
- Save this DAN as `DAN-decide-test-2therapies.pgm`.
- Observe that the optimal strategy is to do the test and apply the second therapy if and only if it gives a positive result. In the decision tree you can check that the expected utility of

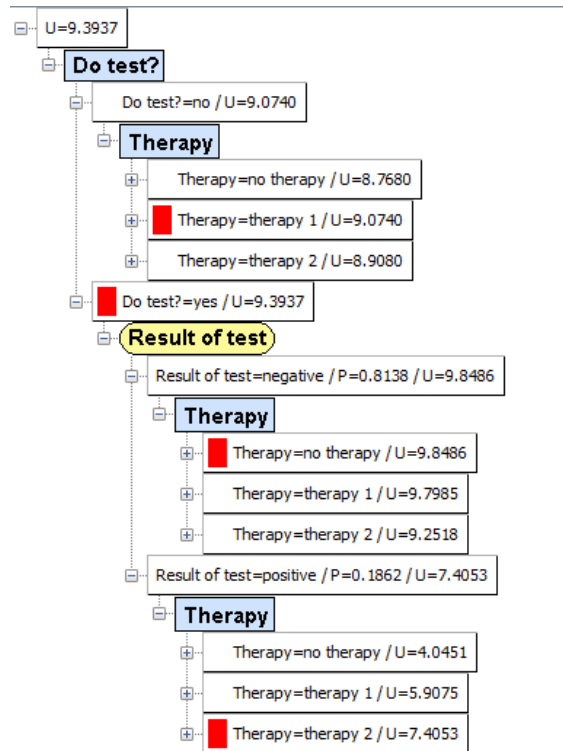


Figure 3.8: Decision tree for Example 3.2. It results from the DAN in Figure 3.5.

doing the test, 9.3937, is higher than that of not doing it, 9.0740—see Figure 3.8—and the optimal branches (those marked with red rectangles) are the same as in the strategy tree.



## 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 many problems that can be represented and solved with both types of models but the DAN is easier to build than the 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 two 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.<sup>1</sup> 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.pgm`. 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.
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,

---

<sup>1</sup>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 [20]. Thus, for more than a quarter of a century a paper widely cited in decision analysis, artificial intelligence, economics, control theory, and several other fields was not available in any periodicals library, until it was included in a special double issue of the *Decision Analysis* journal in 2005 [21], 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 [13] and three relevant AI journals have already rejected it. We hope this paper will not need other 25 years to appear in a journal.

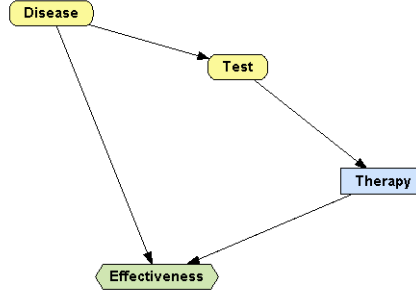


Figure 4.1: ID for Example 3.1. Compare it with the DAN in Figure 3.1.

which is not available for IDs, we draw a link from *Result of test* 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.pgm`.
6. Check that the decision tree and the strategy tree are the same as for the DAN (Figs. 3.3 and 3.3).

#### 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.<sup>2</sup> 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 Fig 3.5, when *Do test?* = *yes*, both this decision and the variable *Result of test* are informational predecessors of *Therapy*, but when *Do test?* = *no*, the only informational predecessor of *Therapy* is *Do test?* 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 a variable *X* is observed when making decision *D* 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 be 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.

A **strategy** is a set of policies, one for each decision in the influence diagram. Each strategy has a **expected utility**, which depends on the probabilities and utilities that define the influence diagram 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.

<sup>2</sup>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 [14]. Therefore, the inference options that we will introduce in this section are also available for symmetric DANs, but in general they cannot be applied to asymmetric DANs.

Test	negative	positive
therapy 2	0	1
therapy 1	0	0
no therapy	1	0

<<Double click to add/modify comment>>

OK Cancel

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

Test	negative	positive
therapy 2	9.251831	7.405263
therapy 1	9.798501	5.907519
no therapy	9.848611	4.045113

<<Double click to add/modify comment>>

OK Cancel

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

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

In order to evaluate this influence diagram, click the **Inference** button (🔍) to switch from **edit mode** to **inference mode**. OpenMarkov performs two operations: the first one, called **resolution**, consists of finding the optimal strategy, as we have just explained. 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 of 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 influence diagram contains only one utility node, the value *9.3937* shown therein is the global maximum expected utility of the influence diagram, which coincides with the one at the root node of the decision tree in Figure 3.3.

## 4.2 An example with two decisions

We will now solve with an ID the problem stated in Example 3.2. The result is shown in Figure 4.5.

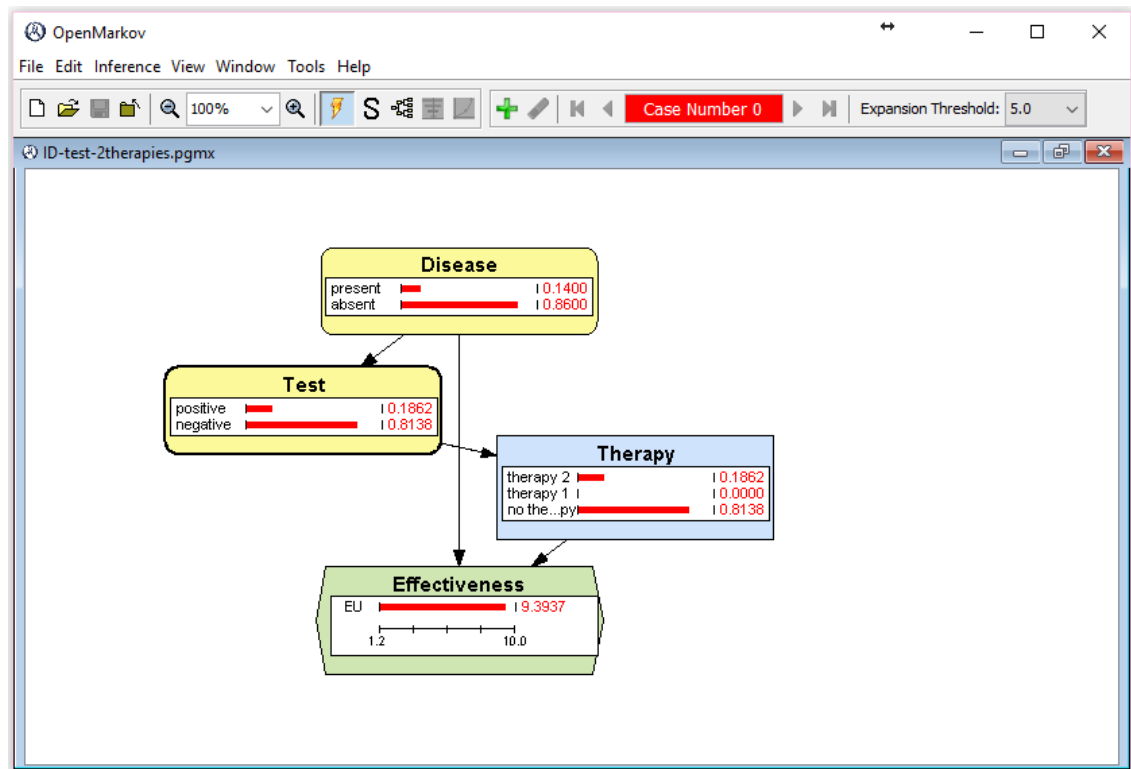


Figure 4.4: Evaluation of the influence diagram ID-test-2therapies.pgm.

#### 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.pgm.
2. Rename the node *Test* as *Result of test*.
3. Create the decision node *Do Test?*. Keep the default states, *yes* and *no*.
4. Draw the link *Do Test?*→*Result of test*.
5. If *Do test?* = *no*, the result of the test can be neither *positive* nor *negative* but, unlike the case of DANs, the variable *Result of test* must take one value also in this case. 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 test* has changed, its conditional probability table has been lost. Open the conditional probability table *Result of test* 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 influence diagram as ID-decide-test-2therapies.pgm.
8. Observe that the strategy is the same as in Figure 3.4.
9. In contrast, the decision tree is bigger than that in Figure 3.8 because the node *Result of test* appears also in the branches for *Do test?* = *no*, and it always has three branches, some of which with zero probability.



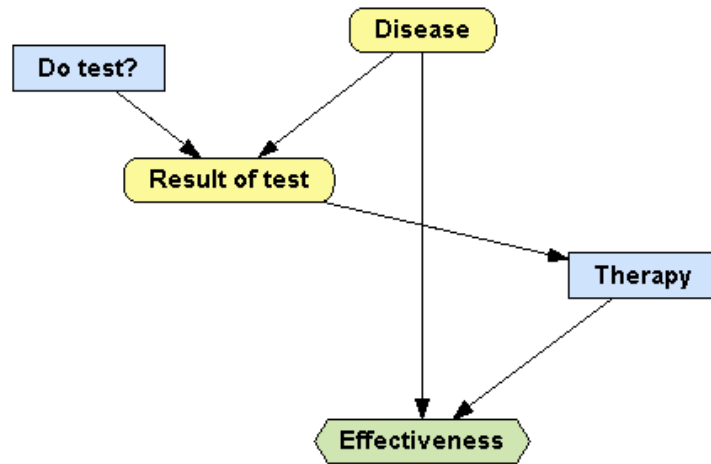


Figure 4.5: An influence diagram with two decisions (ID-decide-test-2therapies.pgm, Example 3.2).

Figure 4.6: States for *Result of test* in the ID for Example 3.2. We have added the dummy state *not done*.

## 4.2.2 Resolution of the influence diagram

Let us now evaluate the influence diagram and examine all the information that OpenMarkov offers us.

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?* is to always do the test (Figure 4.9) because the expected utility for this

Node Potential: Result of test

Relation Type: Table Reorder variables

Do test?	no	no	yes	yes
Disease	absent	present	absent	present
positive	0	0	0.07	0.9
negative	0	0	0.93	0.1
not performed	1	1	0	0

<<Double click to add/modify comment>>

OK Cancel

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

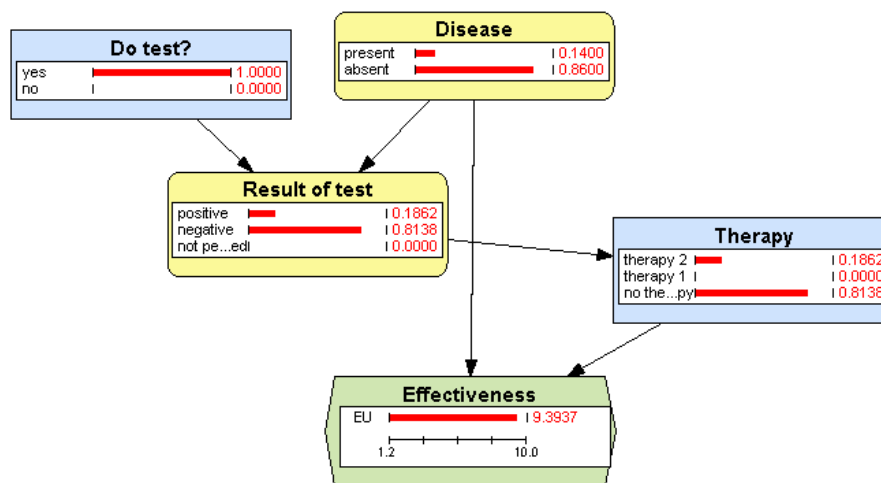


Figure 4.8: Evaluation of the influence diagram ID-decide-test-2therapies.pgm.

option is higher (Figure 4.10).

- 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 the therapies 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

Optimal policy: Do test?

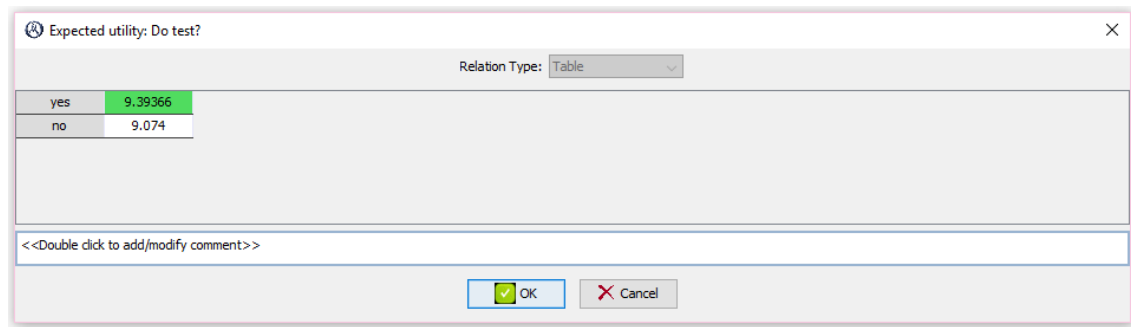
Relation Type: Table

yes	1
no	0

<<Double click to add/modify comment>>

OK Cancel

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



Expected utility: Do test?

