Universiteit Utrecht

Master Thesis Technical Artificial Intelligence

Explaining the reasoning of Bayesian networks

with intermediate nodes and clusters

Author:

Jordy van Leersum (3344878)
J.vanLeersum@uu.nl

Primary Supervisor:

dr. S. Renooij
S.Renooij@uu.nl

Secondary Supervisors:

S.T. Timmer, MSc
S.T.Timmer@uu.nl

prof. dr. ir. L.C. van der Gaag
L.C.vanderGaag@uu.nl

May 2, 2015

Abstract

This thesis presents a method to create an explanation for the reasoning of Bayesian networks in order to explain the most probable value for the node of interest. The first part identifies a set of intermediate nodes which are nodes that can explain the most probable value of the node of interest. These intermediate nodes act as a funnel for the node of interest and summarize the evidence nodes. This set of intermediate nodes is found by the Edmonds-Karp algorithm in combination with one of three weight-assignment functions. One of these functions is purely based on the structure of the graph of the Bayesian network and the other two different functions take the probability distributions of the Bayesian network into account as well. The second part gives arguments that explain the most probable values of the *intermediate* nodes by creating clusters, which are set of nodes that include at least one evidence node of the Bayesian network. The actual explanation takes the output of these two methods to construct an explanation which is an interactive web page, where the explanation of the most probable values of the nodes are supported by verbal expressions and graphical figures.

Contents

1 Introduction **5**

2 Bayesian Networks **6**
2.1 Bayesian network . . . 6
2.2 Semantics of Bayesian networks . . . 7

3 Related Work **10**
3.1 Explanation . . . 10
3.1.1 Content . . . 10
3.1.2 Communication . . . 11
3.1.3 Adaptation . . . 11
3.2 Explanation methods & Projects . . . 12
3.2.1 Explanation of evidence . . . 12
3.2.2 Explanation of the model . . . 12
3.2.3 Explanation of reasoning . . . 13

4 General Method **15**

5 Intermediate Nodes **17**
5.1 Defining intermediate nodes . . . 17
5.2 Maximum Flow and Minimum Cut . . . 17
5.3 Applying Edmonds-Karp to Bayesian Networks . . . 20
5.4 Minimum cuts vs Intermediate nodes . . . 22
5.5 Weight-assignment functions . . . 22
5.5.1 One-for-All . . . 22
5.5.2 Hellinger influence . . . 23
5.5.3 Mutual Information . . . 26
5.6 Experiments for finding intermediate nodes . . . 28
5.6.1 Research questions . . . 29
5.6.2 Methods . . . 29
5.6.3 Experiments . . . 33
5.6.4 Conclusion on finding sets of intermediate nodes . . . 37

6 Clusters **39**
6.1 Generating clusters . . . 39
6.2 Mutual information of a cluster . . . 41
6.2.1 Mutual information for a cluster . . . 41
6.2.2 Alternative mutual information . . . 41
6.3 Clusters in the Oesophageal cancer network . . . 43
6.3.1 Location . . . 43
6.3.2 Shape . . . 45
6.3.3 Length . . . 46
6.4 Summary of the creation of clusters . . . 48

7 The Explanation **51**
7.1 Verbal Expressions 51
7.2 Representation and User-interaction 51
7.3 Explanation of Oesophageal cancer network 52
7.4 Summary on the explanation of the reasoning 55

8 Summary and Conclusion **61**

9 Discussion and Future Research **63**

1 Introduction

Most decisions in life are based on a lot of factors, like the environment or your state of mind. Some decisions are made easily, but a lot of decisions depend on the state of a very complex environment with tons of variables. In order to make a decent decision, you should take into account that variables may not always be in the state that you assume them to be and thus a decision can become very complicated. This is what for instance a doctor has to cope with on a daily basis. It takes years of training to even become slightly comfortable with the medical field and the decisions that arise with it. Even then some decisions are made without being absolutely sure if it is the correct one. So wouldn't it be great if modern day techniques can assist for instance doctors with those decisions.

A well designed Bayesian network can model an environment and its variables. A Bayesian network is a graphical model which captures those variables as nodes and the relations among those nodes as edges. The network captures a probability distribution and after analyzing such a network we can say how likely a certain state of a node will be. In the first part of the following chapter we will go into more detail about the mathematical properties of Bayesian networks.

So what is the problem of applying Bayesian networks in for instance the medical field? Say that a doctor wants to know how likely it is that a patient has a certain condition. A Bayesian network is capable of giving some probability for this condition, when it has been given evidence about the patient and its condition. But a doctor is not only interested in the answer, he also wants to understand how an answer comes about. So the network should be able to explain its outcome and show why it gives a certain answer. The expert can then argue if the outcome of the Bayesian network is valid. Only this is where the problem lies, because understanding the reasoning behind a Bayesian network on such a level of detail presumes that the person should have some sort of mathematical background and it also takes time of which doctors already have little. So in this thesis we come up with a way to give an explanation about the reasoning of a Bayesian network.

In this thesis we present several methods in order to create an explanation for the reasoning of a Bayesian network to explain the most probable value for the node of interest. We will start by presenting background on Bayesian networks and their properties in Chapter 2. In Chapter 3 we will discuss previous work done on explaining Bayesian networks, because we are not the first to address this problem. We will see that we can explain several aspects of Bayesian networks and we will find out how our research fits in. In Chapter 4 we will start with our own method by first giving an overview of the methods we are using and then describe how we will translate the results into an explanation. In the first method, in Chapter 5, we are going to identify a set of intermediate nodes, which is a set of nodes in the Bayesian network that we take to explain the most probable value for the node of interest. In order to find these nodes we will use the *Maximum-Flow-Minimum-Cut* theorem and the associated *Edmonds-Karp* algorithm. We will present three weight-assignment functions that accompany the *Edmonds-Karp* algorithm, where the first function purely focusses on the structure of the graph of the Bayesian network and the other two different functions take the probability distribution into account as well.

Chapter 6 of this thesis presents the second method that explains the most probable values of the intermediate nodes. We present a method that creates clusters for each of the intermediate nodes. These clusters are subsets of all nodes of the Bayesian network including at least one evidence node. Because an intermediate node can have more than one cluster we will order them based on the *Mutual information* between a cluster and the intermediate node. The ordering determines which cluster is best to explain first. We will use these ordered clusters and the earlier found intermediate nodes in the actual explanation introduced in Chapter 7. We end with a conclusion (Chapter 8) and discussion (Chapter 9).

2 Bayesian Networks

In order to give an explanation about the reasoning of a Bayesian network and give more insight in why a network gives a certain result, we need some background on these networks. In the following we will describe the mathematical meaning of a Bayesian network, the properties of such a network and give some insight in how it calculates probabilities.

2.1 Bayesian network

A Bayesian network [22] is a probabilistic graphical model that takes the form of a directed acyclic graph (DAG) combined with a probability distribution. Formally a Bayesian network is a tuple $B = (G, P)$, where G is an acyclic graph and P is a probability distribution. The graph G is in itself a tuple $(V(G), A(G))$ with $V(G)$ being a set of nodes $\{V_1, \ldots, V_n\}$ and $A(G)$ a set of edges $A(G) \subseteq V(G) \times V(G)$.

The nodes $V(G)$ in a Bayesian network represent the statistical variables together with a conditional probability distribution over these nodes given its parents $P(V_i|\pi(V_i))$. For nodes $V(G)$ without any parents the conditional probability distribution is equal to its prior probability distribution $P(V_i|\emptyset) = P(V_i)$. A Bayesian network is a representation of the joint probability distribution on a set of statistical variables.

$$P(V_1, \ldots, V_n) = \prod_i^n P(V_i|\pi(V_i)). \tag{1}$$

We are typically interested in the posterior probabilities, i.e. the probability of a state after entering evidence in the network. The probability assessment for the variables can be obtained in several ways, for instance from statical data, literature on a domain or from experts. Algorithms associated with such a network are able to compute any $P(H|E)$ for hypothesis node H given evidence for E. Such a network is able to encode entire probability distributions over all possible combinations of variables. The probability $P(H|E)$ is essentially computed by Bayes' theorem:

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)}. \tag{2}$$

To calculate probability distributions, algorithms have been designed to do inference on such networks. Probabilistic inference means that available evidence is propagated and the probability distributions of the nodes in the network will be updated. Probabilistic inference in Bayesian networks is in general NP-hard [3], but existing algorithms can compute probabilities in polynomial time in networks of constrained topology. In order to make the above definitions more clear the next part covers a small example of a Bayesian Network.

The Wet Grass example

Figure 1 shows the graph of a small Bayesian network, consisting of three nodes and two edges. In this example we are interested in the probability that the grass will be wet tomorrow. This can be either the result of the sprinkler or the result of the rain or it might be a result of both of them. First we assume that the state of each node can only be *true* or *false*, so either it rained or it did not. In more complex networks it is possible to define more than two states for each node.

For each node it is possible to assign probabilities to those values. For instance the weather report stated that there is only ten percent chance of rain tomorrow, so the probabilities will be $P(Rain = true) = 0.1$ and $P(Rain = false) = 0.9$. Because you want the grass to be wet you programmed the sprinkler in such a way that it should moisten the grass tomorrow, but it is always possible that it malfunctions and so we still have the chance $P(Sprinkler = false) = 0.05$.

Sprinkler		
$P(true)$	=	0.95
$P(false)$	=	0.05

Rain		
$P(true)$	=	0.1
$P(false)$	=	0.9

Wet Grass					
Sprinkler		$true$	$true$	$false$	$false$
Rain		$true$	$false$	$true$	$false$
$P(true)$	=	1	1	1	0
$P(false)$	=	0	0	0	1

Table 1: Probability tables for the example Bayesian network

Assigning probabilities to the values of the node Wet Grass works slightly different, because the result is affected by the possible values of its two parent nodes. For each combination of values of the three variables we need to define the probability. When we look at Table 1 we can see the values assigned to these combinations. If it rained or the sprinkler went on the probability that the grass is wet is exactly 1 and otherwise it is 0. For the ease of the example we assume that are no other possibilities. Therefore the grass will not be wet when the sprinkler stayed off and it did not rain.

When you enter this example in a Bayesian network tool like *GeNIe* [15], we can calculate for example the probability of the grass being wet from these probability tables. In this case $P(WetGrass = true)$ is 0.955 and this is probably what you would expect. Even though there is a small chance of rain, the grass will probably still be wet because of the sprinkler and this is exactly what the Bayesian network encodes.

In Bayesian networks we have the possibility to enter observations for certain variables. This is done when we have full knowledge of the value of a node in the network. When we imagine a Bayesian network for the medical field, one of the nodes could encode the gender of a person, which we can know for certain and therefore can be entered as evidence.

2.2 Semantics of Bayesian networks

A Bayesian network contains an acyclic digraph which means the edges have directions and it is not possible to have cycles in the graph. When there is an edge between two nodes it shows that they can influence each other's probabilities [22]. When there is evidence for nodes present in the network this can have direct effects on the probability distributions of the other nodes. When we look at a whole chain of nodes, the presence of evidence can make it so that this influence is prevented. This is called blocking, which means that the influence of the evidence node along one side of a chain does not change any of the probability distributions for the nodes on the other side of the chain. To give more insight we will look at the definition of blocking.

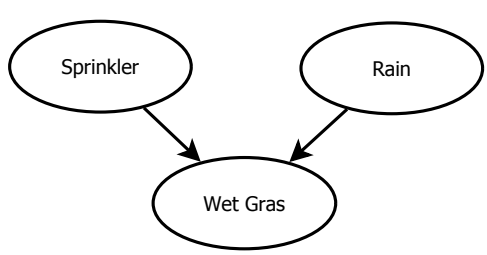

Figure 1: Graph for an example Bayesian Network

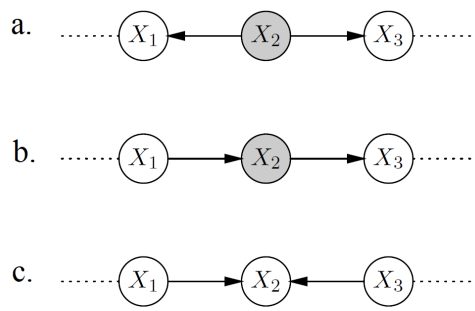

Figure 2: Case distinction of chain blocking [33].

We take the extended definition from [32]:

Definition 2.1.
Let $G = (V(G), A(G))$ be an acyclic digraph and let s be a chain in G between $V_i \in V(G)$ and $V_j \in V(G)$. The chain s is blocked by a (possibly empty) set of nodes $W \subseteq V(G)$ if $V_i \in W$ or $V_j \in W$, or s contains three consecutive nodes X_1, X_2, X_3, for which (at least) one of the following conditions hold:

a. *edges (X_2, X_1) and (X_2, X_3) are on chain s, and $X_2 \subseteq W$;*

b. *edges (X_1, X_2) and (X_2, X_3) are on chain s, and $X_2 \subseteq W$;*

c. *edges (X_1, X_2) and (X_3, X_2) are on chain s, and $\sigma^*(X_2) \cap W = \emptyset$.*

Here $\sigma^*(X_2)$ is a set of all descendants of X_2 including X_2 itself and an edge (X_i, X_j) represents a directed edge from node X_i to node X_j

In Figure 2 we can see a graphical representation of the conditions mentioned in Definition 2.1. In the first two chains we see that X_2 is grey, which means this node is part of the blocking set W. Note that case b is symmetric and that we can reverse all edges to end up in the same general case. In the third case we see that X_2 and all of its descendants (in this case the empty set $\emptyset$) are not in the blocking set W. The concept of blocking generalises to entire sets of nodes if they satisfy the proposed conditions.

With the concept of blocking of single chains we define the notion of *d-separation*, where we are interested in blocking of all the chains between two sets of nodes. So we are interested in a set of nodes which block all chains between these two sets of nodes and therefore we can say that they have no influence on the probability distributions of each other. We define d-separation as follows:

Definition 2.2.
Let $G = (V(G), A(G))$ be an acyclic digraph. Let X, Y, $Z \subseteq V(G)$ be sets of nodes in G. The set of nodes Z is said to d-separate the sets of nodes X and Y in G, denoted as $< X|Z|Y >^d_G$, if for each node $V_i \in X$ and each node $V_j \in Y$ every chain from V_i to V_j in G is blocked by Z.

With the d-separation criterium it is possible to determine whether or not nodes may influence each others posterior probabilities given evidence entered in the Bayesian network. When we establish that two nodes are d-separated it means that they are *conditionally independent* given the nodes in the blocking set. Bayesian networks use their graph to represent the independence relation among the variables associated with the

nodes, and d-separation serves for reading independence statements from the graph. In many occasions the edges represent causal meaning, but in principle this it not the case.

Now that we have some knowledge of Bayesian networks, we will continue with previous research done on the explanation of these networks. We will take a look at the explanation of the evidence, the explanation of the model and explanation of the reasoning.

3 Related Work

Finding a good way to explain the meaning and reasoning of a Bayesian network has been researched before. In fact there have been several projects on this subject and these show that the explanation can be done from several perspectives. The purpose of this chapter is to show the different ways with which we can explain Bayesian networks and indicate which direction this thesis is going to follow. This section is mainly based on the work of Lacave, who has written a review paper [20] in which she explains the work that has been done on this subject already. The first part will be about the explanation done on the evidence, the model and the reasoning and the different ways of looking at Bayesian networks. After that we will describe some explanation methods that have already been developed.

3.1 Explanation

The paper of Lacave [20] makes a summary of several projects done over the past years. One of its purposes is to make the reader more comfortable with the meaning of the word *explanation* and its properties. Lacave states that this word indicates 'understandability' and 'satisfaction'. In Table 2 we repeat a summary given by Lacave [20]. In the following we will explain this table in more detail.

Content	Focus	Evidence / Model / Reasoning
	Purpose	Description / Comprehension
	Level	Micro / Macro
	Causality	Causal / Non-causal
Communication	User-system interaction	Menu / Predefined questions / Natural language
	Presentation	Text / Graphics / Multimedia
	Expression of probability	Numeric / Linguistic / Both
Adaptation	Knowledge of the domain	No model / Scale / Dynamic model
	Knowledge of the reasoning	No model / Scale / Dynamic model
	Level of detail	Fixed / Threshold / Auto

Table 2: Properties of explanation methods [20].

As we can see in Table 2 the first column consists of three rows, namely Content, Communication and Adaptation. These subjects represent the three major subjects in which we can divide the explanation of a Bayesian network. The second column describes the major subjects from the first column in several individual parts and the third describes the properties associated with these subjects. Content describes which part of the Bayesian we are focusing on in our explanation and in how much detail we are going to look at it. The subject of Communication is about how to represent the explanation and Adaptation is for making sure your explanation is understandable for the reader. So for a good explanation we will need properties from every row in the table. In the following sections we will present three major subjects, Content, Communication and Adaptation in more detail.

3.1.1 Content

A Bayesian network is always created for a certain environment and describes a certain situation. If we take a closer look at such a network we can distinguish three major groups, namely the evidence, the model and

the reasoning on which we can *focus*[1] in order to explain their content. When looking at evidence we are interested in determining which values of the unobserved variables justify the available evidence. A process which is called abduction *(section 3.2.1)*.

Observation of evidence and outcome can differ from case to case, but the model stays the same. So explaining the model can be done in a fairly static way. For instance with the use of graphs or just verbally. Sometimes it requires a more dynamical approach in order to give a proper explanation of the Bayesian network, where we focus on the obtained results, the not obtained results or the hypothetical results (what if one or more variables had been observed). Explaining the reasoning behind the network can help the experts and non-experts understand what is actually happening and they can find out if the given results make sense.

The *purpose* of the explanation is twofold. One is to give a description and explain the background or the conclusion. The other purpose is to give the user more comprehension about the things that are going on, make the user understand what the implications of these conclusions are, or maybe even make him understand what has to change in order to get the appropriate results.

According to Sember and Zukerman [26] it is possible to classify expert systems at two different *levels*. First we have the micro level were we look at what happens in a node, when for instance a neighbor's probability is updated due to newly observed evidence. So here we are interested in specific variables and their effect. The other level is the macro level which looks at the main reasoning that is done by the Bayesian network: a certain configuration of evidence nodes indicates a certain outcome.

A Bayesian network is a mathematical model to describe dependencies among nodes, but for humans it is also a way to describe *causal* relations. An edge from node A to node B can often be attributed a causal meaning, when we look at node A this can be the reason that a value of node B holds. Major reasons to choose a model based on causality is that humans tend to understand these relations [30][8] and therefore the use of causality helps as a heuristic for the construction of such a model [24][16].

3.1.2 Communication

User-interaction can be a great help in making users understand what is going on. For instance when a user is able to get an explanation after selecting some nodes in which he is interested, as used in the MYCIN project [29]. Here the idea is that a program can handle questions from users. Here you can think of questions like 'how did the system arrive at this conclusion?'. So a program can be passive by just showing its conclusions in more detail or be more active in trying to find the answer the user is looking for.

Giving an answer is one thing, but the format in which you *display the explanation* can also be very important. Users with no background in mathematics probably are not interested in the exact numbers, but probably want a more verbal explanation. Whether you are an expert or not, graphs can also be an easy way of making things more clear. When giving an explanation you can also vary with the *expression of probabilities*, which you can display numerically, linguistically or qualitatively.

3.1.3 Adaptation

Another important factor for creating an explanation is the user that you are going to present it to. His background can be an important factor, because when he does not have much experience with high level mathematics, it may be better to avoid it. Another important fact is the *user's knowledge about the domain*. The user may or may not be acquainted with details of the domain and therefore the explanation can differ significantly. When creating a program to generate an explanation it is important to consider if it is for a static user model, where the knowledge of the domain is always above a certain level or that it is dynamic

[1] The emphasized words in these section refer to Table 2.

and all kinds of users will be interacting with the program. This also holds for the *user's knowledge about the reasoning model*.

This all is closely related to the *level of detail* which is going to be used for the actual explanation. Instead of taking into account the knowledge about the domain of the expert, we can let the expert himself make the program adapt to his level of knowledge. By making the amount of detail that is shown variable and letting the user handle this, the explanation can be tailored to his own wishes.

3.2 Explanation methods & Projects

Now that we have more insight in the ways that we can explain the information contained inside a Bayesian network, we are going to take a look at some tools that already have been developed. Our review is also based on the same work of Lacave [20]. The tools can be divided in three major subjects, namely explanation of evidence, explanation of the model and explanation of reasoning. This directly refers back to Table 2, where this same distinction is made on the first row.

3.2.1 Explanation of evidence

One way of explaining the evidence in Bayesian networks is using *abduction*. Some of the nodes can be observed values from which we know their actual state for sure. For the other nodes we want to know the most probable explanation (MPE), so the configuration W of unobserved nodes with the maximum a posteriori probability $P(W|e)$. When we want a configuration W, that includes all nodes, we are talking about total abduction, but if we are only interested in a subset of the nodes it is called partial abduction.

The goal of abduction is to give a description of the situation that is the most probable. Such a configuration would indicate the origin of a problem due to the causal relations in the network. Abduction could also work on a network that is not based on causalities, because it is a mathematical concept. With abduction we have no distinction between the micro or macro level and the developed methods have almost no user-interaction, because there are focused on finding the MPE.

This subject has been researched several times, for instance by Pearl [23] with his π-λ propagation. This is a method that is based on the idea that for each value x of a given variable X in configuration W, there is a best explanation for the rest of the variables $W \setminus X$. You can find the best value of X for the MPE by finding the best explanation for $W \setminus X$ and then choosing the best value of X. Santos [25] proposed another method for total abduction where he transforms a Bayesian network into an equivalent linear restriction system $L(W)$. Making sure only relevant variables for the explanation are being selected is done by Shimony [28], who proposed three definitions of irrelevance in partial abduction. A completely different approach is done by Charniak and Shimony [2], who proved that the abduction of Bayesian networks is equal to the problem of finding the minimum cost assignments for the variables in a weighted Boolean function acyclic directed graph (WBFDAG). Gámez [14] came up with algorithms that do abductive inference in Bayesian networks in order to find the most likely explanation for the observed evidence.

3.2.2 Explanation of the model

Giving an explanation of the network can be done statically and the easiest way to do this is by giving a graphical representation of the actual network. HUGIN [18] was the first program to visually represent the graph, but soon GeNIe [15] followed. In the last one it is also possible to define submodels in order to keep larger networks more understandable. These programs improve the description of the model, which is done with graphs, menus and visualizing probabilities.

The program DIAVAL was developed by Dièz [7], who combined the explanation of both the model and the reasoning. This program is designed to have lots of interaction with the user, which is mainly done

with different windows and menus. It can distinguish several types of nodes and links corresponding to the different types of causal interaction, like the noisy OR. Another way of representing information is to turn the probabilistic expressions into linguistic expressions. Druzdzel [8] has created a model for this in which causality plays a important role. Instead of giving the users only the exact probability he describes for instance: "cold can be the cause of sneezing which is very likely ($P = 0.9$)". This method is both comprehensive and descriptive, where he tries to give a linguistic explanation at the micro level, but also shows the actual probabilities.

3.2.3 Explanation of reasoning

In the following part we describe what research has been done in the field of explaining the reasoning of the network. In these projects the emphasis lies on making clear how certain results come about. In this section we will go into more detail than we did in the previous ones, because the subject of explaining the reasoning of Bayesian network is the main goal of this thesis. We will go through the projects in chronological order.

The first method is based on the explanation of local updates. The explanation uses the work of Pearl [22], who showed that we can obtain the posterior probability for some variable B by calculating:

$$Bel(b) = \alpha\lambda(b)\pi(b). \tag{3}$$

Here $\lambda(b)$ is the diagnostic support and $\pi(b)$ is the causal support. Sember and Zukerman [27] came up with a method to justify the value of $Bel(b)$ in terms of $\lambda(b)$ and $\pi(b)$. For instance when the user knows that the causal support $\pi(b)$ has increased he may suspect that $Bel(b)$ increases. When this condition is met in the program it will respond in a positive fashion, but when it differs it will identify the value of $\lambda(b)$ that causes the deviation.

Another method was developed by Elsaesser [13], which generates linguistic explanations that are based on Poyla's *shaded inductive patterns*. These are heuristic rules for describing changes in probabilities. This method is very much based on the micro level, so for instance: "If we have an edge from node A to node B and node B is true, then the existence of A is more credible". It also describes probabilities with values between for instance 0.99 and 0.91 as "almost certain" or "highly probable". In other words it describes probabilities in verbal statements.

Suermondt developed the INSITE method [30][31] in which he tries to find the evidence which influences the posterior probability of the hypothesis H as well as the paths through which the evidence flows. The purpose of INSITE is the comprehension of reasoning by interaction with the user, which is led by way of menus and such. Explanations are presented as a combination of graphics and text and probabilities are expressed by both numbers as linguistic expressions. The influence of evidence E for hypothesis H is measured by a cost-function. For this the notion of cross entropy is being used:

$$H(P(H|E); P(H)) = \sum_{h_i \in H} \left[P(h_i|E) \log \frac{P(h_i|E)}{P(h_i)} \right]. \tag{4}$$

This function is used to measure the influence of evidence e on some variable H. Here h_i are the possible states the variable H may have. Sensitivity analysis is used to compute the cost of leaving out evidence in order to state how influential the findings are.

Another method developed by Suermondt finds nodes called *Knots* [19] in the network[2]. In order to understand what knots are, recall Section 2.2, where we have seen that we can distinguish *chains* in Bayesian networks. These chains are paths that go from an evidence node E to a node of interest H via the edges

[2]Here we review the concept of knots following the definition in Koiter [19].

in the network. A Bayesian network is (or can be) a multiply connected network, which means that there can be more than one direct chain from the same evidence E to the node of interest H. A direct chain in multiply connected graphs can have the same evidence E and hypothesis H as another chain and therefore it is possible to have overlap. To identify nodes that are in overlapping chains Suermondt defines so called knots. Knots are used to avoid discussing some subchains multiple times. A knot in a set S of direct chains from findings E_i to hypothesis H, given the set of evidence E, is a node K_j such that this node K_j is in every chain in S, and $K_j \cup (E \setminus E_i)$ d-separates E_i from H.

Another method by Druzdzel and Henrion [8][10][17] creates scenarios, which are assignments of values of variables in such a way that they are relevant for a certain conclusion and they form a causal story. A scenario could contain all nodes in the network, but in a lot of cases not all nodes are relevant. Relevant nodes are the nodes that affect the posterior probability of the hypothesis, so the value of the node of interest. The explanation consists of the evidence that is relevant, the scenario which is most probable for our hypothesis and a comparison between scenarios.

Druzdzel and Henrion [9][11][12] also developed another method which transforms the given causal Bayesian network into a qualitative probabilistic network (QPN) [35][36]. Relations between nodes are indicated as positive (+), negative (−), zero (0) or unknown (?). With more than two nodes we can determine synergies among those sets of nodes based on their status. The advantage of these networks is that they simplify the reasoning, because we are not dealing with numerical values anymore, but with relations that can influence each other.

The last method we will be mentioning is the graphical method by Madigan, Mosurski and Almond [6] which shows the propagated evidence in a causal Bayesian network. The influence of evidence e on a binary node H is measured by Good's evidence weight:

$$W(H : e) = \log \left(\frac{P(e|h)}{P(e|\bar{h})} \right). \tag{5}$$

This method displays its own graph with varying colors, thickness of nodes and links and including special kinds of links. The purpose is to investigate the relative importance of the findings and get an understanding of the influence of these variables.

In this chapter we have seen what has already been researched for finding and giving a good explanation for a Bayesian network. We can focus on the model, the evidence or the reasoning of such networks. We have seen several methods that have been developed in order to come up with a solution fit to a certain situation. The level of detail or the way in which it is represented can differ in a lot of ways. In the upcoming chapter we will focus of our own method. In this chapter we will give an overview of our own method and we will describe what we do in the rest of this thesis.

4 General Method

We have seen some background on how to explain a Bayesian network, now we will start by elaborating on our own method for creating an explanation of the reasoning of a Bayesian network. In this chapter we want to give the reader an overall idea of what we are going to do. We will be focusing on explaining the reasoning of a Bayesian network. Our method consists of two major parts, where the first one is finding a set of intermediate nodes. This set should be a small set of nodes for which it holds that they contain important information for the explanation of the reasoning. These nodes will function as an explanation for the most probable value of the node of interest. The second part of our method is based on the creation of clusters in order to help explain the most probable values of the intermediate nodes. For each of the intermediate nodes we will construct one of more clusters that can aid us in understanding the most probable value of the associated intermediate node.

We assume that the Bayesian networks we are going to use in our method represent causal reasoning. In short this means that, when we have an edge from node A to node B, the most probable value of node A is (or might be) the reason for the most probable value of node B. This is what we have seen in Chapter 2 with the *Wet Grass* example. These causal reasoning-based networks form the basis for our methods. We can use these kinds of networks for diagnostic purposes. In our methods we assume another property which is that evidence is only entered at the leaves of the network. Just like we have seen in the *Wet Grass* example we can observe if the grass is wet or not, but we want to find out if it was caused by the rain or by a malfunctioning of the sprinkler. We also assume that, when all available evidence is entered in the Bayesian network, we will prune the network so that only the nodes that affect the posterior probabilities of interest are left in[3].

Now assume we have a pruned causal Bayesian network and we are interested in the most probable value of the node that represents the hypothesis. In order to make the problem not too complex we will be assuming that the networks we use only have a single hypothesis node. Then we could clarify the value of this node of interest by the use of the entered evidence. But in our method we want to go a step further. We want to find a set of nodes that are in-between the entered evidence and the node of interest. We will call this set of the nodes: *the intermediate nodes*. We want to explain that the values of (a part of) the entered evidence influences the most probable values of this set of intermediate nodes and that the most probable values of these intermediate nodes explain the most probable value of the hypothesis and therefore should give more insight in the reasoning of the Bayesian network.

So in the next chapter we will be discussing a method for finding this set of intermediate nodes that are in-between evidence and hypothesis and contain some sort of summarized information. Important to note is that neither the evidence nodes nor the hypothesis node should be included in this set of intermediate nodes. We are looking for a set that is completely in-between these nodes. For finding the set of intermediate nodes we will be focusing on the structure and possibly the probability distribution of the network in order to locate a set of nodes that contain or funnel a lot of important information of the Bayesian network.

In order to find this set of intermediate nodes we will take a look at the *Maximum-Flow-Minimum-Cut* theory and we will be using the associated Edmonds-Karp algorithm. We will be combining this algorithm with several weight-assignment functions, namely *One-for-All*, *Hellinger influence* and *Mutual information*. We will elaborate on all of these methods in the upcoming chapter.

The second major part of our method consists of generating clusters, where we take the set of intermediate nodes as input for this second method. In Chapter 6 we will explain how we can construct a cluster of nodes that should aid us in explaining the most probable values of the intermediate nodes. For each child of the set of intermediate nodes, we will create a cluster of nodes that contains its most influential nodes. In each

[3]Techniques for pruning a Bayesian network to a computationally equivalent subgraph can be found in the paper by M. Baker and T. Boult [1].

cluster there should be one or more evidence nodes which help us explain the most probable value of the associated intermediate node. Because each intermediate node has as much clusters as it has children, we will again look at the method of *Mutual information* in order to create an ordering among those clusters. With this we can find out which cluster seems most influential and so we find out which cluster should be prioritized in the eventual explanation of the Bayesian network.

After we have shown our method for creating an ordering among clusters with the calculation of *Mutual information*, we will end our method with a chapter about the final explanation as it is actually presented to the user. Here we will go into more detail on how to present the found results. Our explanation will be presented in a web page where the user can control the level of explanation. We try to give the user more comprehension by accompanying the found results with verbal expressions and graphical representations.

Now that we have seen a global overview of our method, we can take a look at Table 2 from the previous chapter. Our methods focusses on several properties of this table and here we illustrate the design decisions. At the end of this thesis will come back to these decisions to show if our explanation includes all of these properties.

- Content
  - Focus: *Reasoning*
    We find particular reasons for the most probable value of the node of interest.
  - Purpose: *Comprehension*
    We will gain more understanding about the reason the nodes have particular values.
  - Level: *Macro level*
    We focus on finding a set of nodes in order to explain the global reasoning of the network.
  - Causality: *Causal*
    As input for our method we use a causal Bayesian network.
- Communication
  - User-system interaction: *Natural language*
    We present an explanation of the network as a readable document.
  - Presentation: *Textual, graphical and multimedia*
    We present the explanation as a web page, with both a verbal and a graphical explanation.
  - Expression of probabilities: *Both numeric and linguistic*
    We make probabilities more understandable by accompanying them with verbal expressions.
- Adaption
  - Knowledge of domain: *Dynamic model*
    The user should have some background of the domain.
  - Knowledge of reasoning: *Dynamic model*
    The user should have some background of the reasoning.
  - Level of detail: *Dynamic*
    The user is able to look at several levels of explanation.

With these properties in mind we can go into more detail about our own methods. In the next chapter we will be focusing on finding the set of intermediate nodes of a Bayesian network. We explain the used methods and do several experiments on Bayesian networks in order to find out which particular methods works best.

5 Intermediate Nodes

As we have seen in Chapter 3, explaining a Bayesian network can been done in several ways and there are many different areas on which we can focus. In the next part of this thesis we will be focusing on how to explain the reasoning of a Bayesian network. Our goal is to find a way to generate an explanation based on the reasoning of the network for a user with no mathematical background. Because Bayesian networks can be developed for specific domains, like the medical field we mentioned earlier, the user may need to have a background in such a domain to understand the resulting explanation.

Because Bayesian networks can become very large and contain lots of data, we will be focusing on the macro level more than on the micro level. This means we will be looking at the global meaning and reasoning of the network and not so much at the posterior probabilities of each node. For generating the explanation we are first going to find nodes which seem to have a large influence in the result that the Bayesian network is going to give.

5.1 Defining intermediate nodes

Like we said in the introduction, we will be trying to find nodes that seem to be important for the explanation of the most probable value of the node of interest. In order to find these nodes we are looking at the structure of a Bayesian network, which is a graph $G(V, E)$. Recall that we assume that the Bayesian network is built in such a way that the edges between two nodes are causal relations. So when we have an edge from node A to node B, this means that node A is the cause of the effect B. So these edges contain information about the causality of the network and therefore can be used for finding points in the network that we will use in our explanation. These points should be nodes that summarize and/or funnel the reasoning captured by the Bayesian network.

When we are looking at the network, we want to find a non-empty set of nodes that are neither evidence nodes nor the hypothesis node. We want a set of nodes that are more or less "halfway". We refer to these as *intermediate nodes*. The global idea behind this is that we can then say that the observed evidence nodes are a consequence of the intermediate nodes, which in their place are a consequence of the hypothesis. This forms the basis of the first part of our method. A set of intermediate nodes I is defined as follows:

Definition 5.1.
Consider a Bayesian network with nodes $V(G)$, evidence nodes $E \subset V(G)$ and hypothesis node $H \in V(G) \setminus E$. Then the set of intermediate nodes is a set $I \subseteq V(G) \setminus (E \cup H)$ that summarizes the influence from the evidence E onto the hypothesis H.

In order to find such a set of intermediate nodes we need to have a way to create a non-empty set of nodes, which in some sense summarizes all information between the evidence nodes E and the node of interest H for a given Bayesian network. A well-known algorithm that finds a subset of edges from a graph by looking at weights on those edges is the *Edmonds-Karp* algorithm, which is based on the *Maximum-flow-minimum-cut* theorem. We will present a method to use this existing algorithm to aid us in finding a set of intermediate nodes.

5.2 Maximum Flow and Minimum Cut

In order to explain the idea behind the Edmonds-Karp algorithm [4], we first take a look at the slightly simpler version which is the Ford-Fulkerson algorithm. These algorithms use a flow network as their input. It is a directed graph $G = (V, E)$, where every edge (u, v) in E has a capacity c. In a flow network, there are two special nodes, namely the source node s and the sink node t. We can see an example of such a network in Figure 3. These networks are based on the idea that there is a flow of liquid running from one side of

the graph to the other side. The liquid is poured in at the source s and flows towards the sink t. An edge from node u to node v has a certain capacity $c(u, v)$; once its capacity is completely filled this edge (u, v) is saturated and cannot be used anymore. After some time it is impossible to add more flow to the network, because all paths from source s to sink t are blocked (i.e. it is not possible to find a path without at least one saturated edge in it). Note that this does not mean that every edge is completely saturated.

The major use of the algorithm lies in the theorem associated with these algorithms. It is called the *Maximum-Flow-Minimum-Cut theorem* and it says that once the graph G is maximally filled and no more flow $f(u, v)$ for an edge from node u to node v can be added to the network (saturated), we can find the minimum cut. A cut is defined as a set of edges for which it holds that if we remove these edges the graph becomes disconnected and the source s is not connected to the sink t anymore.

Definition 5.2.
The minimum cut is the minimal set of edges for which the combined residue is minimum over all other cuts in the network.

In other words the minimum cut is the smallest possible set of edges that makes the graph disconnected. Residue is, like the name says, the leftover capacity after we subtracted the applied flow. For example when an edge has a capacity of 6 and we applied 4 units of flow to this edge, the residue capacity will be 2. Because we apply as much flow as we can, the minimum cut always has a combined residue of 0 and because the set is minimal in the amount of edges used, the minimum cut is also minimal. Note that this does not mean there can only been one minimal cut. So important to remember is the fact that the graph will be disconnected if we would remove these edges from the graph.

The minimum cut can be seen as a kind of funnel, by which we mean that this part of the network contains the least number of edges over which the reasoning of the network goes. In some sense we can state that the information encoded in the network is condensed by the minimum cut and that the nodes that are connected to these edges summarize the reasoning of the rest of the Bayesian network. If we would try to make the graph disconnected with another set of edges it would contain more edges (or be an equally sized set). By using the Maximum-Flow-Minimum-Cut theorem we ensure that the smallest set of edges is found and therefore we find a part of the Bayesian network which encodes and summarizes the most reasoning of this network.

Now we will go into more detail about how the algorithms formally work. So both the Ford-Fulkerson algorithm as the Edmonds-Karp algorithm find the maximum possible flow for some given network. In Figure 3 we see an example of a flow network. The weights given to the edges represent the capacity that each edge has. Ford-Fulkerson is based on the idea of *residual networks* and *augmenting paths*.

Definition 5.3.
A residual network is the resulting network that arises after flow is applied to the network and the residual capacities are re-calculated.

Definition 5.4.
An augmenting path is a path with positive capacity from source s to sink t.

The algorithm tries to find a path p from source s to sink t regardless of the direction of the edge. This does not mean the direction is completely disregarded, it has an important use in the calculation of new residual capacities. Later on we will explain exactly how this works, but for finding a path p we ignore the direction. An edge from node u to node v can only be included inside path p when the residue capacity $c(u, v) > 0$. When the path finding algorithm finds a path p it will calculate the maximum amount of flow that can be sent across those edges. It calculates the maximum flow by examining each individual edge and choosing the edge with the lowest capacity:

$$\arg \min_{(u,v) \in E} c(u, v) \tag{6}$$

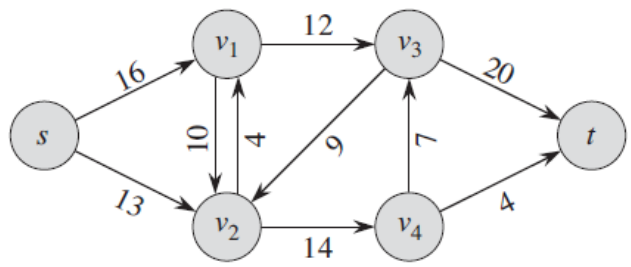

Figure 3: Example of a flow network [4]

So the capacity $c(u, v)$ of edge (u, v) is the constraint for the entire path p, because it can't allow anymore flow and therefore the path p cannot allow more flow. If we consider the path s, v_2, v_4, t in Figure 3, the edge (v_4, t) has the lowest capacity of all the edges in that path and therefore the flow that can travel this path is equal to 4. Now the algorithm will update the residue capacities for each edge e in path p. This will create a residual graph G_f, where the capacity of some edges is updated due to the flow in the network. So the network is now partially filled, but as long as there is a path from source s to sink t, it will continue to apply more flow to the network, until it is completely saturated.