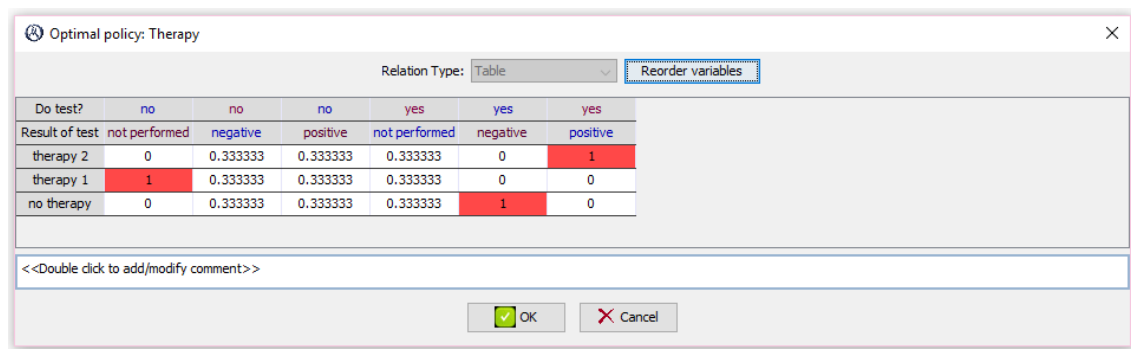
Relation Type: Table

yes	9.39366
no	9.074

<<Double click to add/modify comment>>

OK Cancel

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



Optimal policy: Therapy

Relation Type: Table Reorder variables

Do test?	no	no	no	yes	yes	yes
Result of test	not performed	negative	positive	not performed	negative	positive
therapy 2	0	0.333333	0.333333	0.333333	0	1
therapy 1	1	0.333333	0.333333	0.333333	0	0
no therapy	0	0.333333	0.333333	0.333333	1	0

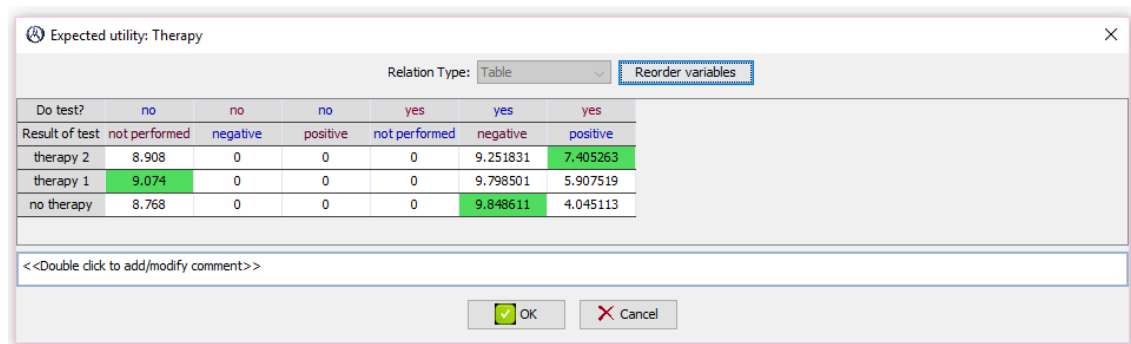
<<Double click to add/modify comment>>

OK Cancel

Figure 4.11: Policy for the decision *Therapy* in the influence diagram ID-decide-test-2therapies.pgm.

when the test is done and gives a positive result. The states in an impossible configuration have the same probability of being the best alternative but, the expected utility of these configurations are zero as shown in Figure 4.12.

There are three columns that indicate “apply the first therapy in 33% of cases, apply the second therapy in 33% and do not apply any therapy in the other 33%”. The reason for this is that, as there was a tie in the expected utilities, OpenMarkov has made the Solomonian decision of not favoring any of the three options: a third of the patients should receive the first therapy, other third should receive the second therapy and the other third should not receive any therapy. However, the content of these columns is irrelevant because they



Expected utility: Therapy

Relation Type: Table Reorder variables

Do test?	no	no	no	yes	yes	yes
Result of test	not performed	negative	positive	not performed	negative	positive
therapy 2	8.908	0	0	0	9.251831	7.405263
therapy 1	9.074	0	0	0	9.798501	5.907519
no therapy	8.768	0	0	0	9.848611	4.045113

<<Double click to add/modify comment>>

OK Cancel

Figure 4.12: Expected utility for the decision *Therapy* in the influence diagram ID-decide-test-2therapies.pgm.

correspond to impossible scenarios: two of them,  $\{Do\ test? = no, Test = negative\}$  and  $\{Do\ test? = no, Test = positive\}$ , are impossible, because when the test is not done it gives neither a positive nor a negative result—see Figure 4.7; the scenario  $\{Do\ test? = yes, Test = not\ performed\}$  is also impossible due to an obvious contradiction—see again Figure 4.7.

The first column means that if the test were not done, the best option would be to apply the first therapy. However, this scenario is also impossible because the optimal strategy implies doing the test in all cases.

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?* 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 Introducing evidence in influence diagrams\*

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

#### 4.3.1 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 influence diagram `ID-decide-test-2therapies.pgm` by **first** evaluating it and **then** introducing the finding *Disease = present*, as we did with Bayesian networks (Sec. 1.3.1). The result, shown in Figure 4.13, is that the test will give a positive result in 90% of cases (in accordance with the sensitivity of the test) and a negative result in the rest. We can also see that 90% of those patients will receive the second therapy, as expected from the policy for *Therapy*. We can also see that the expected value of *Effectiveness* is 5.97; it is the result of a weighted average, because the effectiveness is 6.5 for those having a positive test and 1.2 for those having a negative test (see Table 3.1):  $6.5 \times 0.9 + 1.2 \times 0.1 = 5.97$ . Finally, the answer to the above question is that the expected utility for those patients is 5.97.

**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 influence diagram `ID-decide-test-2therapies.pgm` in a similar way, i.e., introducing the post-resolution finding *Result of test = positive*, as in Figure 4.14, 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):  $6.5 \times 0.6767 + 9.3 \times 0.3233 = 7.4053$ .

#### 4.3.2 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?

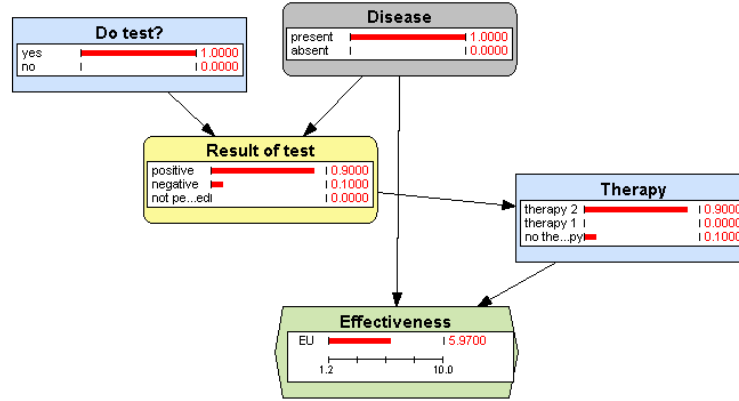


Figure 4.13: Introduction of the post-resolution finding  $Disease = present$ .

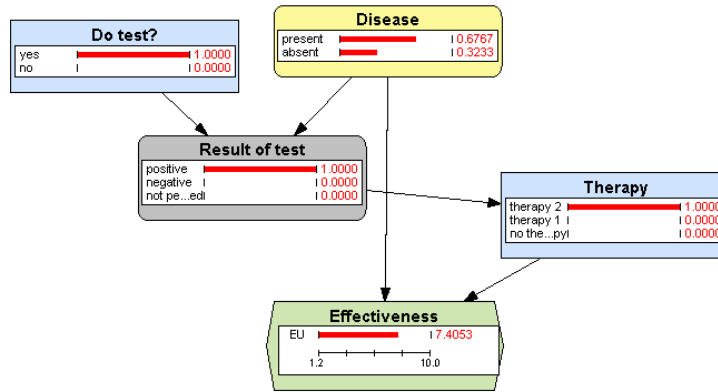


Figure 4.14: Introduction of the post-resolution finding  $Test = positive$ .

One way to solve this problem with OpenMarkov is to use the influence diagram built above, but introducing the finding  $Disease = present$  **before** evaluating this influence diagram—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* in **edit mode** and **then** evaluating the influence diagram—see Figure 4.15. The optimal strategy computed by OpenMarkov is “apply the second therapy”. As we can see performing the test is irrelevant (there is a tie in the expected utilities of both states).

Another difference is that the node *Disease* in Figure 4.15 is colored in dark gray to indicate that it has a pre-resolution finding, while the lighter gray in Figure 4.13 denotes a post-resolution finding.<sup>3</sup>

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,

<sup>3</sup>As a mnemonic, remember that a *darker* gray denotes a *stronger* impact on the influence diagram, 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. You can also remember that a *lighter* gray denotes a finding that arrives *later*, i.e., after the resolution.

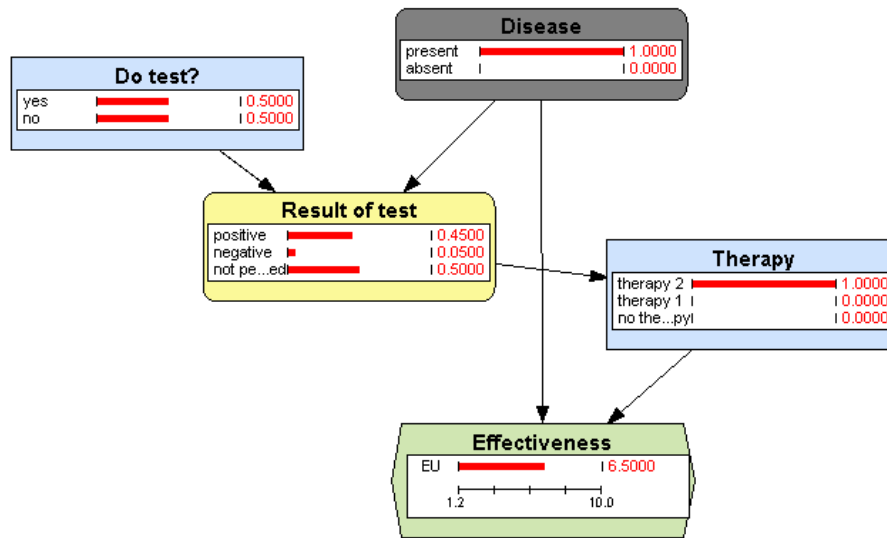


Figure 4.15: Introduction of the pre-resolution finding *Disease = present*.

i.e., how he/she will behave, uses the influence diagram—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 [18], while post-resolution findings correspond to the concept of evidence proposed in [25].

## 4.4 Imposing policies\*

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 influence diagram 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. In this case, the optimal strategy would be “do test, and apply the second therapy only if the result of test is positive”. However, we may wonder what would be the utility if the test is not performed.

In order to solve this problem, open `ID-decide-test-2therapies.pgm`.

1. Be sure that the network don't has any evidence and you are in edition mode.
2. Click on the **Show optimal strategy** icon. Observe that now the optimal policy is the same than in Figure 4.16.
3. Click on the **Inference mode** icon. Enter Double-click on the state *no* of the node *Do Test?* to observe the posterior probabilities of *Disease*.

The result is that OpenMarkov gives an error message, “Incompatible evidence”, because the test cannot give a negative result if the posterior probability of doing the test is one. Then we can

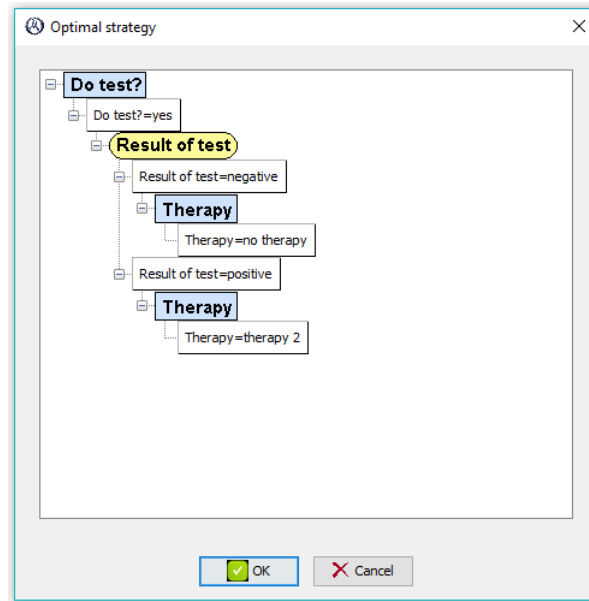


Figure 4.16: Optimal strategy for ID-decide-test-2therapies.pgm.

say: “OK, I know the optimal policy is to do the test, but what would happen if, applying a **non-optimal policy**, I decided to don’t do it?”, which is the question posed in the statement of this example. This kind of **what-if reasoning** can be performed by following these steps:

1. Come back to **edit mode** by clicking on the **Inference** icon, because policies can only be imposed in edit mode.
2. In the contextual menu of *Do Test?*, select **Impose policy**. In the field **Relation type**, select *Table* and introduce the values shown in Figure 4.17. This imposed policy means that 100% of patients will not receive the test.

Node Potential: Do test?	
Relation Type: Table	
yes	0
no	1
<<Double click to add/modify comment>>	
OK Cancel	

Figure 4.17: Imposed policy for the node *Dec:Test*.

3. Click **OK** and observe that the node *Do Test?* is now colored in a darker blue, to indicate that it has an imposed policy.
4. Evaluate the influence diagram. Observe that now the probability of not doing the test is 100%, as we wished.
5. Now the effectiveness is 9.0740 (Figure 4.18), which answers the question posed in the statement of this example. Observe that now, the optimal policy for *Therapy* is to apply the first therapy.

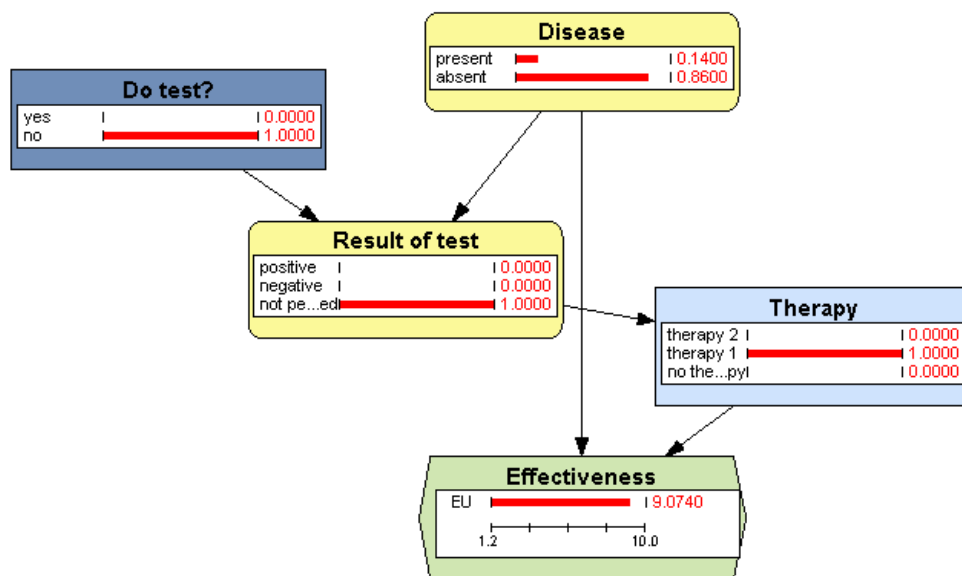


Figure 4.18: Introduction of evidence with an imposed policy.



## Chapter 5

# Multicriteria analysis with DANs and IDs

The decision models we have considered so far have only one criterion: the effectiveness. However, in practice tests and therapies have costs, which must be taken into account, as in the following example:

**Example 5.1.** The test mentioned in the Example 3.2 has a cost of €150 and each possible therapy has a cost shown in Table 5.1. The effectiveness, measured in quality-adjusted life years (QALYs) is the same than in previous examples (Table 3.1).

	<i>Cost</i>
<i>No therapy</i>	€ 0
<i>Therapy 1</i>	€ 20,000
<i>Therapy 2</i>	€70,000

Table 5.1: Cost of each intervention.

The main problem is that now we have two different criteria (cost and effectiveness) and each one is measured in different units (€ and QALYs respectively). So we will explain how to solve this problem using IDs [1] and DANs [11].

### 5.1 Construction of multicriteria networks

In order to resolve this problem we are going to adapt `ID-decide-test-2therapies.pgm` or `DAN-decide-test-2therapies.pgm`. These steps are the same for both networks.

1. As we have two different criteria, one for effectiveness and the other one for the costs, we have to add these criteria to the network. First, double-click on the network panel to open the **Network properties** dialog. Click on **Advanced** tab and then on **Decision criteria**. Use **standard criteria** button to select **Cost-effectiveness (€/QALY)**. The result is shown in Figure 5.1.
2. Open *Effectiveness* **Node properties** dialog and set **Effectiveness** as their decision criterion.
3. Create the utility node *Cost of test*. Open **Node properties** dialog and observe that *Cost* is its decision criterion.
4. Create the utility node *Cost of therapy*. Open **Node properties** dialog and observe that *Cost* is its decision criterion.
5. Draw the links *Do test?* → *Cost of test* and *Therapy* → *Cost of therapy*.

Criterion	Unit
Cost	€
Effectiveness	QALY

Standard criteria

+ Add

- Delete

↑ Up

↓ Down

OK Cancel

Figure 5.1: Cost-effectiveness decision criteria.

Node Potential: Cost of test

Relation Type: Exact

Do test?	no	yes
Cost of test	0	150

<<Double click to add/modify comment>>

OK Cancel

Figure 5.2: Cost of test table.

- Open the potential for the node *Cost of test*. In the *Relation type* field select *Exact*. Introduce the value of Example 3.2, as shown in Figure 5.2.
- Open the potential for the node *Cost of therapy*. In the *Relation type* field select *Exact*. Introduce the values of Table 5.1.
- Save the network as `DAN-decide-test-2therapies-CE.pgm` or `ID-decide-test-2therapies-CE.pgm`.

The resulting network must be similar to influence diagram or the DAN of the Figure 5.3.

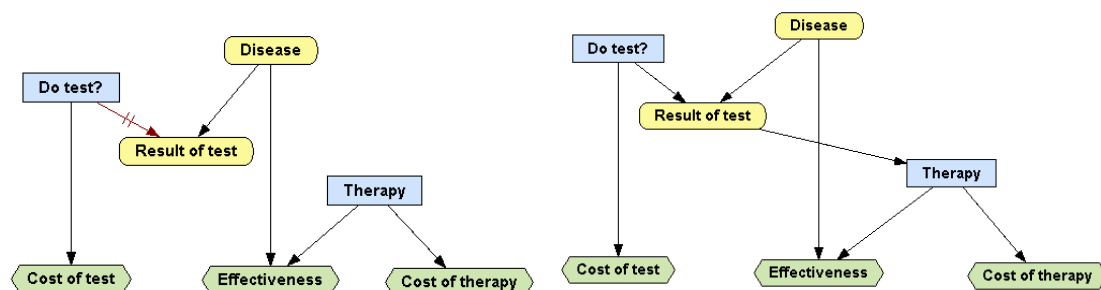


Figure 5.3: An decision analysis network, `DAN-decide-test-2therapies-CE.pgm` (left), and an influence diagram, `ID-decide-test-2therapies-CE.pgm` (right) for the problem in Example 5.1.

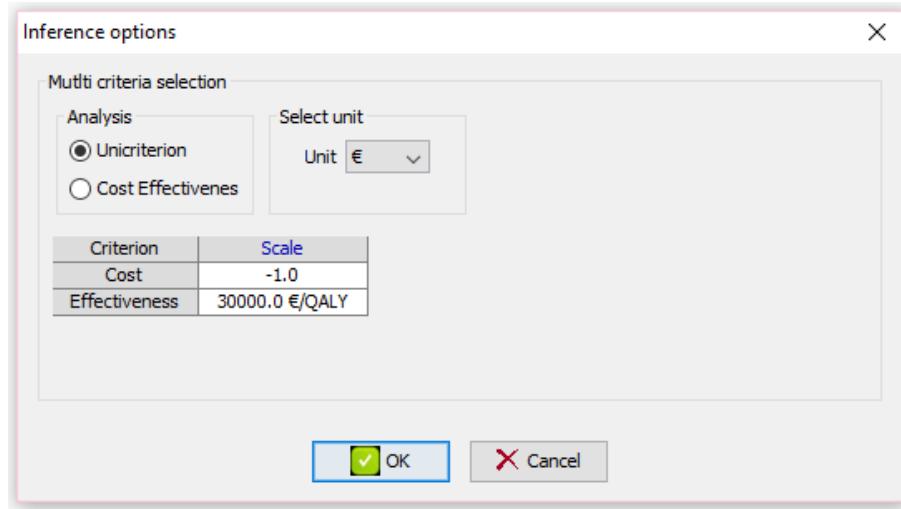


Figure 5.4: Unicriterion transformation.

## 5.2 Analysis of multicriteria networks

OpenMarkov offers two different approaches to solving multicriteria problems. The first one consists in joining all the criteria into a single one. For example, if we have two criteria, one measured in hours per man, and another measured in economic units, we can transform the first one by multiplying the hours per man by the mean salary of our team. With this transformation, we will have only economic units.

The second approach is called cost-effectiveness analysis. Cost-effectiveness analysis (CEA) is a form of multicriteria decision analysis frequently used in health economics [? ]. This approach keeps costs and effectiveness separated. The model needs to have exactly two criteria (one for cost and one for effectiveness) but OpenMarkov allows the user to use or even transform different criteria as cost or effectiveness. For example, let's suppose that we have three different criteria, one for effectiveness, another for economic costs and the last one measured in hours per man. We can transform the last criterion into economic costs and mark it's role in the evaluation as a normal cost.

### 5.2.1 Unicriterion analysis

Effectiveness is measured in quality-adjusted life years (QALYs) while costs are measured in euros (€). To solve Example 5.1 we are going to transform QALYs into euros with a rate of 30,000 €/QALY. The effectiveness will measure a positive outcome in euros but we need to state that costs should be a negative outcome. To do that we can transform the costs into negative.

Open Inference menu and click on Inference options. In the dialog observe that Unicriterion is selected by default. Select € as Unit, set the scale of effectiveness to 30,000 €/QALY and the scale of cost to -1. The result must see similar to shown in Figure 5.4.

At this point we are in the same situation than in chapters 3 and 4 and we can perform the same type of analysis.

Open the optimal strategy and observe that now, when the result of test is positive, the therapy to be applied is the first instead the second. That is because the cost of the second therapy is much higher than the cost of the first therapy and the difference in effectiveness is not enough. We can say that, for this situation, the first therapy is cost-effective compared with the second therapy. This information is more specific in equivalent decision tree. In the Figure 5.5 we can see that, the utility of apply the first therapy (when the result of test is positive) is €157,075 while the utility of apply the second therapy is only €152,007.

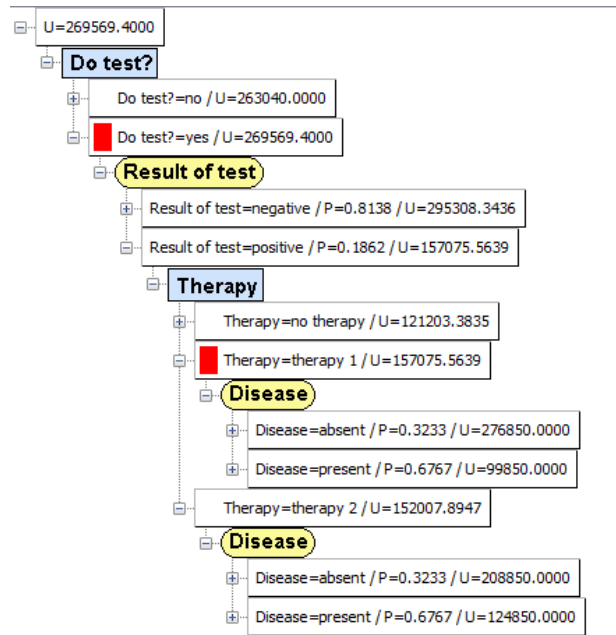


Figure 5.5: Equivalent decision tree for DAN-decide-test-2therapies-CE.pgm and ID-decide-test-2therapies-CE.pgm.

## 5.2.2 Cost-effectiveness analysis

Transforming effectiveness values into economic values is not realistic because the rate or scale depends on the decision maker. Cost-effectiveness analysis doesn't have this limitation and keeps costs and effectiveness separated. Click on the Cost-effectiveness analysis icon (📊) in the toolbar. Cost-effectiveness analysis can also be launched in OpenMarkov in Tools > Cost effectiveness analysis > Deterministic analysis (deterministic analysis as opposed to probabilistic analysis, described later in the tutorial).