We mentioned before that we can find a path p that includes edges that point in the opposite direction. When this is the case the algorithm doesn't calculate the maximum flow in the same way. Normally you would subtract the flow from the capacity and this leaves you with the residual capacity. When an arrow directs in the opposite way of the path p in which it is found, the flow is not subtracted but added to the edge. There is only one restriction and that is that the residual capacity can never be larger than the original capacity. The reason why this is done is to ensure the network will be completely saturated, so it is not possible to add anymore units of flow to the network. So formally we can only include an edge (v, u) if there is already flow going over edge (u, v). The flow that can travel from node v to node u is equal to the flow $f(u, v)$ that is already entered in the residual network.

Let us go back to our example in Figure 3 and assume that we find the path s, v_2, v_3, t. We have an edge (v_2, v_3) which is a backward edge. This edge is only allowed in the path if there was an earlier path in which (v_3, v_2) was included and therefore a certain amount of flow was applied to this edge, for instance a flow of 4. Then the maximum allowed flow over the path (v_2, v_3) is now equal to 4. The calculation of maximum flow for path p is done by taking the minimum capacity $c_f(u, v)$ where

$$c_f(u, v) = \begin{cases} c(u, v) - f(u, v) & \text{if } (u, v) \in E \\ f(u, v) & \text{if } (v, u) \in E \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

In Algorithm 1 we see the pseudo code for the Ford-Fulkerson algorithm. In the first three lines we make sure that the flow in each edge (u, v) is set to zero. Then we start finding path p from source node s to sink node t where we make sure there is available capacity along this path. As long as these paths can be found the algorithm will re-calculate the capacities and flow and deliver a new residual graph G_f. When it is no longer possible to create a new graph G_f this iteration stops and with the current graph we can find the minimum cut. If we would again run flows from the source to the sink, these flows would not be able to pass certain nodes, because all other edges connected to this node are saturated. The set of all these edges is the minimum cut.

This algorithm can easily be extended to the Edmonds-Karp algorithm which is almost the same. The only difference is that the paths in Ford-Fulkerson can be found by any path finding algorithm and the paths

in Edmonds-Karp are found using Breath First Search, which guarantees that it always finds the shortest available path (or at least one of them) which leads to an optimal solution.

Algorithm 1 Ford-Fulkerson (G, s, t)

```
for each edge (u, v) ∈ E(G) do
    (u, v).flow = 0
end for

while there exists a augmenting path p from s to t in the residual network G and
        the minimal capacity for path p > 0 do
    calculate maximum flow f_max for that path p
    for each edge (u, v) ∈ Path p do
        if (u, v) ∈ E then
            (u, v).flow = (u, v).flow + f_max
        else
            (v, u).flow = (v, u).flow − f_max
        end if
    end for
end while
```

5.3 Applying Edmonds-Karp to Bayesian Networks

Now that we have some understanding of flow algorithms, we would like to apply these to the graphs of Bayesian networks. In order to do so, we need to resolve two issues. The first one concerns the general structure of the Bayesian network, which does not necessarily match that of a flow network and the second one concerns the weight (capacity) of the edges. In this section we will be focusing on how to make sure a Bayesian network satisfies the presence of a source and sink node and propose a solution to this problem. Later on we will be looking at how we can define capacities for the edges in the network.

The structure of a Bayesian network does not completely match the structure of a flow network. An important aspect of a flow network is the presence of a source and sink node. The source s has to be a root node with only outgoing edges and it has to be the only one in the graph. The sink t is a node with only incoming edges and again has to be the only one in the graph. A source and sink node are not standard in a normal Bayesian network, but we can easily construct a network where these conditions hold. Without a source and a sink node we do not have a specific starting point and end point. So in order to make sure we can actually apply Edmonds-Karp to our network we make sure every Bayesian network comes with a source and a sink node.

In order to solve this problem we are going to introduce two dummy nodes in the graph of a Bayesian network. These dummy nodes will take the place of the source and the sink. We will connect the first dummy node to all nodes that do not have any parents in the original graph. All the edges that we create will be outgoing edges from the dummy node. By doing this we ensure that there is exactly one node with no incoming edges and only outgoing edges, this will be the dummy source node s_d. We can create the dummy sink node t_d in a similar fashion, by connecting all nodes without any children to this node. Of course this time all edges should be incoming edges for the dummy node. Transforming the Bayesian network's digraph in this way makes it a proper flow network in which each edge (u, v) can be found by the path finding algorithm. To illustrate this process we can look at Figure 4, were we see such a transformation. On the

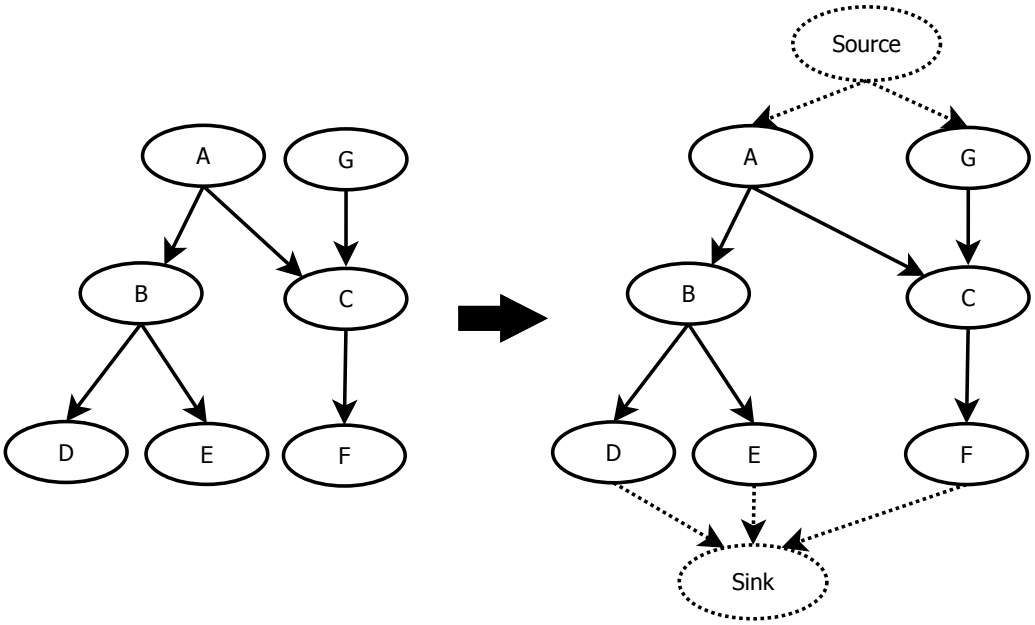

Figure 4: Adding a source and sink.

left hand side we have the original Bayesian network graph and on the right hand side we have our new flow network which is fit for Edmonds-Karp.

Important to understand is that we do not want any of the new dummy edges, the dotted edges in Figure 4, in our final explanation. These edges were not present in the creation of the network, so they do not encode any domain knowledge and are therefore irrelevant for a possible explanation. There is an easy way to make sure none of these edges will be in our results. We will be setting the weight of the new edges to infinity (or in code to the maximum integer value available). The weights we are going to use in the next section will never be close to this value, and therefore the new edges will not be in the answer.

In the next section we will look at three methods that will determine the weights for the edges in our transformed Bayesian network graphs. One method will be completely based on the structure of the network and does not take into account any probability distributions. We also present two different methods that do take the probability distributions of the Bayesian network into account.

The weight of the edges is an important aspect in finding the minimum cut. The weight of the edges says something about how likely it is that it will be a part of the minimum cut. If the weight of one edge is lower than all other edges we can argue that it is more likely for this edge to be in the minimum cut. Less weight means that there is less capacity within that edge and that it is saturated earlier than the other edges. Of course this is entirely dependent on the structure of the model. So if we vary the choice of the weights for the edges we may end up with a different minimum cut and therefore a different basis for our explanation.

5.4 Minimum cuts vs Intermediate nodes

We need to make important decisions about choosing the set of nodes connected to the minimum cut. Remember that the minimum cut is the minimal number of edges for which it holds that when these edges are removed, the graph becomes disconnected entirely, i.e. there is no more path from the source node to the sink node. So here we are talking about edges and not nodes, while we actually want to find a set of nodes. We are going to consider nodes that are connected to the edges of the minimum cut. Note that it is possible to take two different sets of nodes, namely all the nodes on one side of the minimum cut or all the nodes on the other side of the minimum cut. In this thesis we choose to take the set of nodes that was furthest away from the evidence nodes, i.e. the set of intermediate nodes closest to the dummy source node. The reason for this was that we want the set of intermediate nodes to contain nodes that would represent an important indication for the value of the node of interest and therefore these nodes should contain a lot of cause-effect relations contained in the network. We argue that, when an intermediate node is further away from the evidence nodes, it contains more cause-effect relations and therefore should include more reasons for the value of the intermediate node.

Another important decision we made, was that we only looked at the first cut we could find. It is possible that a flow network contains more than one minimum cut and in the next section we will see an example with One-for-All where it occurs that there is more than one minimum cut in the Bayesian network. The question then is which minimum cut do we use for determining the set of intermediate nodes. For this problem we actually made a similar decision as we did above. We always selected the minimum cut that was furthest away from the evidence nodes, because then the set of intermediate nodes contains the most cause-effect relations. This is the first cut we found from the source node, the cut furthest from the evidence nodes (leaves). In our Discussion (Chapter 9) we will come back to this point and discuss the possibilities for other sets of intermediate nodes and minimum cuts.

It is possible that the node of interest is in the minimum cut. This is undesirable behavior, because we want to explain the most probable value of this node of interest. Two situations can occur, where one is that the minimum cut consists of more nodes than the node of interest. In this case we just remove the node of interest from the set of intermediate nodes. The other situation is when the node of interest is the only node in the set of intermediate nodes and if we would remove this node the set of intermediate nodes becomes empty. Then we take the nodes on the other side of the minimum cut as our set of intermediate nodes. In this case it is possible that there are evidence nodes included in the set of intermediate nodes, but we simply remove these nodes from the set. If this set also become empty we were unable to find a set of intermediate nodes, but this means the node of interest was directly connected to the evidence nodes. It is a logical consequence that we can not find a node in between the node of interest and the evidence nodes.

5.5 Weight-assignment functions

In the following we will discuss several ways of choosing the weights of the edges. In order to determine the weight of each edge in a Bayesian network we have experimented with three different methods. Each of these methods is based on its own ideas, where one method is purely based on the structure of the graph of the Bayesian network and the other two different methods also take the probability distribution of the network into account.

5.5.1 One-for-All

In our first method we will be setting the weight w of each edge (u, v) in the Bayesian network to one. This means that, when we run the Edmonds-Karp algorithm, each edge (u, v) can be used at most once, after which the capacity is reduced to zero. The idea of giving all edges equal weight is that we only take into account the structure of the network. Important to understand is that, if we would choose another value

$P(B\|A)$	a	$\bar{a}$
b	0.5	0.1
$\bar{b}$	0.5	0.9

Table 3: Probability table for node B given node A

than one for the weights, this method would still give back the same results as long as all the weights are equal to each other. This means that every edge is equally important.

This method is computationally easy to calculate, because the hardest part is applying the Edmonds-Karp algorithm, which has a running time of $O(|V|\cdot|E|^2)$. This method does not require any hard calculations for determining the weights of the edges, because in this case we do not take the probability distributions into account. However in the following two methods we do use the probability distributions.

To illustrate how this method works we can take a look at the Bayesian network graph given in Figure 5. Here we have three nodes A, B and C and these are connected by two edges (A, B) and (B, C). By assigning weights to these edges, we will be able to determine the minimum cut. All edges in the network shown in Figure 5 will be given a weight of one. Transforming this Bayesian network into a flow network would create a dummy source node s as a parent of node A and a dummy sink node t as a child of node C. If we then apply the Edmonds-Karp algorithm to this network we will have two possible cuts, because both edges have equal weights and therefore will be completely saturated. We will find a minimum cut between A and B and between B and C, but because of our assumption in Section 5.4 about taking the first cut, we take the minimum cut closest to the source node. In this case the edge to be selected would be between node A and node B and therefore the set of intermediate nodes would consist of node A. In this case the edge to be selected would be (A, B) and therefore the set of intermediate nodes would consist of node A, unless node A was the node of interest. In the latter case, we would take the node on the other side of the minimum cut, which is node B.

5.5.2 Hellinger influence

For our second method we are going to include the probability distributions of the Bayesian network, which is opposite to what we have seen with our One-for-All measurement. This method is going to be based on the Hellinger distance [21] *between* nodes as it is used in the tool *GeNIe* [15], and introduced for this purpose by J.R. Koiter [19]. Koiter translates Hellinger distances of probability distributions in general into influence for edges in a Bayesian network. In this section we will take a closer look at his definition and implementation and we will show how we can apply it to our research in order to determine weights on edges. In this section we will describe the influence of an edge based on the Hellinger distance between two nodes, we will refer to this influence as the *Hellinger influence*.

This Hellinger influence is a measure that captures how much effect one node can have on another node connected by a direct edge: the higher the influence value, the stronger the change in posterior probability can be. Koiter describes several methods [19] of determining the distance between probability distributions,

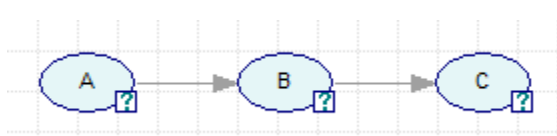

Figure 5: Graph of a simple Bayesian network

which he then uses as an influence measure. In order to explain how we can calculate the Hellinger influence, we first look at a slightly simpler version, which is the Euclidean influence (an influence measure based on the Euclidean distance).

In order to calculate the influence of an edge based on Euclidean influence between two nodes in a Bayesian network, we need to take a look at the difference between the probability distributions of these two nodes. Assume that we have two binary-valued nodes A and B with an edge from A to B. We can identify the following two probability distributions P_a and $P_{\bar{a}}$ from the probability tables from the given Bayesian network, like we can see in Table 3:

$$P_a = \{P(b|a), P(\bar{b}|a)\} \tag{8}$$

$$P_{\bar{a}} = \{P(b|\bar{a}), P(\bar{b}|\bar{a})\} \tag{9}$$

The Euclidean influence E_B^A from node A to node B for the associated probability distributions P_a and $P_{\bar{a}}$ can be calculated by iterating over both distributions in the following formula:

$$E_B^A(P_a, P_{\bar{a}}) = \sqrt{\sum_{b_i \in \{b,\bar{b}\}} (P_a(b_i|a) - P_{\bar{a}}(b_i|\bar{a}))^2}. \tag{10}$$

An important property of this function is that it is symmetric and therefore it holds that the influence $E_B^A(P_a, P_{\bar{a}})$ is equal to the influence $E_A^B(P_{\bar{a}}, P_a)$. With probability distributions this function will always result in a value with a minimum of 0 and a maximum of $\sqrt{2}$, and therefore we can normalize the answer:

$$E_{norm}(P_a, P_{\bar{a}}) = \frac{E_B^A(P_a, P_{\bar{a}})}{\sqrt{2}}. \tag{11}$$

Like we mentioned before we will use the Hellinger influence in our method and not the Euclidean influence. We choose to use the Hellinger influence, because this method tends to be more sensitive than the Euclidean influence, when it approaches zero or one. Moreover this method is not computationally harder than its Euclidean version. The Hellinger influence measure H_B^A, given two probability distributions P_a and $P_{\bar{a}}$, is defined as follows:

$$H_B^A(P_a, P_{\bar{a}}) = \sqrt{\sum_{b_i \in \{b,\bar{b}\}} (\sqrt{P_a(b_i|a)} - \sqrt{P_{\bar{a}}(b_i|\bar{a})})^2}. \tag{12}$$

Just as with the Euclidean influence the result of the Hellinger influence varies from 0 to $\sqrt{2}$, when using probability distributions, and again we can normalize the result:

$$H_{norm}(P_a, P_{\bar{a}}) = \frac{H_B^A(P_a, P_{\bar{a}})}{\sqrt{2}}. \tag{13}$$

We will use this Hellinger influence as an influence measure to determine the weight of an edge. In order to do this we need to address some issues regarding the way we implement this Hellinger influence and the way we translate this to the influence of an edge.

Implementation issues

In order to assign weights to the edges we have to make a few small adjustments. The first adjustment is specific to our approach of the Edmonds-Karp algorithm. The calculated influence is a number between zero and one, but in the introduction we mentioned that one unit of flow was already equal to one and we would only use positive natural numbers. To fix this we will multiply the results of the Hellinger influence measure by one-hundred and then round the answer to a natural number. Note again that if we implemented Edmonds-Karp differently we could have used the original value of the Hellinger influence.

Another adjustment we have to make is based on the meaning of the influence. This is a very essential adjustment, because it involves the meaning of the influence value. When the influence has a high value this edge is important and very sensitive to change. When we take a look at the meaning of the weight in the Edmonds-Karp algorithm we can state that it is exactly the other way around. In our minimum cut we would like to find edges which have importance in the Bayesian network and a strong meaning in order to explain the reasoning. An edge with a lower weight is more likely to run out of capacity than an edge with a higher weight. Therefore we take one minus the influence value in order to maintain the importance of the node. The combined changes lead to the following weight assignment function $f_w(v)$, where v is the influence value given by the normalized Hellinger influence:

$$f_w(v) = \lfloor 100(1-v) \rceil;\ f_w(v) \in \mathbb{N} \tag{14}$$

With the use of the Hellinger influence measure and the adjustments to the influence values we have weights we can assign to the appropriate edges. At this point we can simply run the Edmonds-Karp algorithm in order to find the minimum cut, which now has been influenced by the Hellinger influence measure.

Example
In Table 4 we have probability tables that belong to the network shown in Figure 5. In the middle probability table (for Node B) we can see that no matter what the state is of a node, the outcome will always have a probability of 0.5. In the most right probability table (for Node C) we have a completely different distribution, where the values are closer to one or zero. The leftmost probability table gives the probabilities for the states of node A. With this example we will illustrate the effect of the influence of the edges between the nodes in our example.

$P(A)$	
a	0.5
$\bar{a}$	0.5

$P(B\|A)$	a	$\bar{a}$
b	0.5	0.5
$\bar{b}$	0.5	0.5

$P(C\|B)$	b	$\bar{b}$
c	0.9	0.2
$\bar{c}$	0.1	0.8

Table 4: Probability tables for Node A (left), for Node B (middle) and for Node C (right)

Earlier we have described how we can calculate the Hellinger influence $H(P_a, P_{\bar{a}})$ for two probability distributions P_a and $P_{\bar{a}}$. We will now illustrate how the Hellinger influence between two distributions is translated into an influence between two nodes. Consider the probability table of Node B and let:

$$P_a = \{P(b|a), P(\bar{b}|a)\} \tag{15}$$
$$P_{\bar{a}} = \{P(b|\bar{a}), P(\bar{b}|\bar{a})\} \tag{16}$$

Using these distributions in the Hellinger influence equation results in:

$$H_B^A(P_a, P_{\bar{a}}) = \sqrt{\sum_{b_i \in \{b,\bar{b}\}} (\sqrt{P_a(b_i|a)} - \sqrt{P_{\bar{a}}(b_i|\bar{a})})^2} = 0 \tag{17}$$

The result of this equation gives a value of zero to the edge (A, B), which is of course a logical consequence when all values in the probability table are equal to each other. When we calculate the Hellinger influence with the other probability table of Node C, we use the following distributions:

$$P_b = \{P(c|b), P(\bar{c}|b)\} \tag{18}$$
$$P_{\bar{b}} = \{P(c|\bar{b}), P(\bar{c}|\bar{b})\} \tag{19}$$

$$H_C^B(P_b, P_{\bar{b}}) = \sqrt{\sum_{c_i \in \{c,\bar{c}\}} (\sqrt{P_b(c_i|b)} - \sqrt{P_{\bar{b}}(c_i|\bar{b})})^2} = \sqrt{((\sqrt{0.9} - \sqrt{0.2}) + (\sqrt{0.1} - \sqrt{0.8}))^2} = 0.765 \quad (20)$$

The largest Hellinger influence can be found between the nodes B and C, namely $H_C^B = 0.765$, which means that this edge encapsulates the most influence in this small network. We now translate the found influences into weights using Equation 14. In our example this results in a minimum cut between the nodes B and C, which is different from our result in the One-for-All method. With this example we illustrate that this method can give different results based on the values in the probability tables. Also it is easy to see that if we would assign the probability table of $P(B|A)$ to $P(C|B)$ and visa versa, we would end up with a minimum cut between nodes A and B.

In the above description we have focussed on the influence of one node on another in case there are no other parents. We need to generalize our computations in case there are multiple parents. We follow the suggestion in Koiter [19] to take the average of all differences. We implement the *Average Hellinger influence* as follows. Assume we want to calculate the Hellinger influence H_C^A for an edge (A, C). Node C has n states, node A has m states and there are k parent configurations b_i, $i = i, \ldots, k$.

$$H_C^A(P_{1B}, \ldots, P_{mB}) = \frac{1}{\frac{1}{2}m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} H_C^A(P_{iB}, P_{jB}) \quad (21)$$

$$= \frac{1}{\frac{1}{2}m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \frac{1}{k} \sum_{l=1}^{k} H_C^A(P_{il}, P_{jl}) \quad (22)$$

$$= \frac{1}{\frac{1}{2}km(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \sum_{l=1}^{k} H_C^A(P_{il}, P_{jl}) \quad (23)$$

Now that we have considered a small example to illustrate how Hellinger influence works and showed how we can handle more parents and more values we can go on to our last method, which is Mutual information.

5.5.3 Mutual Information

For our last method we will be using *Mutual Information* [5] in order to determine the weights for the edges. We again do this to give more guidance to the Edmonds-Karp algorithm, because we will not purely look at the structure of the Bayesian network like the One-for-All method, but also take the probability distributions into account just like we did with the Hellinger influence. We can compare the Hellinger influence with mutual information, because they both use the probability distributions of the Bayesian network. The difference lies in how we use the probabilities. With mutual information we can measure the amount of information one variable contains about another variable. In order to explain mutual information we will start with the notion of Entropy on which it is based.

In physics, entropy gets associated with the amount of order/disorder/chaos in a system. Entropy is a measurement which likewise can say something about the uncertainty among variables: the higher the value of the Entropy the more uncertainty there is. For our method we see the value of Entropy as information measure, which indicates how much information one node has about another. The Entropy $H(X)$ [5] for a discrete random variable X is defined as:

$$H(X) = -\sum_{x \in X} P(x) \log P(x). \quad (24)$$

Entropy $H(X)$ makes use of the base 2 logarithm. The value of Entropy itself has the useful property that it is always non-negative, which is a property we want for determining our weights. Mutual information is a measure of the amount of information that one random variable contains about another random variable. We will define the mutual information over the random variables X and Y as follows:

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} P(x,y) \log \frac{P(x,y)}{P(x)P(y)} \tag{25}$$

Mutual information $I(X;Y)$ is related to entropy because it describes the reduction in the uncertainty of X due to the knowledge of Y. In other words we can say:

$$I(X;Y) = H(X) - H(X|Y). \tag{26}$$

In our third method we will use the mutual information between two connected nodes as the weight of the edge that connects them. The idea is that when a node N has more than one incoming or more than one outgoing edge we can compare the strengths of those edges. We compare the amount of information the other nodes connected to node N have about N.

Implementation issues

Just like we have seen in our previous method we need to make some small adjustments in order to make them fit with the Edmonds-Karp algorithm. First of all we will multiply the mutual information by a large number and round it to a natural number in order to make it compatible with our flow assignment. In a pilot experiment we used one-hundred as the multiplier just like the Hellinger influence does. Only this was not sufficient, because although the mutual information was always non-negative, the values would become so small that most edges would still receive a weight between zero and one. So for this method we used a multiplier of thousand. If we would translate this to a formula, where v is the mutual information between of two nodes, we would get:

$$f_w(v) = \lfloor 1000(1-v) \rceil; \; f_w(v) \in \mathbb{N} \tag{27}$$

With this method we are able to find a weight which is based on the information that nodes have about each other, in other words the mutual information. If we apply this to the whole network we are able to determine a minimum cut and therefore the set of intermediate nodes.

Example

To illustrate mutual information we will again use the probability tables shown in Table 4. First we will rewrite the function to make sure we can get all probabilities directly from this table.

$$I(A;B) = \sum_{A} \sum_{B} P(A,B) \log_2 \frac{P(A,B)}{P(A)P(B)} \tag{28}$$

$$= P(a,b) \log_2 \frac{P(a,b)}{P(a)P(b)} + P(\bar{a},b) \log_2 \frac{P(\bar{a},b)}{P(\bar{a})P(b)} + P(a,\bar{b}) \log_2 \frac{P(a,\bar{b})}{P(a)P(\bar{b})} + P(\bar{a},\bar{b}) \log_2 \frac{P(\bar{a},\bar{b})}{P(\bar{a})P(\bar{b})} \tag{29}$$

$$= P(b|a)P(a)\log_2 \frac{P(b|a)P(a)}{[P(b|a)P(a) + P(b|\bar{a})P(\bar{a})]P(a)} \tag{30}$$

$$+ P(b|\bar{a})P(\bar{a})\log_2 \frac{P(b|\bar{a})P(\bar{a})}{[P(b|a)P(a) + P(b|\bar{a})P(\bar{a})]P(\bar{a})} \tag{31}$$

$$+ P(\bar{b}|a)P(a)\log_2 \frac{P(\bar{b}|a)P(a)}{[P(\bar{b}|a)P(a) + P(\bar{b}|\bar{a})P(\bar{a})]P(a)} \tag{32}$$

$$+ P(\bar{b}|\bar{a})P(\bar{a})\log_2 \frac{P(\bar{b}|\bar{a})P(\bar{a})}{[P(\bar{b}|a)P(a) + P(\bar{b}|\bar{a})P(\bar{a})]P(\bar{a})} \tag{33}$$

If we enter all the probabilities given in Table 4, we will get that the mutual information between node A and node B is equal to zero, which is caused by $\log(1) = 0$. This result is the same as the result for the method of the Hellinger influence. This is of course the effect of every probability being the same (0.5). When we apply the equation to find out what the mutual information is between nodes B and C we end up with the following result:

$$= P(c|b)P(b)\log_2 \frac{P(c|b)}{P(c|b)P(b) + P(c|\bar{b})P(\bar{b})} \tag{34}$$

$$+ P(c|\bar{b})P(\bar{b})\log_2 \frac{P(c|\bar{b})}{P(c|b)P(b) + P(c|\bar{b})P(\bar{b})} \tag{35}$$

$$+ P(\bar{c}|b)P(b)\log_2 \frac{P(\bar{c}|b)}{P(\bar{c}|b)P(b) + P(\bar{c}|\bar{b})P(\bar{b})} \tag{36}$$

$$+ P(\bar{c}|\bar{b})P(\bar{b})\log_2 \frac{P(\bar{c}|\bar{b})}{P(\bar{c}|b)P(b) + P(\bar{c}|\bar{b})P(\bar{b})} \tag{37}$$

$$= 0.45\log_2 \frac{0.9}{0.55} + 0.10\log_2 \frac{0.2}{0.55} + 0.05\log_2 \frac{0.1}{0.45} + 0.40\log_2 \frac{0.8}{0.45} \tag{38}$$

$$\approx 0.397 \tag{39}$$

The result of the mutual information for node C, based on the information of node B, has a value of 0.397. So for the assignment of the edge we subtract the found value from one. So edge (A, B) gets a value of a thousand and edge (B, C) gets the value 603. We can now see that this latter edge (B, C) is filled the quickest and therefore the minimum cut would be here. The set of intermediate nodes would then consist of node B, due to our choices in Section 5.4

In these small examples we explained how the weights for each method are calculated and what we have to do in order to find them. In the next section we will be focusing on the experiments we have done on randomly generated Bayesian networks to see how our methods perform.

5.6 Experiments for finding intermediate nodes

In the previous section we have elaborated on finding intermediate nodes with the use of the *Maximum-Flow-Minimum-Cut* theorem. For the assignment of weights to edges we have suggested three different methods that can be used with the Edmonds-Karp algorithm. In the next part we will investigate which of these methods is the most useable for our final explanation. We will start with experiments on randomly generated Bayesian networks, where we first explain how these networks are created and then show the results of the actual experiments. After that we will be looking at a Bayesian network that has been created for a real life purpose to see how our methods perform on such networks.

5.6.1 Research questions

We are going to run experiments in order to find out which of the three weight-assignment functions works the best. This can be done in several ways, but the goal is to find out if it matters which of the three methods we use. So the main question to be answered by these experiments was:

- Does it matter which of the three different weight-assignment functions we use for determining the set of intermediate nodes?

In order to investigate this we further specified the question. First of all we wanted to find out if the results of the given methods differ and if so, how much? We wanted to know in how many of the cases the results would be the same, i.e. do the found sets of intermediate nodes contain exactly the same nodes? We also wanted to know if there was overlap amongst the sets of intermediate nodes. Of course there is complete overlap if the sets of intermediate nodes are the same, but we also wanted to know if, when the sets were not exactly the same, there were nodes that were in both of the sets. We also wanted to investigate the size of the found sets of intermediate nodes. We wanted to find out if the methods would give different sized sets as a result. We were interested in the size, because we wanted the set of intermediate nodes to be a small set of nodes and not a set that contains half of all nodes in the graph. Last of all we wanted to look at the duration of the experiments, where we expect One-for-All to be significantly quicker than the other two. This due to the fact that it does not take the probability distribution of the Bayesian network into account like the other two methods. Overall we are also interested to see if the results answer to these questions, when the density of the graph changes. So the questions in which we were interested are:

- Given the three weight-assignment functions, in how many of the cases do the methods give back exactly the same set of intermediate nodes?
- Given the three weight-assignment functions, in how many of the cases do the methods return sets of intermediate nodes that are partially overlapping?
- What is the average size of the set of intermediate nodes for each of the given three weight-assignment functions?
- How much effect does the computation of the weight-assignment have on the duration of each of the these functions given the number of nodes and edges?
- Given the density of the graph, can we make some distinction between the obtained results?
- Can the set of intermediate nodes be used in an explanation to say something about the value of the node of interest?

In the following we will address these questions. We wanted to do experiments on many different networks, so we decided to randomly create Bayesian networks, because the amount of real-life example Bayesian networks was limited. In this thesis we will use one specific real-life Bayesian network, the Oesophagus Cancer network, but all other networks will be created randomly. In the following part we will describe the creation of these Bayesian networks.

5.6.2 Methods

We wanted to do extensive experimenting on a large amount of Bayesian networks and therefore we decided to create them randomly. In this section we will first describe our method for creating these random Bayesian networks. Afterwards we will describe the drawbacks of this method. We did not only wanted to experiment on randomly generated networks, but we wanted to compare the outcome of these experiments with the outcome of a real-life Bayesian network. In the second part of this section we will describe such a real-life

Bayesian network, namely the Oesophagus Cancer network.

Algorithm 2 RandomBayesianNetwork (int n, int m)

```
int counter = 0
Set connected = Empty set
Set unconnected = Set of n nodes

Node a = Get random node from unconnected
Remove node a from unconnected
Add node a to connected

while unconnected is not empty do
    Node a = Get random node from unconnected
    Remove a from unconnected

    Node b = Get random node from connected
    Add a to connected
    Assign Edge(a, b)

    counter++
end while

while counter < m do
    Node a = Get random node from connected
    Node b = Get other random node from connected

    if Edge(a, b) does not create a cycle then
        Assign Edge(a, b)
        counter++
    end if
end while

for each Node c in connected do
    Create probability table for node c with random probabilities.
    The size of the table is based on the number of parents of node c.
end for
```

Randomly generated Bayesian networks

In order to investigate how our three methods perform, we wanted to do some testing on a large number of Bayesian networks. The best way to do this seemed to be by generating lots of random Bayesian networks. Important to note is that generating random networks might lead to some complications. So first we will focus on how we created such a random network and after that we are going to address some issues with

this kind of generation.

In Algorithm 2 we can see the pseudo code for the generation of random Bayesian networks as we have defined it. The algorithm takes two values as input, namely the number of the nodes n and the number of edges m. We assume the user input such that we can always create a viable connected acyclic digraph, which means that the number of edges m should be at least $n+1$ and at most $m \leq \frac{n(n-1)}{2}$.

In the first three lines we define two sets; one set will contain all nodes that have been connected and one set will have all unconnected nodes. Also we use a counter to check how many edges have been assigned. To initialize the generation of the network we take one random node from the set of unconnected nodes and place this node in the set of connected nodes. We need to do this because now we can always pick at least one node from the set of connected nodes.

On line 9 we start a while loop in which we iterate over every node in the set of unconnected nodes. In each iteration we pick one random node A from the set of unconnected nodes and one random node B from the set of connected nodes. We connect these nodes from node A to node B, place node A into the set of connected nodes and remove it from the set of unconnected nodes. Note that this method will never create a cycle, so we do not need to check for this. At the end of each while loop iteration (line 17) we increase the counter to keep track of the number of added edges.

At the end of the first while loop we have a completely connected directed graph without any cycles. At this point the set of unconnected nodes is empty and the set of connected nodes contains n nodes. After this point we check if all m edges were added. As long as this is not the case we keep adding edges. We do this starting by taking two random nodes A and B from the set of connected nodes. In order to preserve the DAG property, here we have to check (line 24) if creating an edge from A to B does not create a cycle. When this action does not create any cycles we assign this edge and increase the counter by one.

After we assigned m edges to this network we have one thing left and that is creating probability tables, which is done starting at line 30. The size of the table can be found by iterating over the set of connected nodes and for each node we check the number of parents. With this information we can create a probability table where the values are randomly generated. Of course we need to respect the property that all the probabilities for states of one variable add up to one. However, because we work with binary states, for each state with probability p we can assign $1-p$ to the opposite state.

After this we have a valid Bayesian network which can be used in the experiments. But as we mentioned before we want to address some aspects of randomly generating Bayesian networks that are important to note, in particular two aspects. The first one is that the networks we are constructing can be unbalanced, which means that the first selected nodes have a larger probability of having more connections, than nodes that are selected in a later stage. This is the effect of nodes getting randomly selected from the set of connected nodes. The first node that has been added to this set of connected nodes can in theory be selected every time during the execution of the algorithm. As a result, this node will probably be picked more often and will have more connections to other nodes than the nodes selected later on.

The second problem lies in the creation of the probability tables. All the values in the probability tables are randomly chosen from a uniform distribution between $[0, 1]$. A consequence of this is that the expected value is then a half and this is the case for each of the randomly generated Bayesian networks. We can question ourselves if this is fair for Bayesian network, because it is not very likely to see extreme cases. With these two aspects in mind we can use Algorithm 2 for our experiments.

The Oesophageal cancer network

There are two major issues with these randomly generated Bayesian networks and that is they do not have an underlying interpretation and their structure is not similar to that of a Bayesian network. Bayesian networks are designed for a specific situation and environment, and these are impossible to generate randomly. So in order to see if our method performs on an actual usable Bayesian network that contains an underlying meaning, we are going to look at a network which is called the Oesophagus network [34].

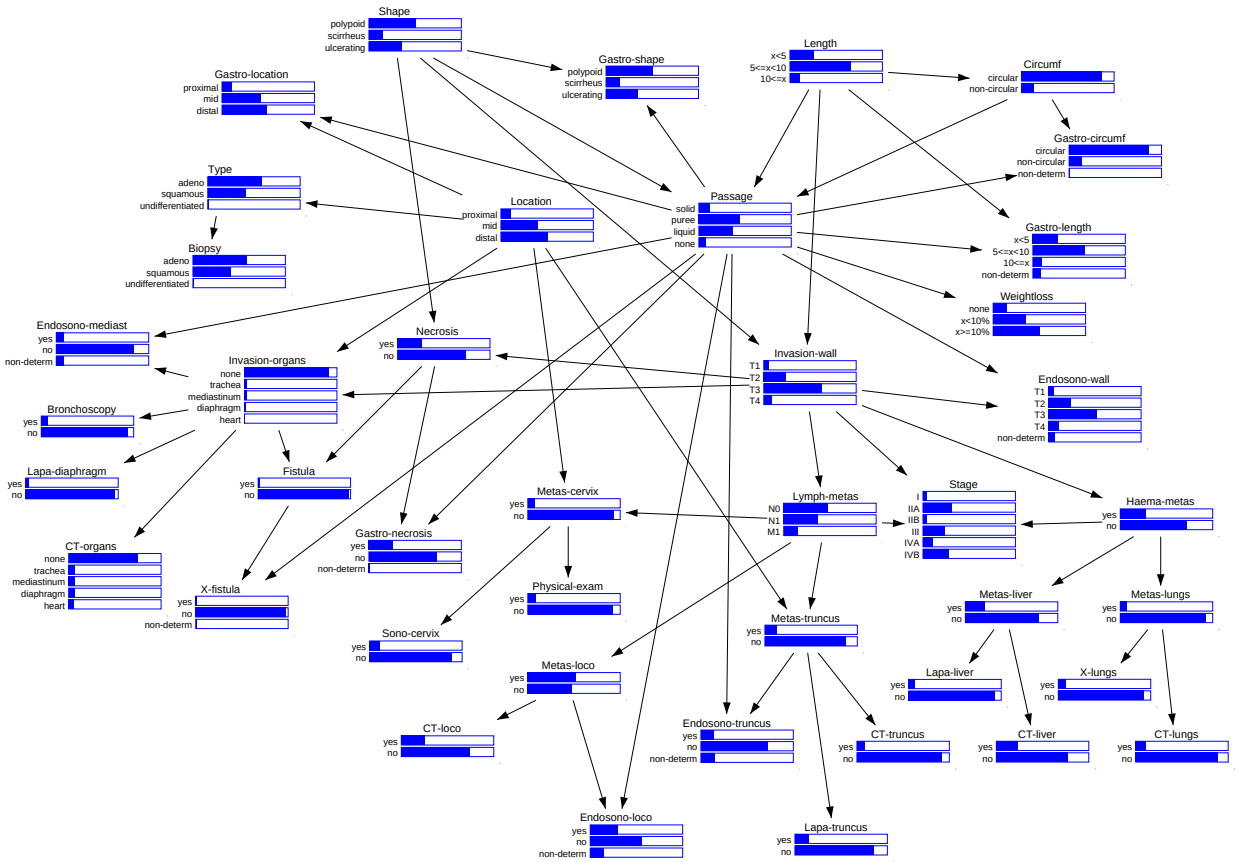

Figure 6: The Oesophageal cancer network

This network describes the presentation characteristics and the growth of an tumor in the *Oesophagus*. The Oesophageal cancer network has been developed for the Netherlands Cancer Institute, Antoni van Leeuwenhoekhuis, where they are specialized in treatments for cancer patients. This Bayesian network takes the length and shape of a tumor into account, but also the pathophysiological processes underlying the invasion into the oesophageal wall and its metastasis. The network further captures the sensitivity and specificity characteristics of the diagnostic tests that are typically performed to assess the stage of a patient's cancer. The symptoms and test results of a patient can be entered in this Bayesian network, for which it can then predict the most probable stage of the patient's cancer. The Oesophagus network was largely developed with the help of two domain experts, B. Taal and B. Aleman of the Antoni van Leeuwenhoekhuis and knowledge engineer L.C. van der Gaag and later on in the probability elicitation phase S. Renooij and C. Witteman were involved. In thesis we have relied on the expertise of L.C. van der Gaag and S. Renooij which they obtained during the construction of this network.