The first dialog launched allows us to select the inference options for cost-effectiveness analysis. Be sure that the criteria have their proper roles selected. This selector lets the network to have more than two criteria and use one or more for each role in the analysis. The inference options will be similar to that shown in Figure 5.6.

Once the inference options are set up, OpenMarkov will ask us for the type of cost-effectiveness analysis. There are two main types of analysis as shown in Figure 5.7, global analysis and for a specific decision. Firstly we are going to analyze the global cost-effectiveness analysis.

### 5.2.2.1 Global cost-effectiveness analysis

Select Global and click OK. The dialog shows a table containing the cost and effectiveness values for all possible intervals of lambda. Lambda represents the rate of transformation of the effectiveness in economic units. In health economics is also called Willingness To Pay (WTP) as is the quantity that the decision maker is disposed to pay per each unit of effectiveness (in this case per each QALY).

Observe that instead of using a fixed WTP, OpenMarkov's cost-effectiveness analysis calculates these intervals for that there are a different optimal strategy (or intervention). Each intervention has specific costs and effectiveness. If we click on the cell of an intervention we can see the optimal strategy in its tree form. Observe that the optimal strategy when the WTP is between 11,171 and

Inference options

Mutlti criteria selection

Analysis

☐ Unicriterion

☒ Cost Effectiveness

Criterion	Role	Scale
Cost	Cost	1
Effectiveness	Effectiveness	1

OK Cancel

Figure 5.6: Inference options for cost-effectiveness analysis.

Scope selector - ID-decide-test-2therapi...

Scope

Type ☒ Global ☐ One decision

Decision Therapy

Scenario

OK Cancel

Figure 5.7: Types of cost-effectiveness analysis.

Cost-effectiveness intervals

$\lambda$ inf.	$\lambda$ sup.	Cost	Effectiveness	Intervention
0.0	11171.3	0.0	8.768	Do test? = no -> Therapy = no therapy
11171.3	33383.5	3874.0	9.11478	Do test? = yes -> IF Result of test = negative -> Therap...
33383.5	$+\infty$	13184.0	9.39366	Do test? = yes -> IF Result of test = negative -> Therap...

Close

Figure 5.8: Global cost-effectiveness analysis.

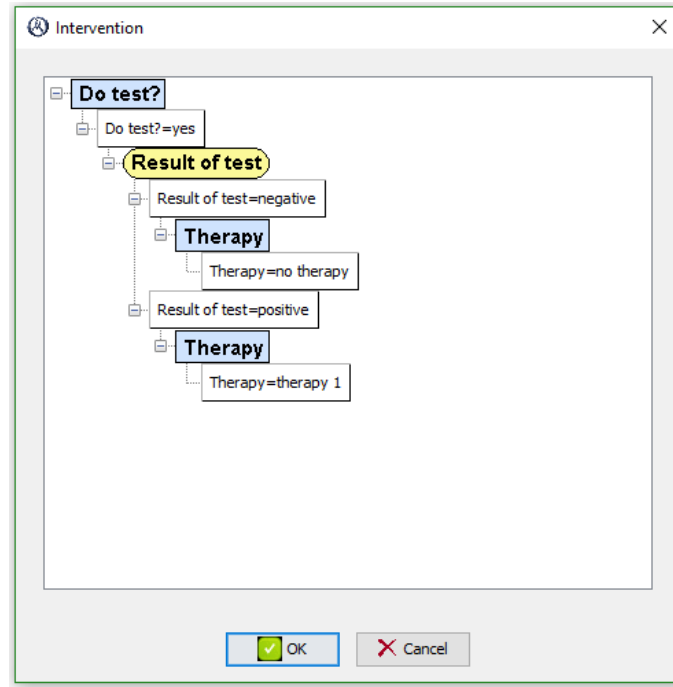


Figure 5.9: Optimal strategy when lambda is between 11,171 and 33,383 euros.

33,383 euros is the same than in the unicriterion analysis (shown in Figure 5.9). That's because the fixed lambda that we use for that analysis was 30,000 euros.

#### 5.2.2.2 Cost-effectiveness analysis analysis for a decision

Click on the icon **Cost-effectiveness analysis** and now select **One decision** as type of analysis. Now we must to select the decision for which we want to do the analysis. Select *Do test?* and click on OK button.

Observe that now, in the **Analysis tab**, the resulting table have two rows, one for *Do test? = no* and the other for *Do test? = yes*. In the left we can see a selector with different intervals. As in the case of global analysis we can click on any green cell to see the tree form of the intervention for that interval and that state of the decision.

CE Plane tab shows the cost-effectiveness plane, where cost is in the vertical axis and effectiveness is in the horizontal axis. Each intervention is represented by a point in the two-dimensional space. In our example, there is a point for *Do Test? = yes* and another for *Do Test? = no*, as can be seen in Figure 5.10.

The last tab named **Frontier interventions** shows the incremental cost-effectiveness ratio (ICER) between the therapies. If we have two therapies *A* and *B*, each one with a cost and effectiveness, this ratio is defined as  $ICER_{B,A} = \frac{C_B - C_A}{E_B - E_A}$  and allows to compare the two therapies. If the ICER is below WTP therapy *B* is said to be cost-effective with respect to therapy *A*.

Select the interval between 10,739 and 33,384 euros and observe that  $ICER_{yes,no} = € 11,171/QALY$ . This means that, for a WTP lower than this value is preferable not performing the test while, for greater WTP is better perform the test. We can see that this value is the same that the obtained in global cost-effectiveness analysis (see Figure 5.8).

Now we are going to analyze the cost-effectiveness for *Therapy*. Click on the icon **Cost-effectiveness analysis**, and select **One decision** as type of analysis. Select *Therapy* and check that the dialog requests a scenario. A scenario is a path from the root to a node. If we observe the equivalent decision tree (see Figure 5.5), we can see that before reaching *Therapy* we must pass through *Do test?* and *Result of test*.

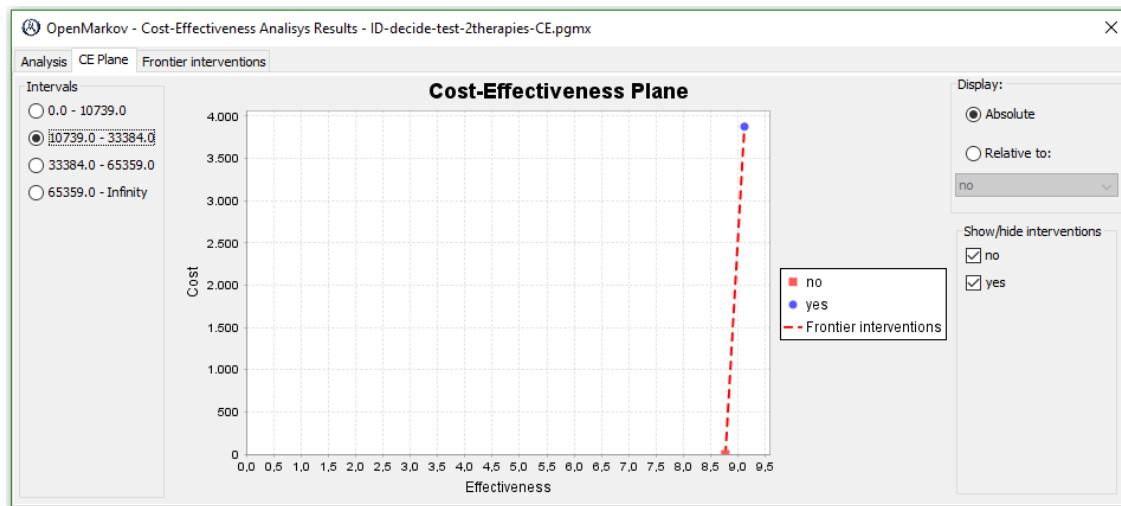


Figure 5.10: Cost-effectiveness plane when analyzing *Do Test?*.

Min	Max	Therapy
0	10,739	no therapy
10,739	33,384	therapy 1
33,384	$\infty$	therapy 2

Table 5.2: Optimal policy for therapy on the scenario *Do test? = yes* and *Result of test = positive*.

Select the scenario *Do test? = yes* and *Result of test = positive*. In CE Plane tab we can observe that, for this scenario, first therapy is expensive but give us more effectiveness than no applying any therapy and second therapy is costly but has more effectiveness than the first therapy. But, what is the optimal policy in each situation?. We can answer this question using ICER to compare the therapies. ICER only allows to compare one intervention with another but we have three possible therapies. OpenMarkov allows us to show or hide an intervention. By hiding an intervention it will not be compared. With this functionality in mind use show/hide functionality in the Frontier interventions tab to obtain the intervals of WTP and which therapy is better to apply in each case. Result is presented on Table 5.2.





## Chapter 6

# Cost-effectiveness with Markov influence diagrams

In this tutorial we will explain how to perform cost-effectiveness analysis with Markov influence diagrams (MIDs) [15]. We take as an example the well-known model by Chancellor et al. [7] for HIV/AIDS, which is now obsolete for clinical practice, but can still be used for didactic purposes. Thus, the web page for the book by Briggs et al. [5] contains an implementation of this model in Excel.<sup>1</sup> However, in this tutorial we implement it as an MID instead of a spreadsheet.

**Example 6.1.** There are two treatments for HIV infection. One of them (monotherapy) administers zidovudine all the time; the other (combined therapy) initially applies both zidovudine and lamivudine but after two years, when the latter becomes ineffective, continues with zidovudine alone. In this model, 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 cycle length of the model is one year. The transition probabilities, assumed to be time-independent, are shown in Tables 6.1 and 6.2. The sum of the probabilities in each column is 1 because the patient must either remain in the same state or transition to one of the other three. The zeros in these tables mean that a patient never regresses to less severe states. In particular, a patient who dies remains in state *dead* for ever. Table 6.3 shows the annual costs associated with different states. The model is evaluated for a time span of 20 years (after which more than 95 per cent of patients are expected to have died). As in the original paper, costs are discounted at 6% and effectiveness at 2% per year.

↓destination   origin →	<i>state A</i>	<i>state B</i>	<i>state C</i>	<i>dead</i>
<i>state A</i>	0.721	0.000	0.000	0.000
<i>state B</i>	0.202	0.581	0.000	0.000
<i>state C</i>	0.067	0.407	0.750	0.000
<i>dead</i>	0.010	0.012	0.250	1.000

Table 6.1: Transition probabilities for monotherapy.

↓destination   origin →	<i>state A</i>	<i>state B</i>	<i>state C</i>	<i>dead</i>
<i>state A</i>	0.858	0.000	0.000	0.000
<i>state B</i>	0.103	0.787	0.000	0.000
<i>state C</i>	0.034	0.207	0.873	0.000
<i>dead</i>	0.005	0.006	0.127	1.000

Table 6.2: Transition probabilities for the first three years of combined therapy.

---


<sup>1</sup><http://www.herc.ox.ac.uk/pubs/books/decision/Ex25sol>.

	<i>state A</i>	<i>state B</i>	<i>state C</i>	<i>dead</i>
<i>monotherapy</i>	£ 5034.00	£ 5330.00	£ 11285.00	£ 0.00
<i>combined therapy</i>	£ 7119.50	£ 7415.50	£ 13370.50	£ 0.00

Table 6.3: Annual costs per states and therapy type.


## 6.1 Construction of the MID



### 6.1.1 Creation of the network

Click on the icon Create a new network () in the first toolbar. In the Network properties dialog, select *MID* as the Network type.



Double-click on the empty network panel to open the Network properties dialog. In the Advanced tab, click Decision criteria and use standard criteria button to select Cost-effectiveness (£/QALY)..

### 6.1.2 Construction of the graph

Click on the icon Insert decision nodes () . Double-click on the empty network panel to create a new node; in the Node properties dialog, write *Therapy* as the Name. Go to the Domain tab and set the states of the variable to *monotherapy* and *combined therapy*. Please note that we are following the convention of capitalizing the names of the variables and using lower case names for their states. Click OK to save the changes.

Next, click on the icon Insert chance node () . Double-click on the network pane to create a new node and name it *Time in treatment* and set its Time slice to 0. As a Comment you may write *How long the patient has been under treatment when the model starts*. 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 to save the changes. Right-click on this node and choose Create node in next slice to create the node *Time in treatment [1]*. Select the Insert link tool () and drag the mouse from the node *Time in treatment [0]* to *Time in treatment [1]* to create a link between these two nodes, as in Figure 6.1.