In Figure 6 we can see the actual Oesophageal cancer network. This network consists of 41 nodes, where 25 nodes in the network are leaf nodes. In other words there are 25 evidence nodes for which we assume that they are all instantiated (otherwise part of the graph can be pruned) and the node of interest is *Stage*. So with the Edmonds-Karp we would like to find a small set of nodes that could say something about the

nodes n	10	20	30	40	50	60	70	80	90	100
edges m	15	30	45	60	75	90	105	120	135	150
All three	0.70	0.80	0.79	0.76	0.76	0.69	0.74	0.72	0.81	0.76
One-Hel	0.86	0.98	0.99	0.99	0.95	0.96	0.97	0.97	0.99	1.00
One-MI	0.71	0.80	0.79	0.76	0.77	0.69	0.74	0.72	0.81	0.76
Hel-MI	0.82	0.81	0.79	0.77	0.78	0.69	0.74	0.74	0.81	0.76

Table 5: Percentage of equal sets: n nodes, $1.5n$ edges

state that the node *Stage* has.

5.6.3 Experiments

Now that we have seen how we will create randomly generated Bayesian networks and what the Oesophagus network describes, we will go on with our experiments. In this section we are going to show the results of the experiments we have done on these networks. First we will look at the experiments we have done with the randomly generated Bayesian network. Later on we will look at the experiments done with the Oesophagus Cancer network.

Randomly generated Bayesian networks
We have applied our three methods to each of the generated Bayesian networks in order to find out what the results are of these methods. For the creation of the random networks we have varied strongly with the number of nodes and the number of edges. We want to find out if the results would differ if we would look at larger and/or denser graphs. For our experiments we have varied the number of nodes n from $n = 10$ till $n = 100$ in steps of 10 and for each value of n we choose the number of edges m to be equal to $m = \frac{3}{2}n$ and $m = 3n$ respectively. So for a random network with 10 nodes we had an experiment with 15 edges and one with 30 edges. These values were chosen to achieve a reasonable density for a Bayesian network; running experiments with more denser graphs took too long. Even now we were not able to finish all experiments due to extreme long running times. For each combination of n nodes and m edges we create one-hundred random Bayesian networks and for every network we found the minimum cuts given one of the three weight assignment functions.

Completely equal sets of intermediate nodes
In Table 5 we can see the percentage of minimum cuts that are completely equal to each other for n nodes and $1.5m$ edges. If we look for instance at 10 nodes and 15 edges, we see a value of 0.86 for the comparison of the intermediate nodes of One-for-All and Hellinger influence. This means that in 86 of the 100 experiments the minimum cut sets consist of exactly the same nodes. In Table 6 we find the results of the experiments with three times the number of edges.

In both tables we distinguish between all of the experiments and comparing each pair of experiments with one another. We can see that the result of all the methods combined is always less than or equal to the lowest value of the pairwise comparison. This is of course logical, because the overlap of two sets will never be larger when a third set is added. In these tables the most interesting results are the ones in which we compare two methods with each other. When we compare One-for-All and Hellinger influence we can

see that in all cases, except one ($n = 10, m = 30$), the percentage of equal sets is strictly larger than for the combinations of the other two methods. It is likely that the other two combinations having a lower value is due to the effect of the method for mutual information. When we looked at the raw data, to see which sets of intermediate nodes were actually found, we established that the set found by mutual information often consists of more nodes than of the other two methods. In these tables we look at completely equal sets, so if one set contains more nodes than the other two the percentage of equality goes down. Later on we will look at partially overlapping sets and then we can see that mutual information almost always contains nodes that are also contained in the sets found by One-for-All and Hellinger.

In Table 5 we can see that the combination of One-for-All and Hellinger influence gives roughly the same answer in most cases. Most results are above 0.95, which means that only a handful of experiments did not have exactly the same results. From these results it seems unnecessary to use a computationally hard algorithm like the method for Hellinger influence if a simple method like One-for-All gives the same output. When we compare One-for-All or Hellinger influence with mutual information we can see that this value drops to a value between 0.71 and 0.82. Recall that mutual information sometimes gives slightly different results, which is due to the fact that this method in some cases gives back larger sets.

In Table 6 we see that the cases in which the minimum cuts are equal is significantly lower than in Table 5. Recall that we were unable to run these experiments for graphs with more than 50 nodes and three times as much edges. So when the density of the Bayesian network increases, the sets of intermediate nodes differ more from each other. We can see this in both the comparison with all three methods and with pairs of methods. In almost all of the comparisons between One-for-All and Hellinger influence we can see that the number of equal sets is larger than fifty percent and there is only one case for which the percentage of equality is smaller than fifty percent.

In these experiments with three times as much edges as nodes the value of One-for-All and Hellinger influence is not strictly larger than a comparison between two other methods. But this is only in the experiment with 10 nodes, in all other experiments it is strictly larger. It also seems that when the size of the randomly generated Bayesian networks increases the value of the percentage of equal sets rises. When we start with a value of 0.36 it increases to a value of 0.81. Furthermore we see that when we compare any method with mutual information it gives less similar results, just like we have seen in Table 5.

Partial overlap among sets of intermediate nodes

In Tables 5 and 6 we have looked at the cases in which the sets of intermediate nodes were completely equal to each other. But it is definitely interesting to look if there is any overlap at all. If the sets have overlap we know that the methods have found at least one node that is in both sets. In Table 7 and Table 8 we can see the percentage of partial overlap of the sets of intermediate nodes. These values include the sets that are

nodes n edges m	10 30	20 60	30 90	40 120	50 150
All three	0.22	0.26	0.27	0.33	0.34
One-Hel	0.36	0.54	0.70	0.73	0.81
One-MI	0.24	0.29	0.31	0.35	0.34
Hel-MI	0.58	0.40	0.31	0.41	0.41

Table 6: Percentage of equal sets: n nodes, $3n$ edges

nodes n	10	20	30	40	50	60	70	80	90	100
edges m	15	30	45	60	75	90	105	120	135	150
All three	0.96	1.00	1.00	1.00	0.99	0.99	0.99	0.99	1.00	1.00
One-Hel	0.96	1.00	1.00	1.00	1.00	1.00	0.99	1.00	1.00	1.00
One-MI	0.96	1.00	1.00	1.00	0.99	0.99	0.99	0.99	1.00	1.00
Hel-MI	1.00	1.00	1.00	1.00	0.99	0.99	1.00	1.00	1.00	1.00

Table 7: Percentage of partial overlap: n nodes, $1.5n$ edges

completely equal to each other, i.e. the percentages from Tables 5 and 6 are included in these values. The most interesting thing to notice is that in almost every single experiment, the set of intermediate nodes has at least some overlap among all of the three measurements. This means that there is at least one node that is in all three sets of intermediate nodes.

Average size of the sets of intermediate nodes
In Table 9 and Table 10 we can see the average size of the minimum cuts the methods have found. The randomly generated Bayesian networks in Table 9 are very sparse and we can see this in the results. The average of each method is very close to one, especially the method that uses the Hellinger influence barely finds any sets that are larger than one. The method that uses mutual information gives the largest sets. More often than the others this method finds sets that are larger than one node. In Table 10 we can see that the average size is slightly increased, which is a logical consequence for a denser Bayesian network. Interesting to see is that Hellinger influence again gives sets of a size close to one. One-for-All and mutual information both give larger sets, which are both around the value of two.

Duration of the experiments
Finally the two tables (Tables 11 and 12) should give an impression of the time it takes to compute the sets and run each of the experiments[4]. We can clearly see the difference among the three methods. One-for-All always returns the results in less than half a second and the other two methods take significantly more time.

[4]OS: Windows 7 Professional N (64-bit), CPU: Intel i7-3610QM, RAM: 8.00GB, Programming language: JAVA 7

nodes n	10	20	30	40	50
edges m	30	60	90	120	150
All three	0.98	1.00	1.00	1.00	1.00
One-Hel	0.98	1.00	1.00	1.00	1.00
One-MI	0.99	1.00	1.00	1.00	1.00
Hel-MI	1.00	1.00	1.00	1.00	1.00

Table 8: Percentage of partial overlap: n nodes, $3n$ edges

nodes n	10	20	30	40	50	60	70	80	90	100
edges m	15	30	45	60	75	90	105	120	135	150
One-for-All	1.23	1.03	1.01	1.00	1.06	1.07	1.02	1.06	1.00	1.00
Hellinger influence	1.01	1.00	1.00	1.01	1.04	1.02	1.05	1.02	1.03	1.00
Mutual information	1.23	1.23	1.25	1.34	1.29	1.39	1.35	1.46	1.37	1.33

Table 9: Average size minimum cut: n nodes, $1.5n$ edges

nodes n	10	20	30	40	50
edges m	30	60	90	120	150
One-for-All	2.85	3.21	2.00	1.58	1.25
Hellinger influence	1.10	1.16	1.27	1.21	1.19
Mutual information	1.76	2.11	2.37	2.27	2.28

Table 10: Average size minimum cut: n nodes, $3n$ edges

nodes n	10	20	30	40	50	60	70	80	90	100
edges m	15	30	45	60	75	90	105	120	135	150
One-for-All	62	61	64	95	78	77	32	31	141	76
Hellinger influence	2314	3732	8619	13943	5528	22532	45354	94811	166041	395856
Mutual information	501	1045	1969	4038	6897	12945	27833	60293	109758	261351

Table 11: Elapsed time (ms): n nodes, $1.5n$ edges

In Table 11 we see that our largest experiment with Hellinger influence takes 395856 milliseconds ($\approx$ 6.6 minute) and with One-for-All it only take 76 milliseconds. The results in Table 12 show this same effect, where One-for-All is done within a second and the other two methods take exponential time to calculate.

Another observation we can make is that the experiments of Hellinger influence and mutual information take much more time due to the density of the Bayesian network. The calculations of the posterior probabilities grow exponentially because nodes have more parents. We were unable to calculate results for 60 or more nodes and our largest experiment with 50 nodes and 150 edges took more than 22.2 hours for just the Hellinger influence to calculate.

In this section we have seen how all three methods work on randomly generated Bayesian networks. First, we have compared the set of intermediate nodes found by all of the methods. We have looked at completely equal sets and partially equal sets. Here we saw that in sparse networks the completely equal sets were quite similar ($\approx$ 75%), but when the network becomes denser the results significantly drop in equality. However when we looked at partial overlap the results show that in almost all of the cases there is at least one node that is the same in each of the sets of intermediate nodes. The average size of the sets showed that One-for-All and mutual information give back slightly larger sets than the Hellinger influence for which the size was

nodes n	10	20	30	40	50
edges m	30	60	90	120	150
One-for-All	47	62	140	125	249
Hellinger influence	5029	64746	905479	6047967	80185283
Mutual information	572	5683	71179	409542	4614083

Table 12: Elapsed time (ms): n nodes, $3n$ edges

almost always equal to one. When we compare running times among the three methods, One-for-All has a running time that is negligible if we compare it to the other two methods. Hellinger influence was clearly the slowest of all the methods, but also mutual information made it so that running experiments, with more then 50 nodes and 3 times the amount of edges, was too hard.

These experiments were all done on randomly generated Bayesian networks, which means that there was no underlying meaning behind these network. So at this point it is still hard to conclude that one method is clearly better than the other, so in the next section we will try our method on a Bayesian network that has been created for an actual real life problem.

The Oesophageal cancer network
In order to find out which of our three methods gives which set of intermediate nodes we have applied the Oesophageal cancer network to our program. Interesting to see was that all of our methods return the same result. Every time the Edmonds-Karp algorithm returns the following set of intermediate nodes:

$$\{Location, Shape, Length\}$$

It is interesting to see that One-for-All gives back the same results as the Hellinger influence and mutual information even though these methods have to do computationally hard calculations. So this goes back to what we have seen in the previous section, that One-for-All works as a heuristic in most cases equally well as Hellinger influence and mutual information.

All methods found the same set of intermediate nodes, but the goal of this section was to find out if this set of intermediate nodes actually can be used in an explanation to say something about the value of the node of interest, namely *Stage*. In order to find out if these nodes are meaningful, we showed the results to experts on this Bayesian network, namely S. Renooij and L.C. van der Gaag. They stated that these three nodes are very good for a possible explanation, because they describe the state of the cancer by Location, Shape and Length which are very important indications for the most probable state of *Stage*. So the set of intermediate nodes should be a good basis for the actual explanation.

5.6.4 Conclusion on finding sets of intermediate nodes

In this chapter we presented a method that is able to determine a set of intermediate nodes of a Bayesian network, which is a set of nodes that should explain the most probable value of the node of interest. In order to find this node we used the Maximum-Flow-Minimum-Cut theorem and the associated Edmonds-Karp algorithm. For the assignment of weights to the edges we presented three different methods, where the first is completely based on the structure of the graph and other two take the probability distribution of the Bayesian network into account as well. In order to find out which was best to use we applied the methods to randomly generated Bayesian networks and to the Oesophageal cancer network. In both experiments we

found that One-for-All was the best heuristic for the determination of the set of intermediate nodes. In the experiments with randomly generated Bayesian networks we were reluctant to state that One-for-All was the best method, because these networks did not encode any theoretical underlying situation. In the experiment with the Oesophageal cancer network it showed that all methods gave back the same results and due to the performance we concluded that we One-for-All was the best method to use.

Now that we have a set of intermediate nodes that can explain the most probable value of the node of interest, we are interested in explaining the most probable values of the intermediate nodes. This is what we will do in the second part of our method to explain the reasoning of a Bayesian network. We are going to present a method that will explain the most probable value of the associated intermediate node based on a set of clusters.

6 Clusters

In the previous chapter we discussed a method for finding a set of intermediate nodes based on the *Maximum-Flow-Minimum-Cut* theorem. Using this theorem we find a set of edges (the minimum cut) based on the structure of the network that makes the network disconnected when they are removed from the network. This set is minimal in size, which means that the set contains the least number of edges for which this condition holds. Given this set of edges we can determine a set of intermediate nodes by taking all nodes on the side of the edge furthest away from the evidence nodes. The set of intermediate nodes contains nodes that intend to summarize cause-effect relations of the network in order to explain the most probable value of the node of interest. In this chapter we will take these intermediate nodes and present a way to explain their most probable value given the entered evidence.

The method that we will present in this chapter takes the set of intermediate nodes as input. We are going to explain the most probable value of these intermediate nodes. If we go back to the Oesophageal cancer network in Figure 6 and take a look a the node *Shape*, we can see that this node has almost all evidence nodes as descendants. From the 25 evidence nodes present in this network only one node, namely *Biopsy*, is not a descendant of the node *Shape*. The idea of finding intermediate nodes was to find a set of nodes that was between the node of interest and the evidence nodes. With this set of nodes we could go a step further than just creating an explanation for the hypothesis purely based on the evidence nodes. Now that we have this set of intermediate nodes we still have the case that a node might be explained by all evidence nodes in the network. In this chapter we present a way to cluster evidence nodes in order to find combinations of evidence nodes that can explain the most probable value of the intermediate node.

In this chapter we will start with explaining how we generate clusters. After that we are going to use mutual information to find an ordering among the clusters. This ordering defines which set of evidence we are going to address first in our final explanation. Once we have defined everything that is necessary we are going to apply these methods to the Oesophageal cancer network we have seen earlier.

6.1 Generating clusters

We have presented a way of finding the set of intermediate nodes in the previous chapter. Now we will find the associated evidence nodes that have the most influence on the posterior probability of the intermediate node. We will be doing this by creating clusters for each of the intermediate nodes. The goal of these clusters is to find subsets of evidence that are present in the Bayesian network and with which we can explain the value of the intermediate node. An intermediate node can have more than one cluster and later on we will present a way to order the clusters based on the influence they have on the associated intermediate node. The clusters will always include at least one evidence node. If we have these clusters, we can make an ordering with the use of mutual information, which we have seen earlier. In this section we will focus on the definition and the creation of clusters.

Figure 7 shows a few small Bayesian networks, which will be used for explaining the creation of clusters. This Figure 7 shows the same network three times, but for now we will only be focusing on the leftmost one. Consider a non-empty set of intermediate nodes I found by the Edmonds-Karp algorithm in combination with one of the three methods. For each of the nodes $N \in I$ we can create a set of clusters $C = \{C_1, ..., C_n\}$. The number of clusters n is based on the number of outgoing edges an intermediate node N has. In other words the number of clusters an intermediate node N has is equal to the number of children this intermediate node N has.

In Figure 7 we can see node A has two children and therefore two outgoing edges, namely to node B and to node C. Given the definition we should get two clusters, but in order to generate clusters we first have to define these clusters.

Definition 6.1.

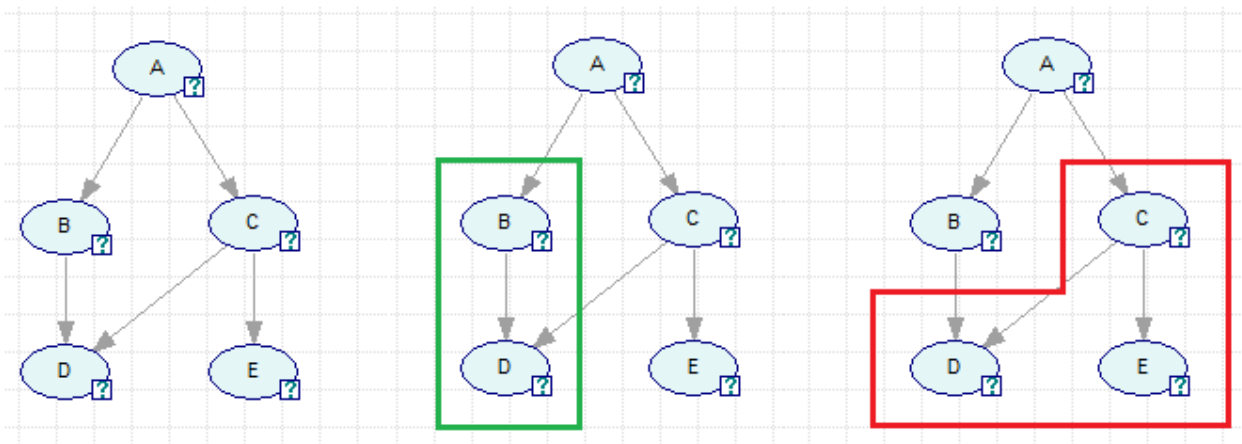

Figure 7: Example of clusters

Consider an intermediate node N, and let X be a child of N. Then a cluster C_i is the subgraph $G_{C_i} = (V_{C_i}, E_{C_i})$ created by the reflexive transitive closure $\sigma^*(X)$. Let $|\sigma(N)| = n$ then the cluster set C for N consists of clusters for all n children.

For our example in Figure 7 this means we can create two clusters, namely one for node B and one for node C. If we now look at the Bayesian network in the middle of the figure, we can see the cluster for node B. As the definition says we will look at the reflexive transitive closure of $\sigma^*(B)$ in order to find the subgraph which forms the cluster. Just as we can see in Figure 7 this consists of node B itself and evidence node D and the edge (B, D). We can do the same thing for the other child of intermediate node A and then we get the cluster shown on the right.

If we compare the two clusters with each other we can distinguish three ways in which these clusters can relate to each other. It is important to recognize these cases because later on it will help us to see how combinations of evidence relate to each other. We can have the following cases:

- Clusters are completely separated.
- Clusters are partially overlapping.
- One cluster is a subset of another cluster.

These properties will be important for the way we are going to construct our final explanation. In this explanation we are going to translate these clusters into arguments.

Definition 6.2.
An argument is the conjunction of the evidence nodes in the cluster with their value.

If clusters are completely separated we have sets of evidence that are completely separated effects of the associated intermediate node. For our explanation this means we can distinguish several reasons for the value of the intermediate node and with these evidence nodes we can construct arguments in our final explanation. When one cluster is a subset from another we can question ourselves if we might not need to explain the smallest argument because it is redundant or if the largest argument might consist of too many evidence nodes. Overlapping arguments can be similar if there is much overlap, but quite different when the overlap

is very small. In order to get more insight in when to explain which cluster, we are going to use mutual information once again. With mutual information we will construct an ordering among the clusters and then we will determine which cluster is the most influential and best to explain first.