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*, *state B*, *state C* and *dead*. Save the changes. Right-click on the node and choose Create node in next slice to create the node *State [1]*. As the state of the patient in cycle 1 depends on the state on the previous cycle, on the therapy applied, and on the duration of the therapy (because lamivudine is given only in the combined therapy for the first two years), add links from *State [0]*, *Therapy*, and *Time in treatment [1]* to *State [1]*, as in Figure 6.1.

Click on the icon Insert value node () and create the node *Cost [0]*; before closing the Node properties dialog, observe that the Decision criterion assigned to this node is *cost* because by default the decision criterion for value nodes is the first of the decision criteria of the network. As the cost per cycle depends on the state patient and on the therapy applied, draw links from *State [0]*, *Time in treatment [0]*, and *Therapy* to *Cost [0]*. Finally, create the value node *Life years [0]* and set its Decision criterion to *effectiveness*. Draw a link from *State [0]* to *Life years [0]* as the amount of life accrued in a cycle only depends on the patient's state. If you wish to move any node, use the Select object tool () .

Save the network as `MID-Chancellor.pgm`.

### 6.1.3 Specification of the parameters of the model

After introducing the structural information of the MID, it is time to set parameters of the model, i.e., the conditional probabilities and the utility functions. First, we introduce the prior probability of *State [0]*, that is, the proportion of patients in each state when the model starts. We assume that initially all patients are in *state A*. In order to specify this, right-click on the node *State [0]* and chose Edit probability... in the contextual menu (or alt-click on this node) and select *Table* as

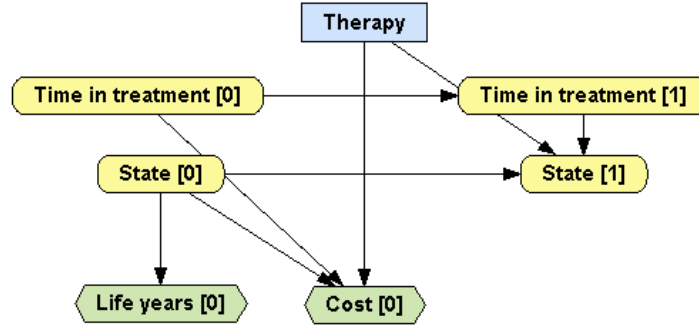


Figure 6.1: MID for Chancellor's HIV model.

the **Relation type**. In the table, set the probability of *state A* to 1 and observe that the probabilities of the other states become 0. Click OK.

Analogously, select **Edit probability...** for the node *Time in treatment [1]* and select *Cycle-LengthShift* as the **Relation type**. This means that the value of *Time in treatment* increases in each cycle by as much as a cycle length. Therefore, setting the value of *Time in treatment [0]* value to 0 years will make *Time in treatment [1]* take the value 1 year, and so forth. If the cycle length were half a year, the increase would be 0.5. Now, we must introduce the initial value of *Time in treatment [0]*. Select **Edit probability** for the node, set *Delta* as the **Relation type** and 0 as the numeric value.

We then introduce the probability of *State [1]* conditioned on its parents, *State [0]*, *Therapy*, and *Time in treatment [1]*. We might encode it as a table, but given that some of the values of that table repeat themselves, it is more appropriate to use a tree: on the contextual menu of *State [1]* select **Edit probability...** and choose *Tree/ADD* as the **Relation type**. (We will explain the meaning of ADD very soon.) A small tree will appear inside an editor—see Figure 6.2. The root of the tree is the first node that you declared as parent of *State [1]*, i.e., it depends on which link you drew first. If this root node is not *Therapy*, right-click on it, click **Change variable** and select *Therapy*. The tree then has two branches, *monotherapy* and *combined therapy*. Remember that when *monotherapy* is applied, the probability of the patient being in a state in cycle 1 depends on the state in cycle 0, as indicated by Table 6.1; it does not depend on the duration of the treatment. Consequently, right-click on the text  $P(\text{State [1]})$  in the *monotherapy* branch, select **Add variables to potential**, check the box for *State [0]*, and click OK. Observe that the text has changed into  $P(\text{State [1]} | \text{State [0]})$ . Right-click on this text, select **Edit potential**, chose *Table* as the **Relation Type**, introduce the transition probabilities specified in Table 6.1, and click OK. This branch should look like in Figure 6.2.

When the treatment is *combined therapy*, the probability of *State [1]* depends not only on *State [0]* but also *Time in treatment [1]*, because the patient receives zidovudine plus lamivudine for three years and then zidovudine alone, as in the case of monotherapy. Therefore, the probability of *State [1]* for *combined therapy* might be encoded as a three-dimensional table. However, it is better to represent it as a tree with two branches: one for the first three cycles and another one for the rest of the time, as in Figure 6.2. To introduce these branches, right-click on *Therapy=combined therapy* and select **Add subtree** and *Time in treatment [1]*. This creates 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 2 as the threshold value. Notice that the radio button **Included in the first interval** is checked, which makes the threshold belong to the first sub-interval. This means that the original interval will be partitioned into  $[0, 2]$  and  $(2, \infty)$ . Obviously, if we selected the option **Included in the second interval** the partition would be  $\{[0, 2), [2, \infty)\}$ . As we did in the *monotherapy* branch, add the variable *State [0]* to the potential  $P(\text{State [1]})$ , right-click on the text  $P(\text{State [1]} | \text{State [0]})$  in this branch, choose **Edit potential**, select *Table* as the **Relation Type**, introduce the transition probabilities specified in Table 6.2, and click OK. This branch should look like in Figure 6.2.

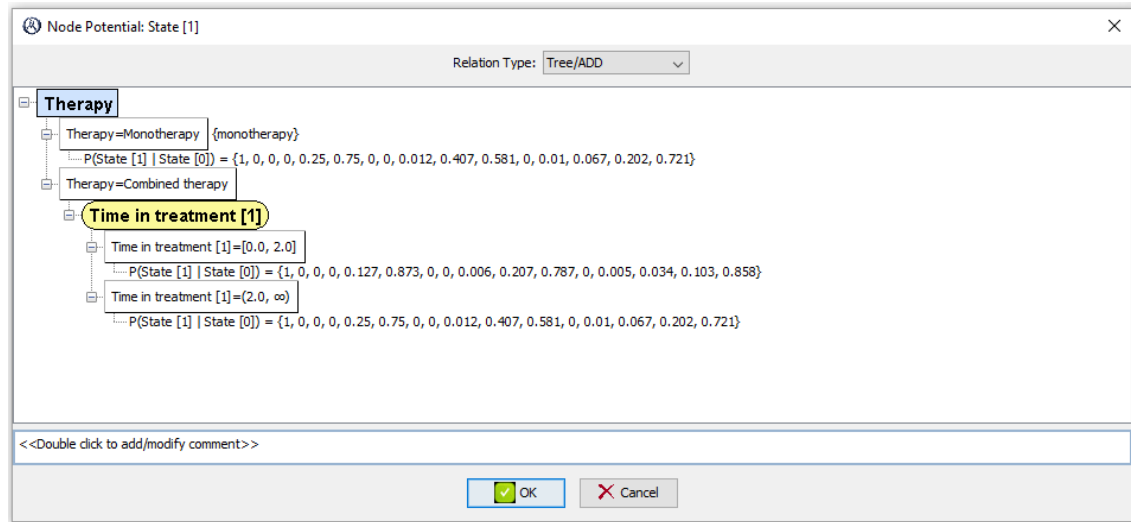


Figure 6.2: Conditional probability for *State [1]*, represented as a tree.

Finally, the probability of *State [1]* for *combined therapy* after the second year is supposed to be the same as that of *monotherapy*. Therefore, it would be possible to proceed as above and copy again the values of Table 6.1 into a new table inside this probability tree, as in Figure 6.2. One disadvantage of this representation is the need to introduce the same parameters twice; if there is uncertain about a parameter, its second-order probability distribution will also have to be introduced twice. A more important disadvantage is that if a parameter changes in the light of new evidence, it will have to be updated in two places, with the risk of inconsistencies between the branches of the tree. Finally, the main drawback of this representation is that when performing sensitivity analyses OpenMarkov will sample those parameters independently, without realizing that each parameter of the model appears twice in this probability tree.

Fortunately, it is possible to tell OpenMarkov that two branches of the tree are the same. Properly speaking, in this case we do not have a tree, but an *algebraic decision diagram* (ADD) [2], because a node in the “tree” belongs to more than one branches. Given that trees are a particular case of ADDs, OpenMarkov and ProbModelXML have only one type of potential for both of them, called *Tree/ADD*—see [?] for a more detailed explanation.

The ADD for our example can be encoded as follows. Right-click on the node *Therapy=Monotherapy* and select **Set label**, name it as “*monotherapy*”. Right-click on the node *Time in treatment [1]=(2.0 ∞)*, select **Set reference** and *monotherapy*, and click **OK**. The result will be as shown in Figure 6.3

Save the network.

The utility potential for the node *Cost [0]* can also be specified as a tree/ADD. The process is very similar: in the contextual menu of this node, select **Edit utility**, select *Tree/ADD* as the **Relation type**. Right-click on the root of the tree, *State [0]*, select **Change variable** and choose *Therapy*. In the branch *monotherapy*, right-click on *Cost [0]*, select **Add variables to potential** and choose *State [0]*. Right-click on *Cost [0](State [0])*, select **Edit potential**, choose *Exact* as **Relation type**, introduce the cost associated each state, as specified in the first line of Table 6.3, and click **OK**.

Right-click on *Therapy=combined therapy*, select **Add subtree** and *Time in treatment [0]*. Right-click on *Time in treatment [0]=[0.0, ∞)* and select **Split interval**. Set 2 as the **Threshold value** and observe that the option **Included in first interval** is selected. In the branch for the interval  $[0.0, 2.0]$ , right-click on *U(Cost [0])*, add *State [0]* as a variable, right-click on *Cost [0](State [0])*, select **Edit potential**, choose *Exact* as **Relation type**, introduce the cost associated each state, as specified in the second line of Table 6.3, and click **OK**.

Right-click on the node *Therapy=monotherapy* and set the **label** *cost-monotherapy*. Right-click on the node for the interval  $(2.0, ∞)$  and set the **reference** *cost-monotherapy*. The resulting

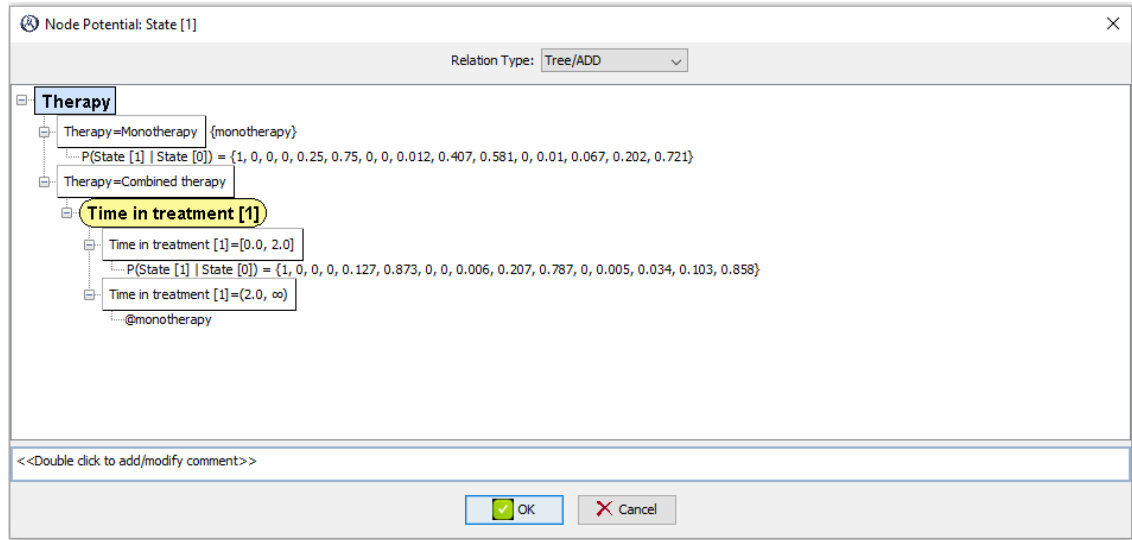


Figure 6.3: Conditional probability for *State [1]*, represented as an ADD.