6.2 Mutual information of a cluster

Mutual information is a method to determine how much information one variable has about another variable, this as have already seen in section 5.5.3. When one variable is influenced by several other variables we can use mutual information to determine which of these other variables contains the most information about this one variable. The variable with the highest mutual information is the most influential. We used mutual information to determine the strength of an edge, but now we want to apply mutual information to clusters, which means we will not be comparing edges with one another, but we will compare entire sets of nodes and edges with each other.

6.2.1 Mutual information for a cluster

Until now we have only seen how to determine the mutual information between two nodes, but in this section we will use mutual information for an entire cluster of nodes. Recall the equation for mutual information as stated in Section 5.5.3:

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} P(x,y) \log \frac{P(x,y)}{P(x)P(y)}. \tag{40}$$

Important to realize is that this equation gets two sets of variables as input. For assigning weights to edges we have only used sets of size one, i.e. individual nodes, but now we will actually use larger sets. When we look at the Bayesian network in the middle of Figure 7, we have an intermediate node A and a cluster consisting of nodes B and D. If we look at the network most right we have the same intermediate node A and a different cluster consisting of nodes C, D and E. This situation fits the use of mutual information, because we have the shared intermediate node A, and we want to know which cluster has the most influence on this node A. So in terms of mutual information we can compare the following two values:

$$I(\{A\}\,;\{B,D\}) \tag{41}$$

$$I(\{A\}\,;\{C,D,E\}) \tag{42}$$

By calculating these combinations of intermediate node and cluster, we have two values that allow us to compare the 'strength' of these clusters: the cluster with the highest value of mutual information is the cluster that has the most effect on the posterior probability of node A.

There is one major problem with this method and that is the running time of the algorithm. We have used mutual information before, but that was for an input of just two nodes. Now we are looking at sets of a variable size. The equation for mutual information iterates over all possible values of the given nodes. This means that the running time becomes $O(v_1 \cdot \ldots \cdot v_n)$, where v_i stands for the number of values of node i. For a Bayesian networks with n nodes, which are all binary-valued, the running time is already $O(2^n)$, i.e. exponential in the number of nodes. This behavior is of course undesirable and for a lot of network hard to calculate, so we propose an alternative method that still involves mutual information, but is computationally easier.

6.2.2 Alternative mutual information

During our research it quickly became clear that the running time of $O(2^n)$ of mutual information would restrict us extremely. The calculation of the mutual information of clusters for the Oesophageal cancer

network took too much time and therefore the experiment failed. In order to achieve results we decided to approach the calculation of mutual information slightly different. Recall that mutual information is a method to determine how much information one set of variables has about another set of variables. Now that we are not able to calculate these values for a cluster exactly, we can still try to capture the information of the cluster in one value by calculating the mutual information over all edges in the cluster C_i and sum up all these values.

$$I_{sum}(X;Y) = I(X;\sigma(X)) + \sum_{(u,v)\in E_{C_i}} I(u;v) \tag{43}$$

The underlying idea is that we take all cause-effect relations in the cluster into account. For the right most Bayesian network of Figure 7, this means that the summed mutual information $I_{sum}(A;C,D,E)$ [5] would be equal to:

$$I_{sum}(A;C,D,E) = I(A;C) + I(C;D) + I(C;E). \tag{44}$$

We will be using this alternative approach in order to be able to construct an ordering among the found clusters. In order to support our alternative method we tried to find properties that would hold for both the calculation of the exact mutual information as for the calculation of the summed mutual information. We were able to determine that a subset of a cluster and an intermediate node always have a lower mutual information than the intermediate node and the entire cluster. Let us first look at this property for the normal version of mutual information:

Proposition 6.1.
Consider an intermediate node X and two of its clusters Y and Z such that $Y \subseteq Z$. Then $I(X;Y) \leq I(X;Z)$.

Proof 6.1.
We know the result of mutual information always returns a non-negative value, so it holds that $I(A;B) \geq 0$. Therefore the following proof can be obtained:

$$\begin{aligned}
I(X,Z) &= \sum_{x\in X}\sum_{z\in Z} P(x,z)\log\frac{P(x,z)}{P(x)P(z)} && (45)\\
&= \sum_{x\in X}\sum_{z\in Z\setminus Y} P(x,z)\log\frac{P(x,z)}{P(x)P(z)} + \sum_{x\in X}\sum_{y\in Y} P(x,y)\log\frac{P(x,y)}{P(x)P(y)} && (46)\\
&= \sum_{x\in X}\sum_{z\in Z\setminus Y} P(x,z)\log\frac{P(x,z)}{P(x)P(z)} + I(X;Y) && (47)\\
&= I(X;Z\setminus Y) + I(X;Y) && (48)\\
&\geq I(X;Y) && (49)
\end{aligned}$$

This property also holds for the summed mutual information. In the following we will proof this proposition and we will do this in a similar fashion as we did with the previous proposition. We again take into account the outcome of mutual information is always a non-negative value.

Proposition 6.2.
Consider an intermediate node X and two of its clusters Y and Z such that $Y \subseteq Z$. Then $I_{sum}(X;Y) \leq I_{sum}(X;Z)$.

[5] For the ease of explanation we have left out the braces and commas, that indicate the set structure.

Proof 6.2.
We know the result of mutual information always returns a non-negative value, so it holds that $I(A;B) \geq 0$. Therefore the following proof can be obtained:

$$I(X,Z) = I(X;\sigma(X)) + \sum_{(u,v)\in E_Z} I(u,v) \tag{50}$$

$$= I(X;\sigma(X)) + \sum_{(u,v)\in E_{Z\setminus Y}} I(u,v) + \sum_{(u,v)\in E_Y} I(u,v) \tag{51}$$

$$= \sum_{(u,v)\in E_{Z\setminus Y}} I(u,v) + I(X;Y) \tag{52}$$

$$\geq I(X;Y) \tag{53}$$

We have seen how we can calculate the summed mutual information and the shared properties. The summed mutual information also has disadvantages, because it is a summation of several individual values. The mutual information of the intermediate and its direct child can be equal or very close to the mutual information of the intermediate node and the entire cluster. This makes it that the result of the summed mutual information can be much larger than the exact mutual information. Note that it is not a upper bound for the exact mutual information. Nonetheless we will use the summed mutual information on the Oesophageal cancer network. We are going to look at which clusters can be found and how they relate to each other.

6.3 Clusters in the Oesophageal cancer network

To find out how our method performs we applied the method of generating clusters on the Oesophageal cancer network that we have seen in the previous chapter in Figure 6. With the use of the *Maximum-Flow-Minimum-Cut* theorem we already found a set of intermediate nodes consisting of $\{Location, Shape, Length\}$. In this section we will look at the clusters of each of the intermediate nodes and their ordering in terms of mutual information. Initially we wanted to apply both the exact calculation of mutual information and our simplified version where we sum up all edges in the cluster, but this network was already too complex for mutual information to be calculated exactly. In this section we will analyze all intermediate nodes individually and see if we have a useful ordering. We will have a section for each of the intermediate nodes and these are associated with graphical representations of the clusters. In these figures we will see the Oesophageal cancer network like shown in Figure 6, but the intermediate node will have a green border and the clusters a red border. To indicate which cluster is based on which edge we have colored the associated edge green.

6.3.1 Location

The first intermediate node we are analyzing is the node *Location*. Figure 8 shows the clusters that can be created for this intermediate node. The green rectangle indicates the intermediate node *Location* and the red shapes represent the clusters. The node *Location* has five (green) outgoing edges which means we can construct five clusters. This figure also shows that all of the clusters are completely separated from one another and not one cluster is overlapping. Due to the structure of the clusters and the case that there is no overlap this is the ideal case for an explanation. With this structure it would be possible to construct five separate arguments for the value of the intermediate node. All of the clusters have individual evidence nodes and for each of the clusters the combined evidence can be viewed as being the result of a situation captured in the intermediate node and can therefore explain its value.

In order to find out which of the clusters has the most influence on the associated intermediate node we are going to calculate the mutual information of each of the clusters. The cluster with the highest

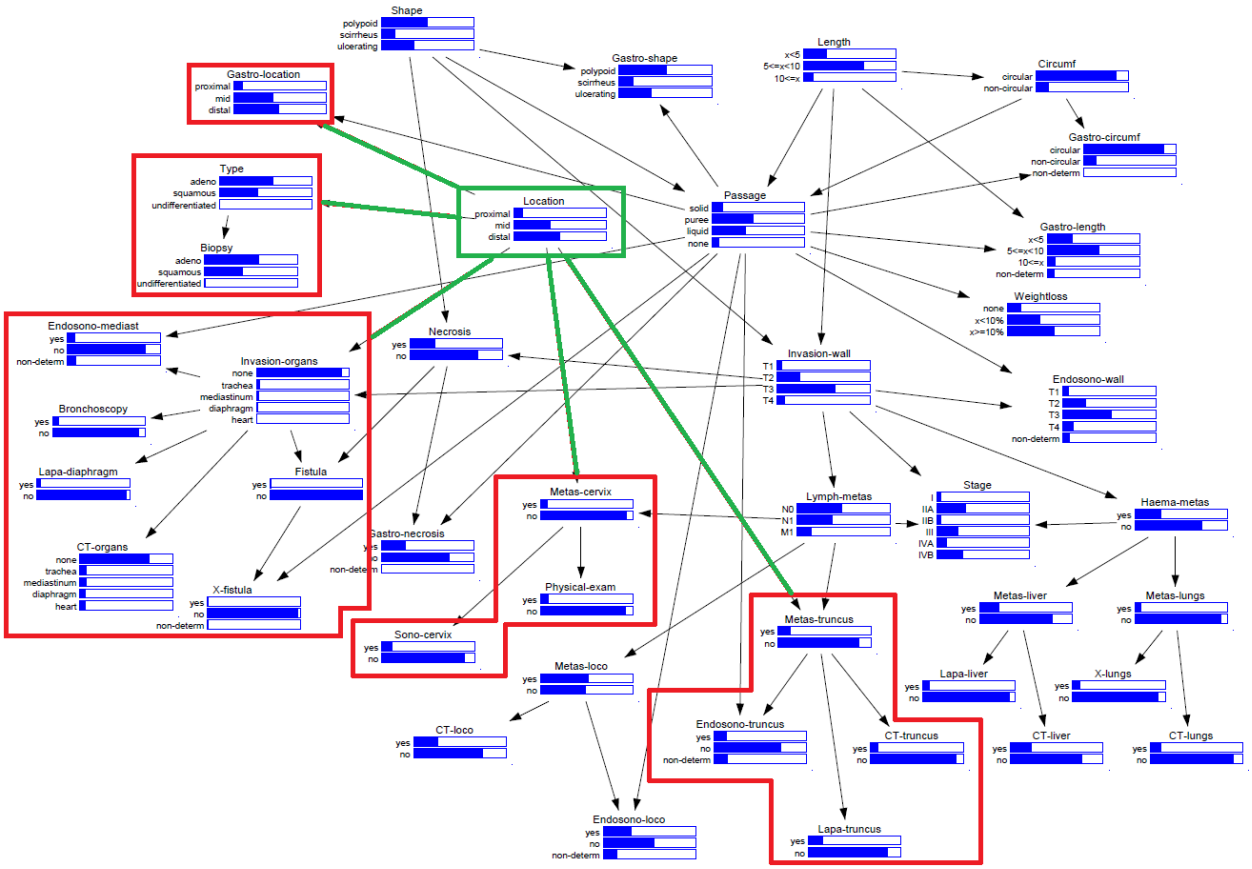

Figure 8: Clusters for intermediate node Location

mutual information is the most influential and therefore should explain the most about the value of the intermediate node. Note that we were unable to calculate the mutual information exactly and we chose to use the summation of all pairs of nodes in the cluster that are connected by an edge. This resulted in the following decreasing ordering, where we first indicate the found value of the summed mutual information of the cluster, followed by the evidence nodes that are contained in the associated cluster.

1. (1.2956) $\{Biopsy\}$
2. (0.8334) $\{Gastro\text{-}location\}$
3. (0.7969) $\{Endosono\text{-}mediast, Bronchoscopy, Lapa\text{-}diaphragm, CT\text{-}organs, X\text{-}fistula\}$
4. (0.5440) $\{Endosono\text{-}truncus, Lapa\text{-}truncus, CT\text{-}truncus\}$
5. (0.5118) $\{Sono\text{-}cervix, Physical\text{-}exam\}$

We can see that the cluster containing node *Biopsy* has the strongest influence on the value of the intermediate node given the value of mutual information. In decreasing order we can see the evidence nodes that have a lower influence. Given this ordering and the fact that all sets are disjoint we can given five arguments in decreasing order of mutual information that can explain the value of the intermediate node *Location*. In the next chapter we use this information to create an example explanation.

6.3.2 Shape

The second intermediate node that we will be analyzing is the node *Shape*. This node has four outgoing edges, which means we will get four clusters and unlike what we have seen with the node *Location* we have overlap between the clusters. The visualization of these clusters is shown in Figures 9 and 10. In Figure 9 we can see two clusters, where one cluster is a subset of the other cluster. The smallest cluster only contains one node, namely *Gastro-shape*. The larger cluster also includes this node and ten other evidence nodes.

In Figure 10 we can see the other two clusters. Here we have again that one cluster is a subset of another cluster. The smaller cluster contains three nodes, where two of these nodes are evidence nodes, namely *X-fistula* and *Gastro-necorsis*. If we again look at Figure 9, we can see that these two evidence nodes are also included in the largest cluster. Only the third node, namely *Fistula*, is not in this cluster and therefore these clusters are only overlapping instead of one being a subset of the other. Furthermore we can see that the small cluster consisting of node *Gastro-shape* is completely separated from the large cluster in Figure 10. We can see that both the large clusters of Figure 9 and Figure 10 are overlapping and some evidence nodes are used for different arguments.

If we calculate the summed mutual information for this intermediate node we end up with the following ordering in decreasing order (Note that we obviously do not include the node of interest *Stage* in the set of evidence nodes for our explanation, although it is contained in the large cluster of Figure 10):

1. (9.7743) $\{Endosono\text{-}mediast, Bronchoscopy, Lapa\text{-}diagragm, CT\text{-}organs, X\text{-}fistula,$ $Gastro\text{-}necrosis, Sono\text{-}cervix, Physical\text{-}exam, CT\text{-}loco, Endosono\text{-}loco, Endosono\text{-}truncus,$ $Lapa\text{-}truncus, CT\text{-}truncus, Lapa\text{-}liver, CT\text{-}liver, X\text{-}lungs, CT\text{-}lungs, Endosono\text{-}wall\}$
2. (2.1102) $\{Gastro\text{-}shape, Gastro\text{-}circumf, Gastro\text{-}length, Weigthloss, Endosono\text{-}wall,$ $Endosono\text{-}truncus, Endosono\text{-}loco, Gastro\text{-}necrosis, X\text{-}fistula, Endosono\text{-}mediast, Gastro\text{-}location\}$
3. (1.3779) $\{Gastro\text{-}shape\}$
4. (1.0258)$\{X\text{-}fistula, Gastro\text{-}necrosis\}$

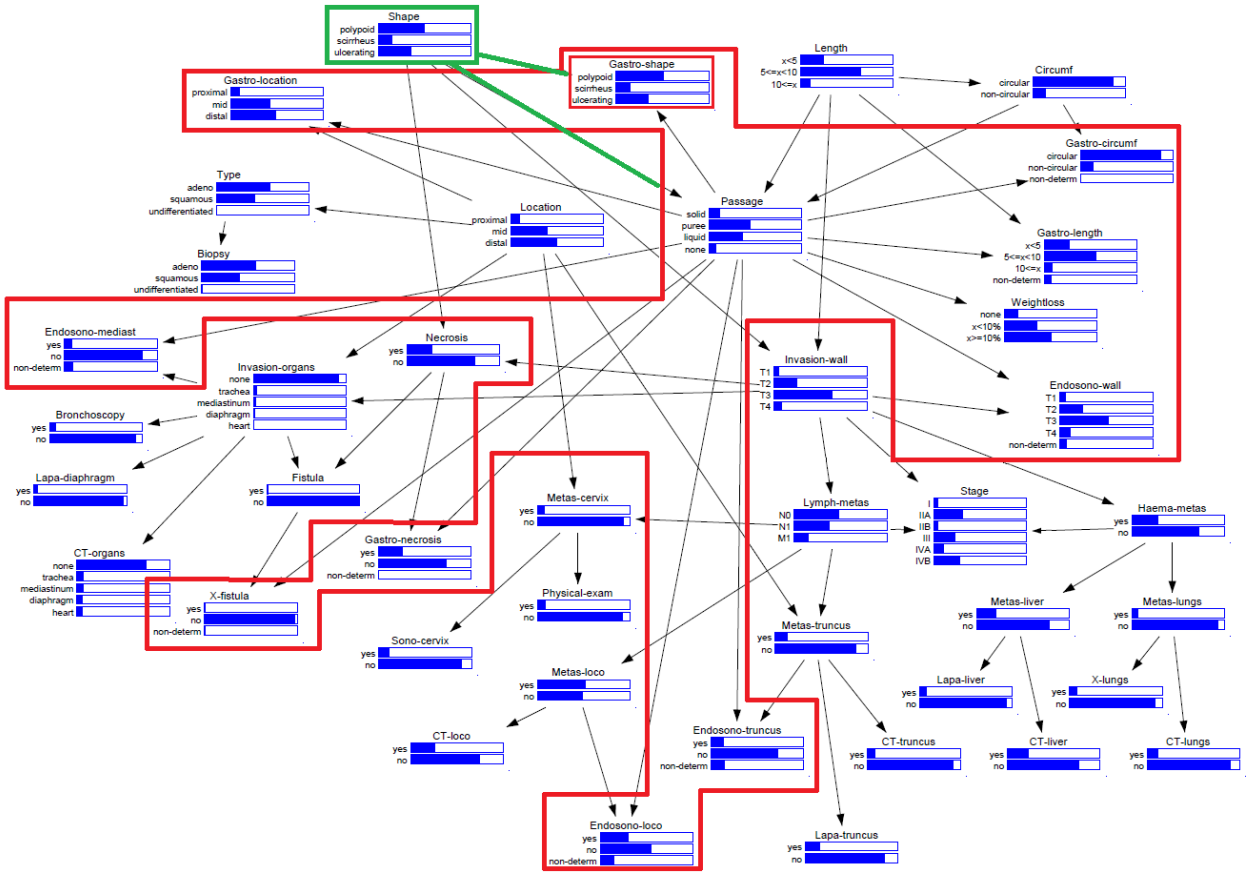

Figure 9: Clusters for intermediate node Shape (Part 1)

In this ordering we see that the sets with the highest mutual information are the largest sets. This is what we expected, because set 3 is a subset of set 2 and we know that the mutual information of set 3 must be smaller. This argument also holds for set 4 which is a subset of set 1, like we have seen in Figure 10. This ordering implies that we may first want to explain the two largest clusters and then the smaller ones. How we handle this in the explanation will be shown in the next chapter.