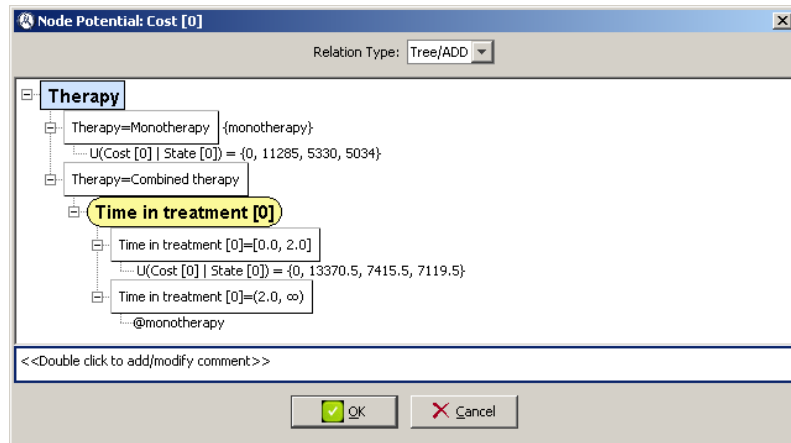


Figure 6.4: Value of *Cost [0]* as a function of its parents.

potential should resemble that of Figure 6.4. Click OK to save this potential.

The value of *Life years [0]* given its only parent *State [0]* is very simple: it is 1 for all states except for *dead*, which is 0. This potential can be specified as a table containing four values, but we will specify it as a tree. Right-click on *Life years [0]* and choose **Edit potential** from the contextual menu. Select *Tree/ADD* as the **Relation type**. A tree with *State [0]* as the root node will appear. Right-click on the *state A* branch and choose **Add states to branch** in the contextual menu. In the dialog that will appear, check the boxes for *state B* and *state C*, and click OK. Right-clicking on the node *Life years [0]* in this branch, set the **relation type** to *Exact* and write 1 in the only cell in the table. Next, edit the potential of the *dead* branch in a similar way introducing 0 in the only cell of this table. The resulting potential should look like the one in Figure 6.5

Save the network again.

## 6.2 Evaluation of the MID

The model contains all the relevant qualitative and quantitative information and is now ready for evaluation. There are currently two ways of evaluating the model in OpenMarkov:

1. show the temporal evolution of a variable, and

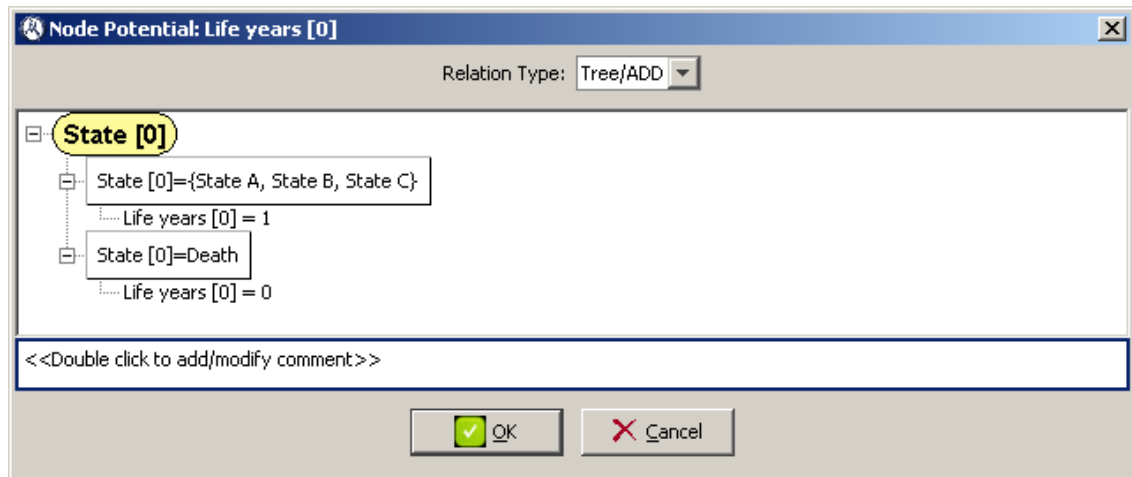


Figure 6.5: The effectiveness, represented by node *Life years [0]*, depends on the patient's state, *State [0]*.

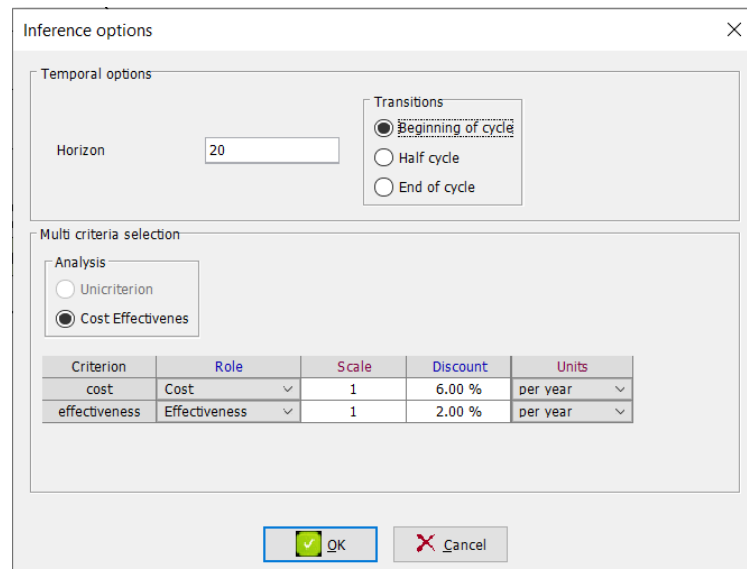


Figure 6.6: Cost-effectiveness analysis options.

2. perform a cost-effectiveness analysis (CEA).

Before explaining these options in further detail, we describe the dialog used to configure both types of evaluation.

### 6.2.1 Configuring the evaluation options

A dialog with the configurable options for cost-effectiveness analysis will pop up, as shown in Figure 6.6. These options include:

- **Horizon:** The number of cycles the CEA will take into account.
- **Transitions:** In discrete-time Markov models, transitions can happen either at the beginning, at the end, or in the middle of a cycle. The option chosen affects the computations [17, 16].
- **Discounts:** A discount can be applied for each criterion, specified either per year or per cycle. If the discount is specified annually and the cycle length is different from one year,

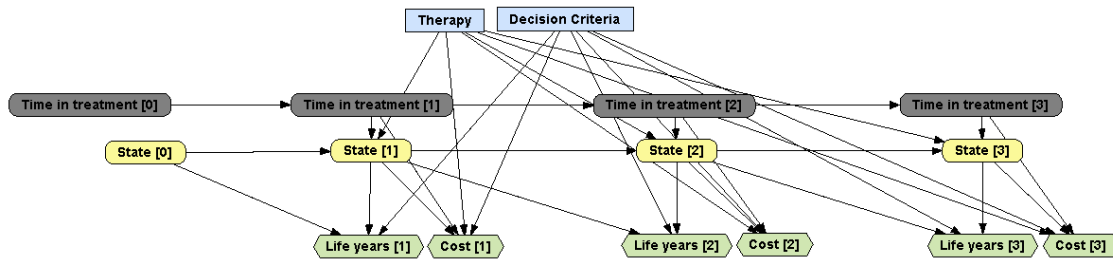


Figure 6.7: MID model expanded to three time slices

OpenMarkov automatically adjusts it; for example, if the cycle length were 3 months and the user specified a 6% annual discount, OpenMarkov would apply a discount rate of 1.535 per cycle, because  $(1 - 0.01535)^4 = 1 - 0.06$ .

## 6.2.2 Expanding the model

As shown in our example, when building a temporal model, i.e., a model that represents the evolution of a system over time, only the sufficient information is entered as to be able to expand the network to a variable number of cycles by creating instances of new time slices. Usually, it is enough to represent the first time slice and those variables in the second time slice that depend on the first to be able to expand a model.

In order to expand a network in OpenMarkov, right-click on an empty spot in the network panel and choose the option **Expand Network CE** in the contextual network. The options dialog described in the previous section will pop up and after entering the appropriate values (especially the number of cycles to expand) and clicking OK, a new network panel will be opened containing the expanded MID. Figure 6.7 shows the MID for the Chancellor model, expanded to three cycles.

## 6.2.3 Temporal evolution of variables

Analyzing the evolution of a variable over time can give us very useful information. In OpenMarkov, given a strategy, the evolution of a chance or value variable can be plotted. In order to do so, first impose a policy (as described in Section 4.4) to each decision node in the model, then right-click on the node whose temporal evolution the user wants to analyze and choose **Temporal Evolution** in the contextual menu. After introducing the evaluation parameters in the dialog described in Section 6.2.1, the temporal evaluation of the variable will be represented in a dialog with two tabs. The first will show a plot as the one in Figure 6.8, while the other will contain a table with the values of the variable in each time slice.

## 6.2.4 Cost-effectiveness analysis

Cost-effectiveness analysis can be launched in OpenMarkov in **Tools > Cost-effectiveness analysis > Deterministic analysis** (as opposed to probabilistic analysis, described below).

The results for deterministic cost-effectiveness analysis are separated in three tabs. The first tab shows a table containing the cost and effectiveness values for every strategy.

The second tab shows the cost-effectiveness plane, where cost is in the vertical axis and effectiveness is in the horizontal axis. Each intervention is represented by a point in the two-dimensional space. In our example, there is a point for monotherapy and another for combined therapy, as can be seen in Figure 6.10.

The third tab shows a list of frontier interventions along with the Incremental Cost-Effectiveness

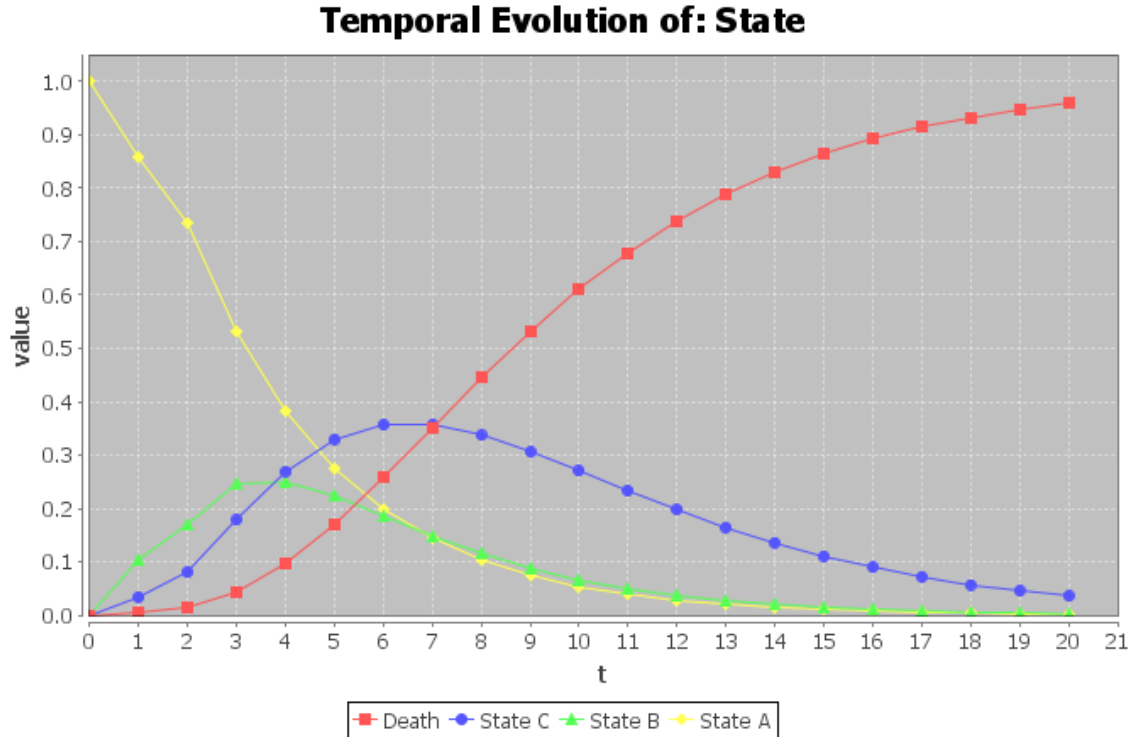


Figure 6.8: Evolution of variable *State* over time for combined therapy.

Cost-Effectiveness Analysis Results		
Analysis	CE Plane	Frontier interventions
Therapy	Combined therapy	Monotherapy
effective...	8.325	7.538
cost	50562.046	44613.851

Figure 6.9: Cost-effectiveness analysis for Chancellor's model.

Ratio (ICER). In our example, no intervention dominates the other, so both are frontier interventions.

### 6.3 Probabilistic sensitivity analysis

Usually the parameters of a model (such as transition probabilities) are estimated (based on data or an expert's knowledge) and therefore are subject to uncertainty. In order to represent this uncertainty, probability distributions are used instead of point estimates. In this section of the tutorial we will further develop our running example by setting probability distributions to some of the parameters and then we will run a probabilistic analysis based on a number of simulations.

**Example 6.2.** Following with our running example of the HIV/AIDS model, let us consider that instead of having point estimates for transition probabilities in monotherapy we use Dirichlet distributions with the parameters in Table 6.4.

Let us also consider to model direct medical and community care costs associated with each state with gamma distributions. As little information was given in the original article about the variance of costs, we will assume that the standard error is equal to the mean. Direct medical and community care costs are shown in Table 6.5. Drug costs will remain fixed: £ 2278 for zidovudine and £ 2086 for lamivudine.



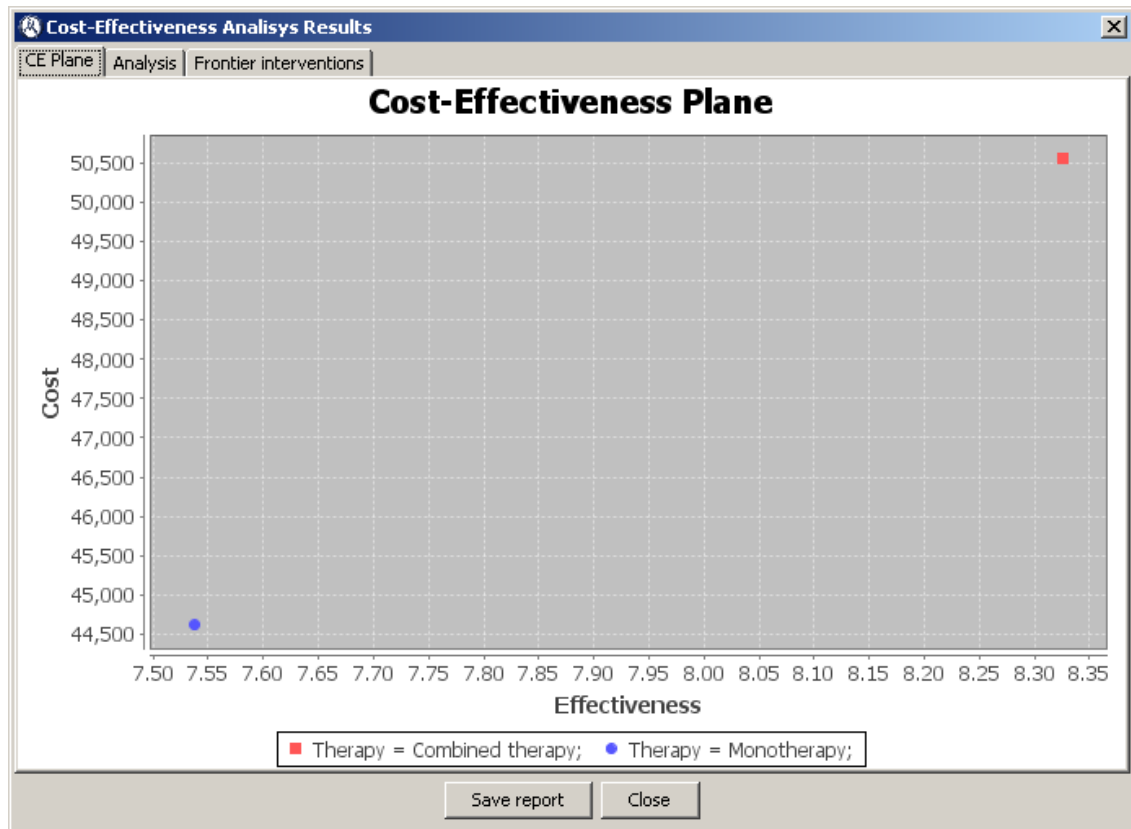


Figure 6.10: Cost-effectiveness plane for Chancellor's model.

Cost-Effectiveness Analysis Results			
Analysis	CE Plane	Frontier interventions	
Strategy	Effectiveness	Cost	ICER
Therapy = Monotherapy;	7.538	44614	
Therapy = Combined therapy;	8.325	50562	7550

Figure 6.11: Frontier interventions for Chancellor's model.

### 6.3.1 Introducing uncertainty in the model

First, the transition probabilities need to be modified to specify that these should be drawn from a Dirichlet distribution. Right-click on the *State [1]* node and choose **Edit probability...** in the contextual menu. The dialog that will pop up should look like Figure 6.2. Right-click on the text  $P(\text{State [1]}|\text{State [0]})$  just below the *Monotherapy* branch and choose **Edit potential**. Right-click on any cell in the *state A* column and choose *Assign uncertainty* in the contextual menu and a dialog similar to that in Figure 6.12 will pop up. Click on the cell in the *Distribution* column in the first row and choose *Dirichlet* from the drop-down list. Enter the distribution's parameter as

	<i>state A</i>	<i>state B</i>	<i>state C</i>
<i>state A</i>	1251	-	-
<i>state B</i>	350	731	-
<i>state C</i>	116	512	1312
<i>dead</i>	17	15	437

Table 6.4: withDirichlet distribution parameters for transition probabilities

	<i>state A</i>	<i>state B</i>	<i>state C</i>
<i>Direct medical costs</i>	£ 1701	£ 1774	£ 6948
<i>Community care costs</i>	£ 1055	£ 1278	£ 2059

Table 6.5: Direct medical and community care cost means associated with each state

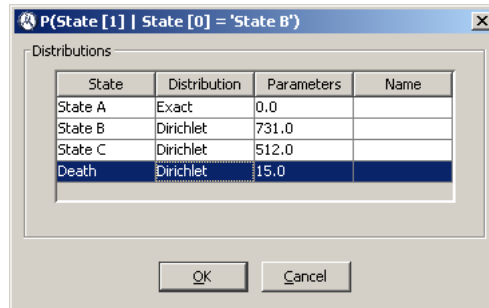


Figure 6.12: Transition probabilities as Dirichlet distributions for  $P(\text{State } [1] \mid \text{State } [0] = \text{state B})$ .

specified in Table 6.4 (in this case 1251). Introduce the parameters for all transition probabilities, making sure the values specified as dashes in Table 6.4 are represented by Exact distributions whose parameter is zero (as in Figure 6.12).

Second, the *Cost* variable needs to be split into three different variables representing *Community care cost*, *Direct medical cost* and *Drug cost*. Note that *Drug cost* will have the same parents as the old *Cost* node had, that is, *Therapy*, *Time in treatment [0]* and *State [0]*, but *Community care cost* and *Direct medical cost* will only have one parent, *State [0]* as these are supposed to be the same for both therapies. The resulting MID should look as in Figure 6.12.

Defining the utility function for *Drug cost [0]* is very similar to how we defined the utility

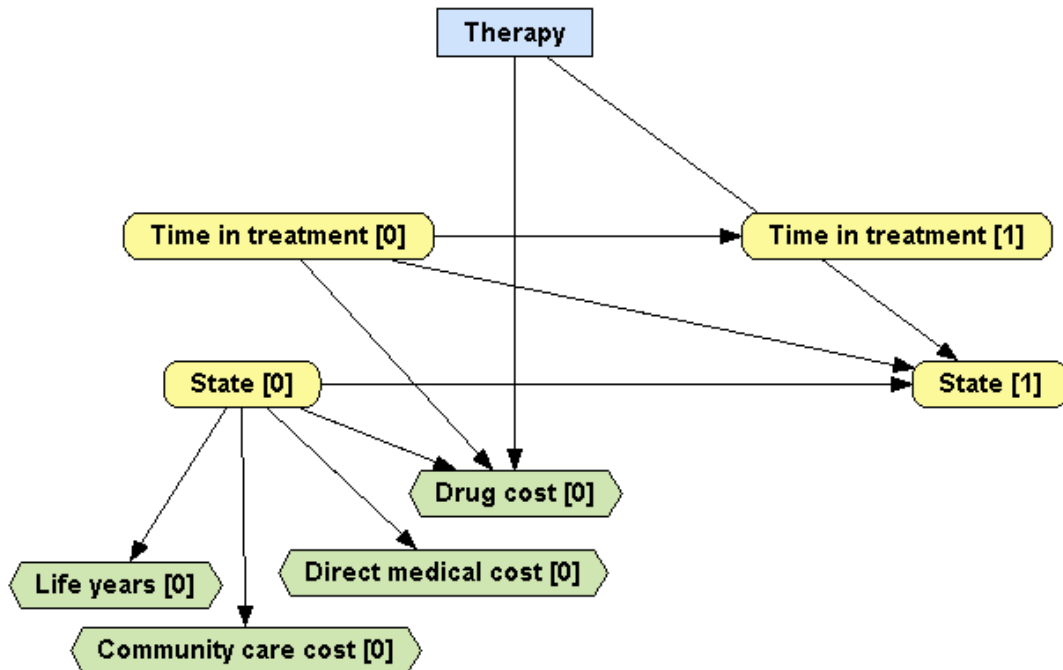


Figure 6.13: Chancellor's model modified for probabilistic sensitivity analysis.

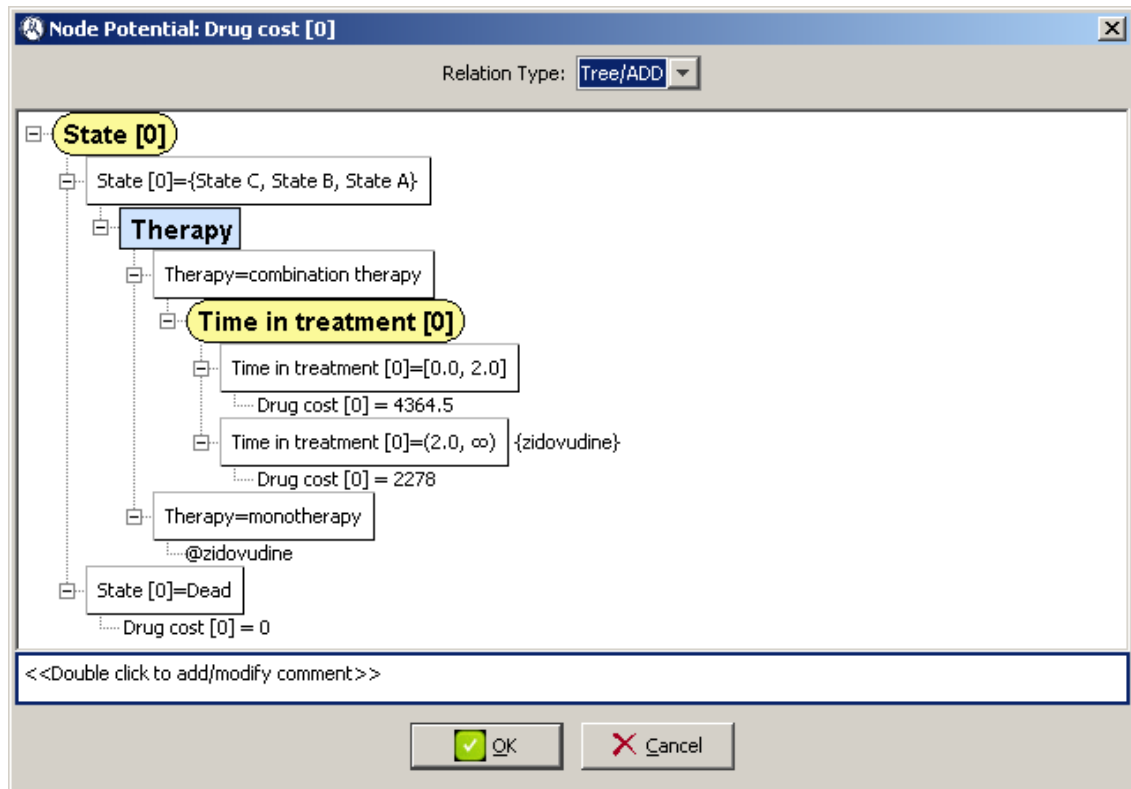


Figure 6.14: Utility function for *Drug cost [0]*.

function for the old *Cost [0]* node in Section 6.1.3, as it contains no uncertainty. Following the same steps as in the mentioned section and with the help of Figure 6.14 you should be able to define *Drug cost [0]*'s utility function.

Let us now use gamma distributions to define community care and direct medical costs. Right-click on *Direct medical cost [0]* and choose **Edit utility...** from the contextual menu. Next, select *Table* from the **Relation Type** drop down list. Right-click on the only cell of the *state A* column and choose **Assign uncertainty** from the contextual menu. Choose "*Gamma-mv*" from the distribution type drop down list in the only line in the table that pops up. **OpenMarkov** will then ask for the parameters of the distribution  $\mu$  and  $\sigma$ , i.e., the mean and the standard error, which in this case are both 1701 (as we have assumed that the standard error is equal to the mean). Do the same for *state B* and *state C* using the appropriate values of the first row of Table 6.5. The fourth state, *dead*, can be ignored as there is no cost attached to that state. Repeat these steps for *Community care cost [0]*, this time using the values of the second row of Table 6.5.