6.3.3 Length

The final intermediate node we are analyzing is the node *Length*. Just like the previous intermediate node we have split up the visualization of the clusters in Figures 11 and 12. In Figure 11 we can see three clusters, but we have visualized this slightly differently. We can see three outgoing edges from the intermediate node, where one goes to the cluster containing one node and one edge goes to the node *Passage*. The last edge goes to *Circumf*, which then goes to *Passage* and creates the same cluster except the node *Circumf*. We chose this visualization just to keep the figure clear and not too crowded with red lines. We can see that this figure contains several subsets. All other clusters are subsets of the cluster created by the intermediate node *Circumf*. The cluster containing only node *Gastro-length* is a subset of both other clusters. So here we see

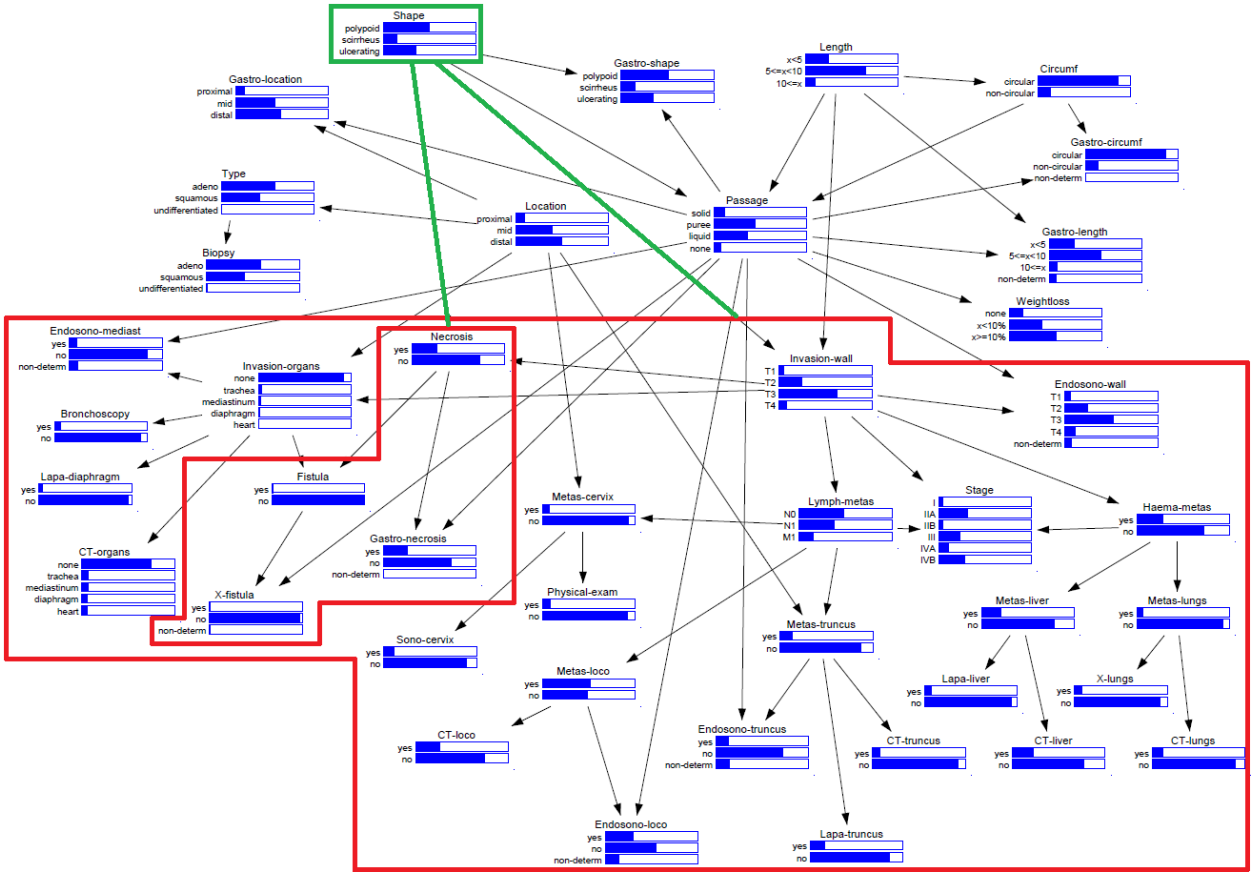

Figure 10: Clusters for intermediate node Shape (Part 2)

again a different situation, with clusters being subsets of other clusters. Note also that the cluster created by the edge between *Length* and *Passage* is the same as the cluster in Figure 9, because both intermediate nodes have *Passage* as a direct child. This means we can make the same argument as an explanation for the value of both the intermediate nodes.

In Figure 12 we see the fourth cluster for the intermediate node *Length*. This cluster is disjoint from the smallest cluster in Figure 11, but has overlap with the other two clusters. We have also seen this cluster before with intermediate node *Shape*. The node *Invasion-wall* is also a direct child of both intermediate nodes and therefore we also see this cluster in Figure 10.

If we again look at the result of calculating the summed mutual information in decreasing order, we end up with the following:

1. (9.8947) $\{Endosono\text{-}mediast, Bronchoscopy, Lapa\text{-}diagragm, CT\text{-}organs, X\text{-}fistula, Gastro\text{-}necrosis, Sono\text{-}cervix, Physical\text{-}exam, CT\text{-}loco, Endosono\text{-}loco, Endosono\text{-}truncus, Lapa\text{-}truncus, CT\text{-}truncus, Lapa\text{-}liver, CT\text{-}liver, X\text{-}lungs, CT\text{-}lungs, Endosono\text{-}wall\}$
2. (2.8952) $\{Gastro\text{-}shape, Gastro\text{-}circumf, Gastro\text{-}length, Weigthloss, Endosono\text{-}wall, Endosono\text{-}truncus, Endosono\text{-}loco, Gastro\text{-}necrosis, X\text{-}fistula, Endosono\text{-}mediast, Gastro\text{-}location\}$
3. (2.2313) $\{Gastro\text{-}shape, Gastro\text{-}length, Weigthloss, Endosono\text{-}wall, Endosono\text{-}truncus, Endosono\text{-}loco, Gastro\text{-}necrosis, X\text{-}fistula, Endosono\text{-}mediast, Gastro\text{-}location\}$
4. (0.8891) $\{Gastro\text{-}circumf\}$

Here we see that the cluster with the largest mutual information is the cluster shown in Figure 12. If we compare this value with the mutual information for this similar cluster in Figure 10 it is almost the same and this difference is only caused by the different edge from intermediate node to node *Invasion-wall*. Furthermore we see that results 2 and 3 have the same evidence as we noted earlier, but the cluster that includes node *Circumf* has a slightly higher result for our summed mutual information. The cluster that only includes the evidence node *Gastro-circumf* has the lowest mutual information and this is what we expected based on the fact that it is a subset of result 2 and 3.

In the next chapter we will use these results for creation of the final explanation. With the found clusters and ordering among these clusters we can construct arguments to explain the value of the intermediate node.

6.4 Summary of the creation of clusters

In our experiments on the Oesophageal cancer network we have seen that we find four to five clusters for each of the intermediate nodes. For the clusters of each of the intermediate nodes we have calculated the summed mutual information. Here we see that larger clusters almost always have higher values for the mutual information. This is to be expected when we sum up values. From several preliminary experiments it seemed that the probability that the summation of several values for mutual information is higher than a single value for mutual information is very high.

In the next chapter we will be translating the evidence in a cluster into an argument that should explain the value of the associated intermediate node. In our experiment we have also seen how sets of evidence (arguments) relate to each other. With the intermediate node *Location* we have seen five completely separated arguments and in the other experiment we have seen arguments that are partially overlapping and even arguments that are completely contained in other arguments. When one argument is contained in another argument it doesn't mean we do not have to explain the smaller argument even if it has a lower mutual information. The smaller argument might just include the evidence nodes that capture the reason why a intermediate nodes has a certain most probable value and in the larger argument the other evidence nodes might only have a small contribution to this reason.

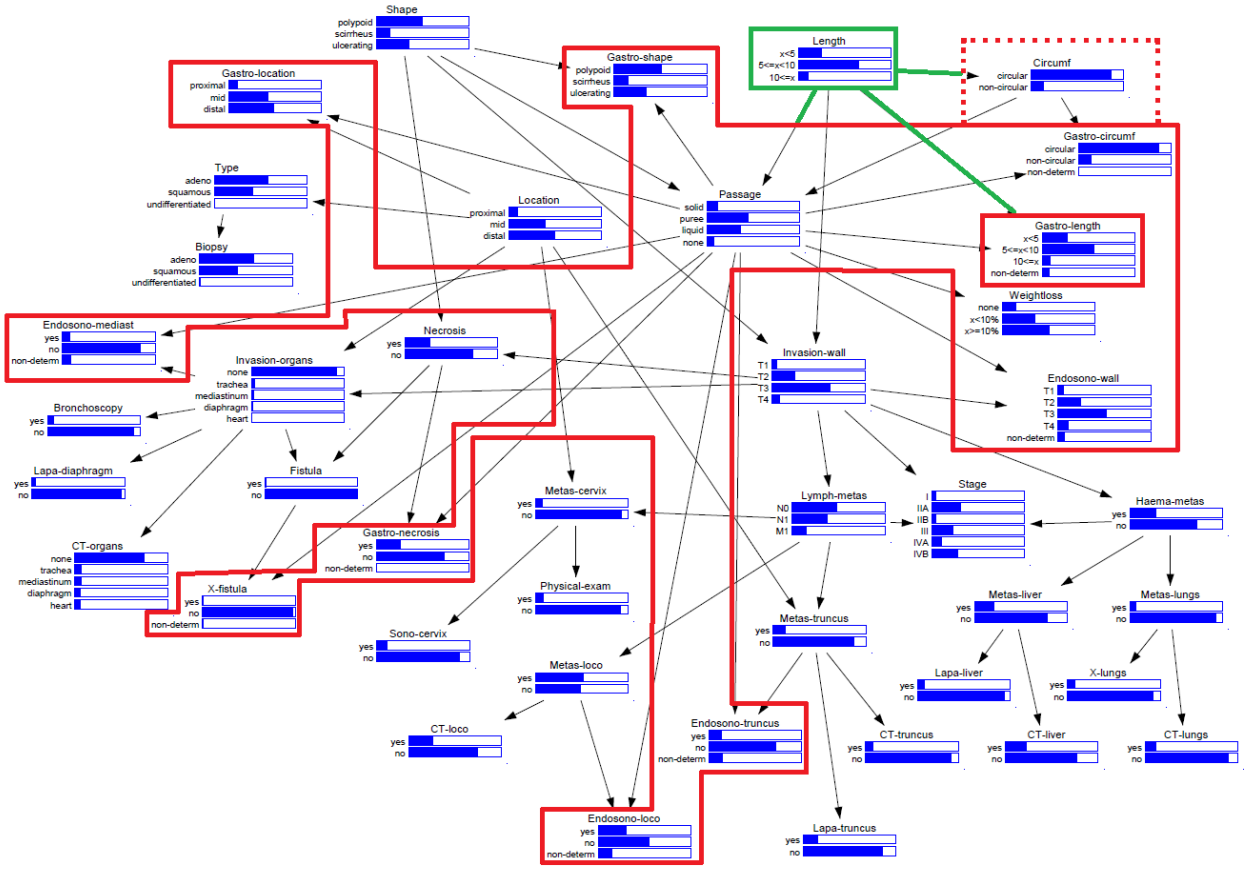

Figure 11: Clusters for intermediate node Length (Part 1)

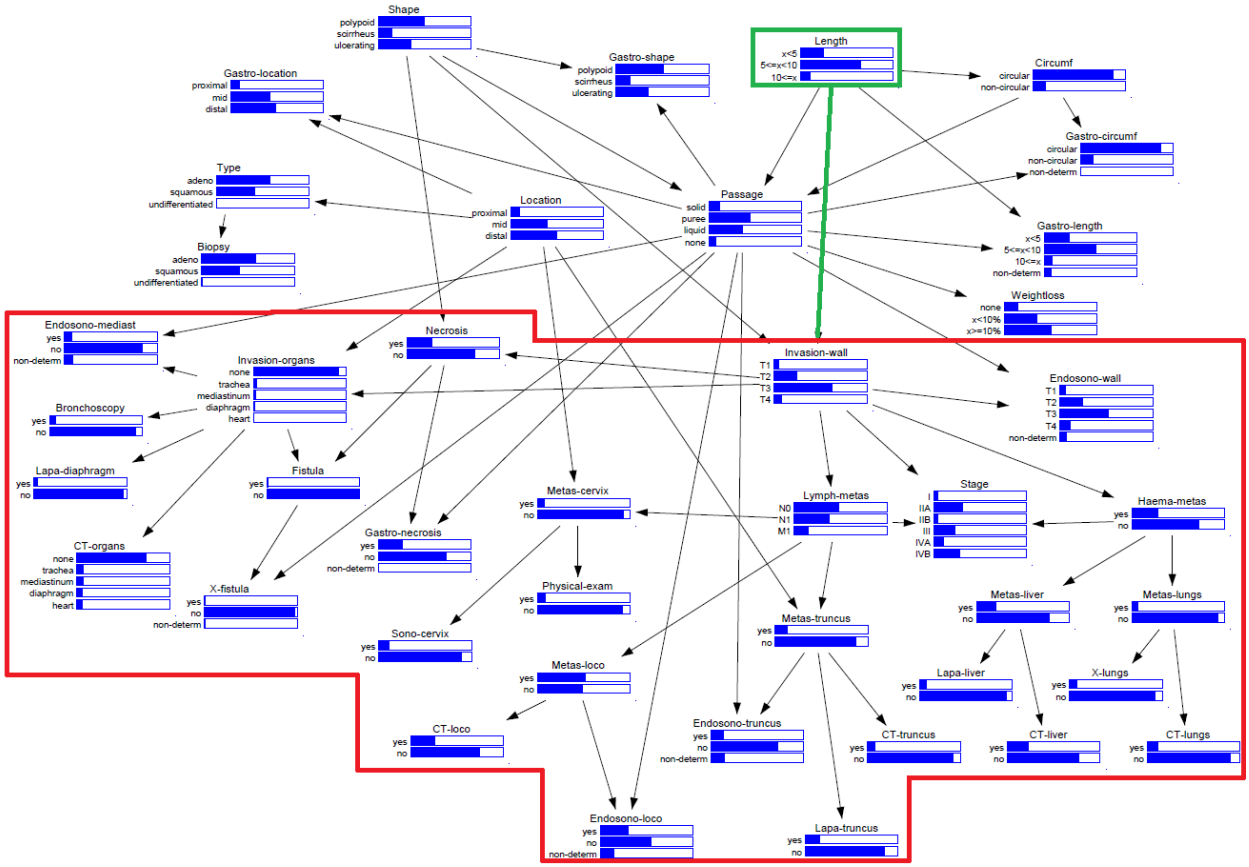

Figure 12: Clusters for intermediate node Length (Part 2)

7 The Explanation

The goal of this thesis was to come up with a method that would aid experts in understanding the reasoning of a Bayesian network. It should explain why the node of interest has a certain probability for a certain value. In this chapter we will present the actual explanation based on the methods we have presented in the earlier chapters. We started our method by finding a set of intermediate nodes in Chapter 5, which is a small set of nodes that should be somewhere between the evidence nodes and the node of interest. These nodes can be found based on the underlying structure of the Bayesian network, possibly in combination with the probability distributions. After that we presented a method that help us explain the most probably value of these intermediate nodes by the use of clusters in Chapter 6. For each intermediate node we constructed a number of clusters based on the number of children this intermediate node has. After we created the clusters for each of the nodes, we calculated the mutual information of the clusters in order to compare the clusters. With the use of mutual information we can state that one cluster is more influential than another cluster based on this value. The higher the mutual information the stronger the influence on the associated intermediate node. So with this information we can make an ordering among these clusters, where the cluster with the highest value is the most influential cluster. In each cluster there is at least one evidence node, but probably more, and these evidence nodes will aid in explaining the value of the associated intermediate node. The set of intermediate nodes combined with the ordering based on the clusters of each intermediate node will form the input for the ultimate explanation. In this chapter we will present a way of translating our input to an actual explanation. We will again illustrate our method using the Oesophageal cancer network.

7.1 Verbal Expressions

Until now we have focused mostly on the structure of the Bayesian network, but the probabilities are just as important for our final explanation. Interpreting probabilities is hard for human beings. The meaning of a probability can differ from case to case, where for instance a probability of 0.9 is extremely high for winning a game of cards, but it is too low if your life is depending on it. Assigning probabilities might be even harder, but luckily we are given a network with probabilities that already are assigned. In our explanation we want to translate the found probabilities back to verbal expressions. Druzdzel [8][10] addressed these verbal expressions for the explanation of decision support systems. In his paper he describes that verbal expressions are 'more digestible' for the readers of an explanation. In this paper he presents a simple table for the translation of probabilities into verbal expressions. In Table 13 we see this general translation. Here he translates probabilities in a certain range into adjectives or adverbs. In our explanation we are going to use this table in order to help the reader understand the probability for the value of a certain node. We will mostly be using the *Adjectives*, because we will be using sentences like: "Node X having value Y is impossible ($P = 0$)." Note that Table 13 is a general table and for a better explanation a translation that is more focused on the subject might be better.

7.2 Representation and User-interaction

Our explanation tool is designed as an interactive web page, which will consist of three levels of explanation. These levels define how much detail we give about the results. The first level is the actual conclusion of the network, so the most probable value of the node of interest. We will explain this value with a combination of verbal expressions and the actual probabilities as shown in the previous section. The second level consists of the intermediate nodes we have found for the Bayesian network. We represent the most probable values for these nodes in the same way as we do for the node of interest. The combination of intermediate nodes and the node of interest is the initial explanation as we will show it to an expert. This combination should give the expert an initial idea of the result, but when he is not satisfied yet we can zoom in and find more

Probability range	Adjectives	Adverbs
0.00	impossible	never
0.00 - 0.10	very unlikely	very rarely
0.10 - 0.25	unlikely	rarely
0.25 - 0.40	fairly unlikely	fairly rarely
0.40 - 0.50	less likely than not	less often than not
0.50	as likely as not	as often as not
0.50 - 0.60	more likely than not	more often than not
0.60 - 0.75	fairly likely	fairly often
0.75 - 0.90	likely	commonly
0.90 - 1.00	very likely	very commonly
1.00	certain	always

Table 13: Mapping from probability ranges to verbal expression [8].

details for each of the intermediate nodes.

Because we represent the results in a web page we can expand the information of the intermediate node and give a third level of explanation. To explain the value of the intermediate nodes we have constructed a set of clusters for each of them. Due to our ordering made by the summed mutual information we can order the clusters by strength of influence, where the first argument is the cluster with the highest mutual information. Recall that an argument consists of the conjunction of the evidence nodes that are present in the associated cluster combined with their observations as entered in the network. In order to capture the properties of clusters being disjoint, partially overlapping or subsets, we strengthened these arguments with a Venn-diagram of the situation. In order to illustrate an actual example we will be looking at the Oesophageal cancer network once more. For this Bayesian network we will enter evidence based on an example patient and construct an explanation.

7.3 Explanation of Oesophageal cancer network

In Figure 6 we can see the Oesophageal cancer network as used in the experiments of Chapter 5 and Chapter 6. In this Bayesian network the node of interest is the node *Stage* and the set of intermediate nodes was: $\{Location, Shape, Length\}$. In Chapter 6 we have found four to five clusters for each of the intermediate nodes. In this section we will use the earlier found results on the Oesophageal cancer network in order to generate our final explanation. We already discussed that we will be using a combination of verbal expressions and probabilities. We also described the representation and the user-interface, where we can zoom in to get a more detailed description and combine it with a graphical representation like a Venn diagram

In order to generate an explanation of the Oesophageal cancer network, we need to give the Bayesian network input for the states of the evidence nodes. For this we were given an example patient with realistic values for the evidence nodes as we can see in Table 14. When we entered this evidence in the Oesophageal cancer network we found that the most probable value for *Stage* was *IVA* with a probability of 0.901. This is the information we need for the first level of explanation the network. Level two was based on the values of the intermediate nodes $\{Location, Shape, Length\}$. The most probable values for these nodes were: *distal* for *Location* with 0.981, *scirrheus* for *Shape* with 1.000 and $5 \leq x < 10$ for *Length* with 0.984. The third level were the clusters and their ordering which we will use as arguments to explain the intermediate nodes.