After doing this, the model is ready for probabilistic analysis.

### 6.3.2 Running the analysis

To launch the probabilistic analysis, go to **Tools > Cost effectiveness analysis > Sensitivity analysis**. This will summon a dialog very similar to the one in Figure 6.6 but with an additional panel at the bottom to determine the number of simulations that will be run in the analysis. Clicking **OK** will launch the analysis, which will usually take a few seconds; the execution time will depend on the size of the network, the number of cycles and the number of simulations.

The results of the analysis will be shown in a tabbed pane. The analysis summary and frontier interventions panels will be very similar to the ones described in Section 6.2.4 and are therefore no longer described here. However, the cost-effectiveness plane will have substantial differences, as the result of every simulation will be represented in the plane, and therefore each intervention will be represented as a cloud of points instead of a single point. Figure 6.15 shows the cost-effectiveness

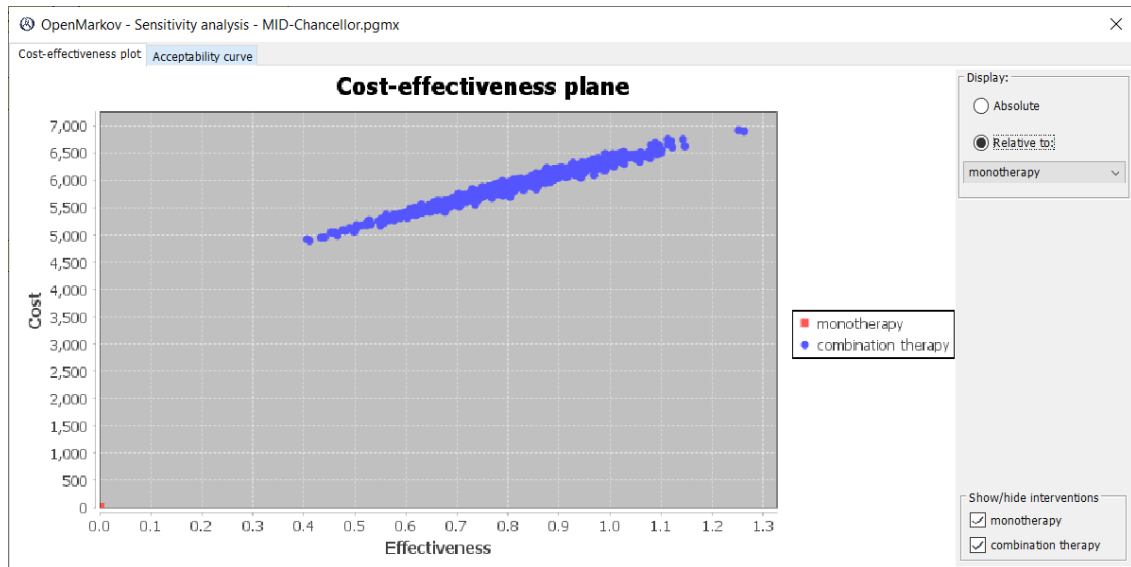


Figure 6.15: Probabilistic sensitivity analysis: cost-effectiveness plane for Chancellor's model.

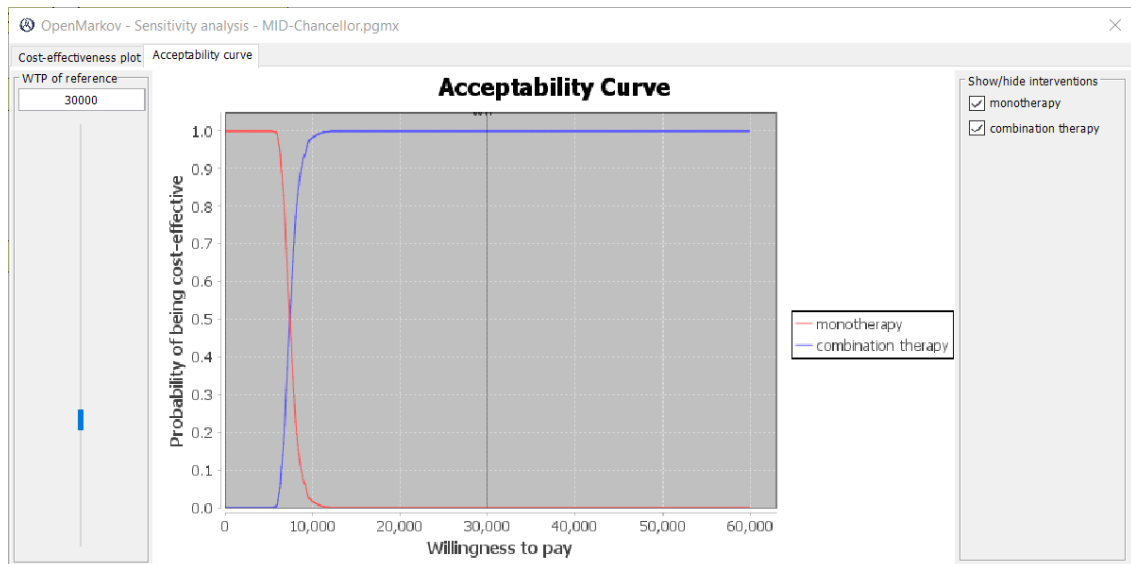


Figure 6.16: Probabilistic sensitivity analysis: acceptability curve for Chancellor's model

plane resulting from the probabilistic analysis of the Chancellor's MID.

In addition to the panels that were present in the deterministic analysis, there are two new panels that show the results of probabilistic analysis. The first is the cost-effectiveness acceptability curve (CEAC), which represents the probability of a treatment being cost-effective (i.e. acceptable) depending on the willingness to pay, that is, the amount of money we are willing to pay for a unit in effectiveness. Figure 6.16 shows the CEAC resulting from the probabilistic analysis of the Chancellor's MID. Note that there is a curve for each possible intervention, and that the probability of the less costly intervention is 1 when the willingness to pay is zero (cost is all that matters). This probability decreases and cuts the curve of the costlier (but more effective) intervention exactly at the ICER. Finally, the more effective intervention's probability of cost-effectiveness rises to one as the willingness to pay approaches infinity.

Finally, the panel of Expected Value of Perfect Information represents the expected costs of

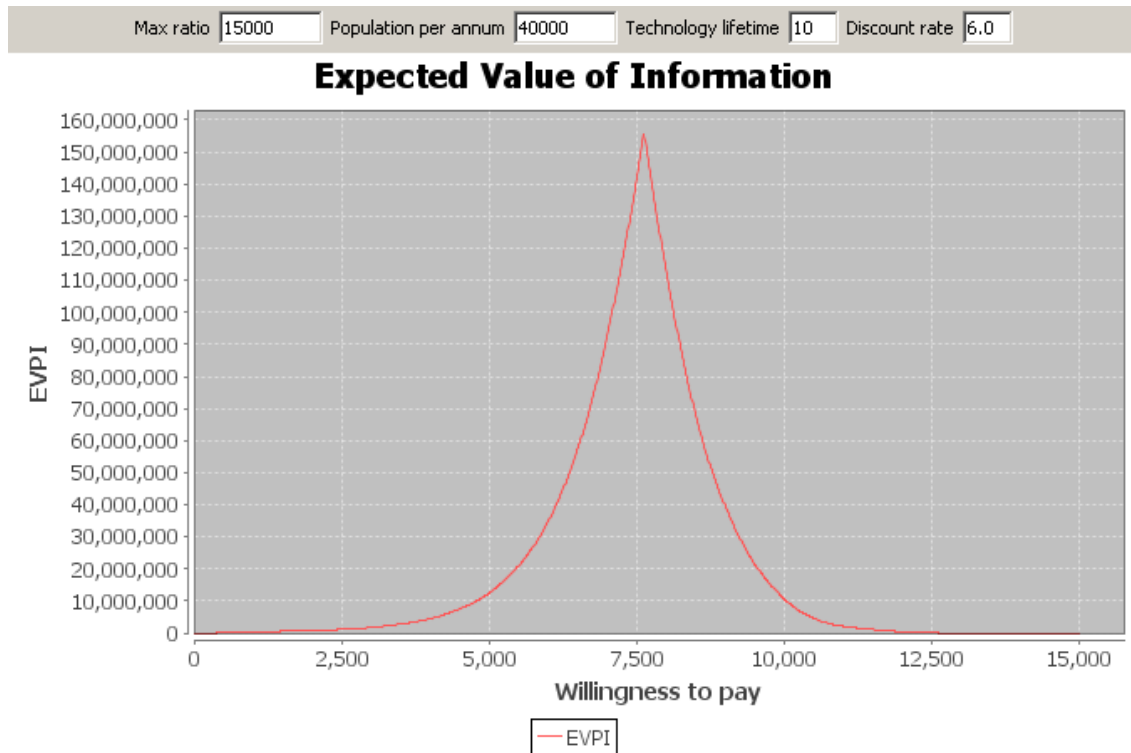


Figure 6.17: EVPI in Chancellor's model

uncertainty, as perfect information can eliminate the possibility of making the wrong decision. It is also the maximum that the health care system should be willing to pay for additional research. EVPI is calculated for the total population of patients who stand to benefit from additional information over the expected lifetime of the technology, although the EVPI for future patients is discounted at a certain rate. The graph can be customized through the parameters at the top of the chart: the max ratio (the maximum value of cost-effectiveness threshold to take into account), the population of patients, the lifetime of the technology and the discount rate.

## **Acknowledgment**

This work has been supported by the Spanish Government under grants TIN2006-11152, TIN2009-09158, PI13/02446, TIN2016-77206-R, and PID2019-110686RB-I00, and co-funded by 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] Bahar, R. I., Frohm, E. A., Gaona, C. M., et al. (1993). Algebraic decision diagrams and their applications. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD'93)*, page 188–191, Santa Clara, CA.
- [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*, page 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], page 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., Arias, M., and Pérez-Martín, J. (2021). Cost-effectiveness analysis with unordered decisions. *Artificial Intelligence in Medicine*, 117:102064.
- [12] Díez, F. J., Luque, M., and Bermejo, I. (2018). Decision analysis networks. *International Journal of Approximate Reasoning*, 96:1–17.
- [13] Díez, F. J., Luque, M., and König, C. (2012). Decision analysis networks. In [6], page 83–90.
- [14] Díez, F. J., Luque, M., König, C., and Bermejo, I. (2014). Decision analysis networks. Technical Report CISIAD-14-01, UNED, Madrid, Spain.
- [15] 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.
- [16] Elbasha, E. H. and Chhatwal, J. (2016a). Myths and misconceptions of within-cycle correction: a guide for modelers and decision makers. *Pharmacoeconomics*, 34:13–22.

- [17] Elbasha, E. H. and Chhatwal, J. (2016b). Theoretical foundations and practical applications of within-cycle correction methods. *Medical Decision Making*, 36:115–131.
- [18] Ezawa, K. (1998). Evidence propagation and value of evidence on influence diagrams. *Operations Research*, 46:73–83.
- [19] Herskovits, E. (1990). *Computer-based probabilistic-network construction*. PhD thesis, Dept. Computer Science, Stanford University, STAN-CS-90-1367.
- [20] 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*, page 719–762. Strategic Decisions Group, Menlo Park, CA.
- [21] Howard, R. A. and Matheson, J. E. (2005). Influence diagrams. *Decision Analysis*, 2:127–143.
- [22] Kjærulff, U. and Madsen, A. L. (2010). *Bayesian Networks and Influence Diagrams: A Guide to Construction and Analysis*. Springer, New York.
- [23] Koller, D. and Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, Cambridge, MA.
- [24] Korb, K. B. and Nicholson, A. E. (2010). *Bayesian Artificial Intelligence*. CRC Press, 2nd edition.
- [25] 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.
- [26] 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.
- [27] Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press.
- [28] Neapolitan, R. E. (2004). *Learning Bayesian Networks*. Prentice-Hall, Upper Saddle River, NJ.
- [29] Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA.
- [30] Raiffa, H. (1968). *Decision Analysis. Introductory Lectures on Choices under Uncertainty*. Addison-Wesley, Reading, MA.
- [31] Raiffa, H. and Schlaifer, R. (1961). *Applied Statistical Decision Theory*. John Wiley & Sons, Cambridge, MA.
- [32] Spirtes, P., Glymour, C., and Scheines, R. (2000). *Causation, Prediction and Search*. The MIT Press, Cambridge, Massachusetts, 2nd edition.