Node	**Value**
Weightloss	<10%
Gastro-circumf	circular
Gastro-length	5-10 cm
Gastro-shape	scirrheus
Gastro-location	distal
Gastro-necrosis	no
Endosono-wall	T3
Endosono-mediast	no
Endosono-loco	yes
Biopsy	adeno
X-fistula	no
CT-organs	none
CT-lungs	no
CT-loco	yes
CT-liver	no
X-lungs	no
Bronchoscopy	no
Lapa-diaphragm	no
Lapa-truncus	yes
Lapa-lever	no
Sono-cervix	no
Endosono-truncus	non-determ
CT-truncus	yes
Physical-exam	no

Table 14: Example patient for the Oesophagus network

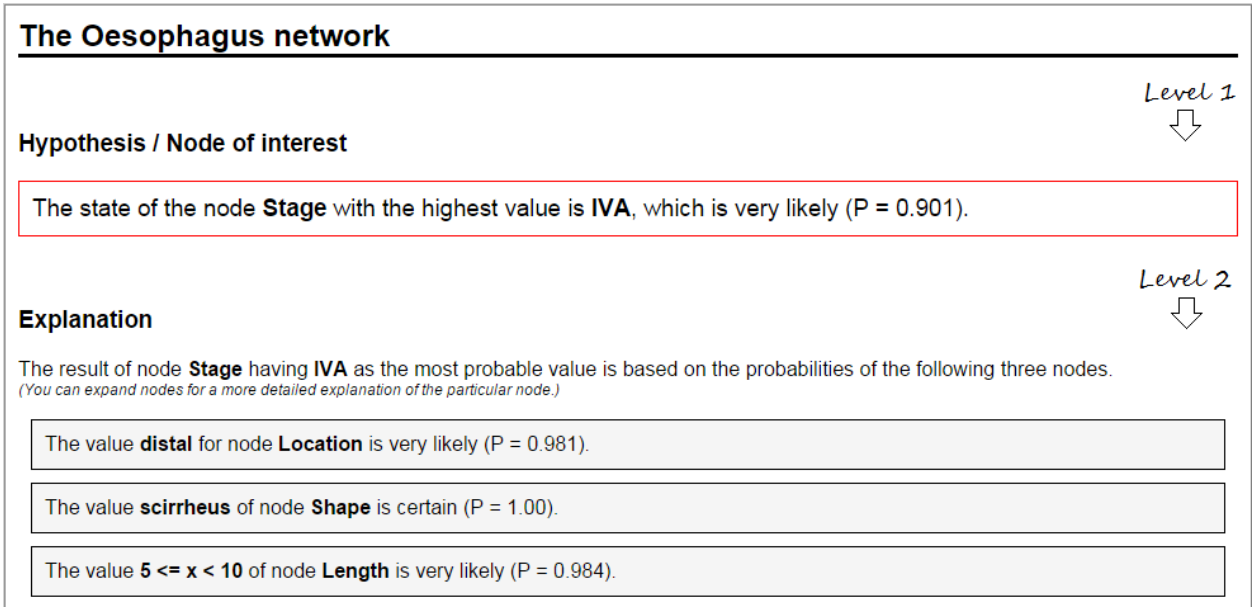

Figure 13: First and second level of explanation.

In Figure 13 we see our first two levels of explanation[6]. This is meant to be an overview of the results, and the initial explanation of the results. We can see the result of the entered evidence, which is that the most probable value for node *Stage* is IVA with a probability of 0.901. We explain this conclusion by showing the user the set of intermediate nodes and their most probable values. We can see that node *Location* and node *Length* are very likely to have these values and for node *Shape* we know his value for certain. This is a short explanation based on the results of Bayesian network itself, backed by the values of the intermediate nodes.

When the user is not yet satisfied by the results he is presented, we can go one step further and come to a third level of explanation. To get more insight in the value of an intermediate node we can expand the intermediate nodes. Expanding the intermediate node *Location* gives us the explanation as shown in Figure 14. We can notice our five earlier found clusters as arguments with the same ordering as we have presented them in Chapter 6. All nodes and values are boldfaced to emphasize them and make it easier to scan through them. In the right corner we can see the Venn-diagram that is associated with our argumentation. The numbers in the Venn-diagram represent the arguments given to the user on the left hand side and the letter L represents the node *Location*. In Chapter 6 we have seen that all clusters were completely disjoint and this is exactly what is visualized here in the Venn-diagram. We can see that all the arguments are based on their own evidence.

Expanding the intermediate node *Shape* gives us the explanation shown in Figure 15. This explanation is significantly larger than that of *Location* even though it contains one less argument. This is caused by the clusters containing more evidence nodes. In Chapter 6 we have seen that arguments were overlapping and some clusters were subsets of each other. We can see this structure reflected in the associated Venn-diagram. The evidence nodes of argument 4 are included in that of argument 2 and the evidence nodes of argument 3 are completely included in argument 1. We can also see that some of the evidence nodes of argument 4 are

[6]The level indication is not part of the actual explanation, but added here to illustrate the different levels

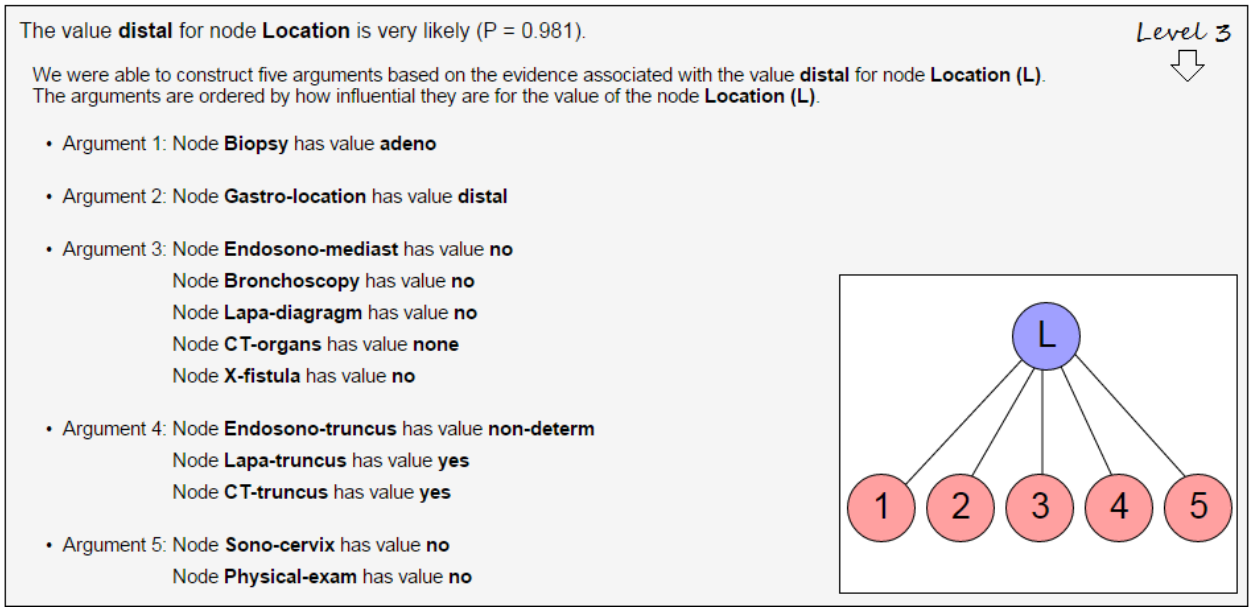

Figure 14: Third level of explanation for intermediate node Location

in argument 1. We can also see the overlap between argument 1 and argument 2.

The largest explanation is created by intermediate node *Length* and is divided over Figures 16 and 17. In this explanation we have the four clusters as arguments, were the first two arguments are the same as for the intermediate node *Shape*. Just as what we have seen with the previous two intermediate nodes, we have the arguments ordered by summed mutual information. These arguments are associated with a Venn-diagram to help the user understand how the arguments are connected to each other. In this Venn diagram we can again clearly see how the four arguments are related to each other. Here we can see that argument 1 is overlapping with both argument 2 and argument 3. Furthermore we can easily distinguish that argument 3 and argument 4 are subsets of argument 2. Finally we can also see that argument 4 is a subset of argument 3 as well.

7.4 Summary on the explanation of the reasoning

In this chapter we have seen how we combine the earlier found intermediate nodes and clusters to give the user an explanation of the reasoning of the network. We have presented the Oesophageal cancer network with evidence of an actual patient and constructed an explanation based on these results. For our explanation we used a combination of techniques in order to make the explanation more understandable for the reader, like accompanying probabilities with verbal expressions and translating the relations among clusters into Venn diagrams.

The found results have been discussed with S. Renooij, who was involved with the construction of the Oesophageal cancer network as mentioned earlier and this gave the following conclusion. The explanation of the most probable value of the intermediate node *Location* was good given the first two arguments. The first argument showed an evidence node, *Biopsy*, with a strong correlation to the location of the cancer and the second argument showed an evidence node, *Gastro-location*, that directly measures the location. So these

arguments showed to be a good explanation for the expert.

The explanation of the other two intermediate nodes showed very large clusters as the strongest arguments, which was caused by the property that subsets of clusters always have a lower mutual information for the associated intermediate node. The fact that we chose a patient that has values for all of the evidence nodes, made it so that we could not prune the graph and therefore arguments tended to be large. We used this case because it is the worst-case scenario. Normally a patient would not have as much values for the evidence nodes and therefore the network can be pruned and this would result in a simpler explanation.

For the explanation of the intermediate node *Shape* we saw that the third argument actually should have been the best argument, because this was a direct test for the most probable value of the associated intermediate node. We knew the value of the node *Gastro-Shape* and knowing this value gives us certainty about the outcome of *Shape*. In this case we argue that we only need argument 3 and our method should be extended. If we detected that knowing *Gastro-Shape* gave us certainty oabout *Shape* we could have simplified our explanation.

The explanation for the third intermediate node *Length* gave three large arguments, where two of them were the same. The two arguments were created from different clusters, but we can argue that we could have simplified our explanation by not showing the same argument twice. These argument were the strongest given the domain knowledge. The fourth and last argument were the arguments that said the least about the most probable value of the associated intermediate node.

In Chapter 4 we showed that we wanted to create an explanation that is based on these design choices:

- Content
  - Focus: *Reasoning*
    We find particular reasons for the most probable value of the node of interest.
  - Purpose: *Comprehension*
    We will gain more understanding about the reason the nodes have particular values.
  - Level: *Macro level*
    We focus on finding a set of nodes in order to explain the global reasoning of the network.
  - Causality: *Causal*
    As input for our method we use a causal Bayesian network.
- Communication
  - User-system interaction: *Natural language*
    We present an explanation of the network as a readable document.
  - Presentation: *Textual, graphical and multimedia*
    We present the explanation as a web page, with both a verbal and a graphical explanation.
  - Expression of probabilities: *Both numeric and linguistic*
    We make probabilities more understandable by accompanying them with verbal expressions.
- Adaption
  - Knowledge of domain: *Dynamic model*
    The user should have some background of the domain.
  - Knowledge of reasoning: *Dynamic model*
    The user should have some background of the reasoning.
  - Level of detail: *Dynamic*
    The user is able to look at several levels of explanation.

For the content of the explanation we focused on the reasoning, where we tried to give the user more comprehension on a macro level. We have based our explanation on causal reasoning. For the communication we chose to make an interactive web page, where the user has influence on what level of detail he wants to see the results. We tried to make the results more readable by accompanying them with verbal expressions. For an user to understand the result it is still important that he has some background of the domain and the reasoning. The combination of all these properties lead to the explanation as shown in this chapter.

The value **scirrheus** of node **Shape** is certain (P = 1.00).

We were able to construct four arguments based on the evidence associated with the value **scirrheus** for node **Shape (S)**.
The arguments are ordered by how influential they are for the value of the node **Shape (S)**

- Argument 1: Node **Endosono-mediast** has value **no**
  Node **Bronchoscopy** has value **no**
  Node **Lapa-diagragm** has value **no**
  Node **CT-organs** has value **none**
  Node **X-fistula** has value **no**
  Node **Gastro-necrosis** has value **no**
  Node **Sono-cervix** has value **no**
  Node **Physical-exam** has value **no**
  Node **CT-loco** has value **yes**
  Node **Endosono-loco** has value **yes**
  Node **Endosono-truncus** has value **non-determ**
  Node **Lapa-truncus** has value **yes**
  Node **CT-truncus** has value **yes**
  Node **Lapa-liver** has value **no**
  Node **CT-liver** has value **no**
  Node **X-lungs** has value **no**
  Node **CT-lungs** has value **no**
  Node **Endosono-wall** has value **T3**

- Argument 2: Node **Gastro-shape** has value **scirrheus**
  Node **Gastro-circumf** has value **circulair**
  Node **Gastro-length** has value **5 <= x < 10**
  Node **Weightloss** has value **x<10%**
  Node **Endosono-wall** has value **T3**
  Node **Endosono-truncus** has value **non-determ**
  Node **Endosono-loco** has value **yes**
  Node **Gastro-necrosis** has value **no**
  Node **X-fistula** has value **no**
  Node **Endosono-mediast** has value **no**
  Node **Gastro-location** has value **distal**

- Argument 3: Node **Gastro-shape** has value **scirrheus**

- Argument 4: Node **X-fistula** has value **no**
  Node **Gastro-necrosis** has value **no**

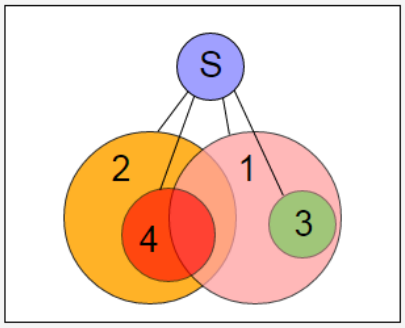

Figure 15: Third level of explanation for intermediate node Shape

The value **5 <= x < 10** of node **Length** is very likely (P = 0.984).

We were able to construct four arguments based on the evidence associated with the value **5 <= x < 10** for node **Length (L)**.
The arguments are ordered by how influential they are for the value of the node **Length (L)**

- Argument 1: Node **Endosono-mediast** has value **no**
  Node **Bronchoscopy** has value **no**
  Node **Lapa-diagragm** has value **no**
  Node **CT-organs** has value **none**
  Node **X-fistula** has value **no**
  Node **Gastro-necrosis** has value **no**
  Node **Sono-cervix** has value **no**
  Node **Physical-exam** has value **no**
  Node **CT-loco** has value **yes**
  Node **Endosono-loco** has value **yes**
  Node **Endosono-truncus** has value **non-determ**
  Node **Lapa-truncus** has value **yes**
  Node **CT-truncus** has value **yes**
  Node **Lapa-liver** has value **no**
  Node **CT-liver** has value **no**
  Node **X-lungs** has value **no**
  Node **CT-lungs** has value **no**
  Node **Endosono-wall** has value **T3**

Figure 16: Third level of explanation for intermediate node Length (Part 1)

Node **Gastro-necrosis** has value **no**
Node **Sono-cervix** has value **no**
Node **Physical-exam** has value **no**
Node **CT-loco** has value **yes**
Node **Endosono-loco** has value **yes**
Node **Endosono-truncus** has value **non-determ**
Node **Lapa-truncus** has value **yes**
Node **CT-truncus** has value **yes**
Node **Lapa-liver** has value **no**
Node **CT-liver** has value **no**
Node **X-lungs** has value **no**
Node **CT-lungs** has value **no**
Node **Endosono-wall** has value **T3**

- Argument 2: Node **Gastro-shape** has value **scirrheus**
  Node **Gastro-circumf** has value **circulair**
  Node **Gastro-length** has value **5 <= x < 10**
  Node **Weightloss** has value **x<10%**
  Node **Endosono-wall** has value **T3**
  Node **Endosono-truncus** has value **non-determ**
  Node **Endosono-loco** has value **yes**
  Node **Gastro-necrosis** has value **no**
  Node **X-fistula** has value **no**
  Node **Endosono-mediast** has value **no**
  Node **Gastro-location** has value **distal**

- Argument 3: Node **Gastro-shape** has value **scirrheus**
  Node **Gastro-circumf** has value **circulair**
  Node **Gastro-length** has value **5 <= x < 10**
  Node **Weightloss** has value **x<10%**
  Node **Endosono-wall** has value **T3**
  Node **Endosono-truncus** has value **non-determ**
  Node **Endosono-loco** has value **yes**
  Node **Gastro-necrosis** has value **no**
  Node **X-fistula** has value **no**
  Node **Endosono-mediast** has value **no**
  Node **Gastro-location** has value **distal**

- Argument 4: Node **Gastro-circumf** has value **circulair**

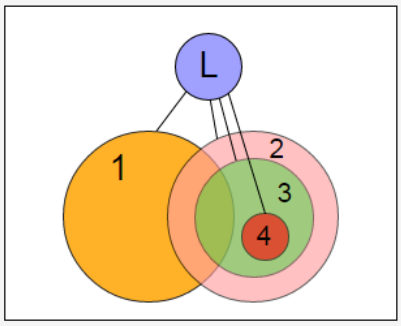

Figure 17: Third level of explanation for intermediate node Length (Part 2)

8 Summary and Conclusion

In this thesis we have presented a method that is able to create an explanation of a Bayesian network mainly based on two parts, namely the finding of intermediate nodes and the generation of clusters for these intermediate nodes. The results of these methods were combined in an explanation, which is an interactive web page where a user is able to determine the level of explanation he wants to see. In this chapter we are going over each of the methods in order to draw an overall conclusion of our method.

The first part finds the set of intermediate nodes with the Maximum-flow-minimum-cut theorem and the associated Edmonds-Karp algorithm. This set of nodes is used to explain the most probable value of the node of interest. This idea was based on finding a funnel in the graph of the Bayesian network that summarizes the reasoning. If we could find the smallest set of edges to make the graph disconnected, we would find this funnel. We presented a way to transform the graph of a Bayesian network into a flow graph, so that we were able to apply Edmonds-Karp to this graph. The minimum cut is found based on saturated edges of the flow graph, where each edge has its own weight. We presented three different weight-assignment functions, where One-for-All purely looks a the structure of the graph and the other two also take probability distributions into account.

In order to find out which of these weight-assignment functions was the best to use, we did several experiments. We investigated if the found sets of intermediate nodes had similar nodes or were even completely equal to each other. We also looked at the size of the found sets and the duration of the algorithms. Given the experiments on randomly generated Bayesian networks we were able to determine that in most of the cases the results of the weight-assignment functions was fairly the same. In almost every experiment there was overlap amongst the nodes and we concluded that One-for-All was the best method to use as a heuristic. This was due to the similarity of the found sets and the fact that One-for-All is fairly easy to calculate, because it does not take the probability distribution of the Bayesian network into account. We also applied the three methods to an actual Bayesian network, the Oesophageal cancer network, and all three methods gave back the same results. The only difference amongst the methods was the running time of the algorithms and then One-for-All is quickest.

With the second part we presented a method that provides the basis for explaining the most probable values of the intermediate nodes. We generated clusters that included at least one evidence node, which should summarize the information captured in the Bayesian network that can explain the most probable value of the intermediate node. In order to find out which cluster had the most influence on the associated intermediate node, we wanted to use mutual information; a measure to determine how much information one set of variables (the cluster) has about another set of variables (intermediate node). If we compare the mutual information of the intermediate node with each of the clusters we can determine which cluster is best to explain first. Unfortunately the amount of memory it takes to calculate mutual information for clusters was extremely high. So we were forced to come up with an alternative, which was the summed mutual information. We applied this method on the Oesophageal cancer network and we found four or five sets of clusters for each of the intermediate nodes. We translated these clusters into arguments based on the evidence in the cluster.

The explanation of the most probable value of the intermediate node *Location* was good given the first two arguments. The first arguments showed an evidence node with a strong correlation to the location of the cancer and the second argument showed an evidence node that directly measures the location. The explanation of the other two intermediate nodes showed very large clusters as the strongest arguments, which was caused by the property that subsets of clusters always have a lower mutual information for the associated intermediate node. The fact that we chose a patient that has values for all of the evidence nodes, made it so that we could not prune the graph and therefore arguments tended to grow large. We used this case because it is the worst-case scenario. Normally a patient would not have as much values for the evidence nodes and therefore the network can be pruned and this would result in a simpler explanation. For the

explanation of the intermediate node *Shape* we saw that the third argument actually should be the strongest argument, because this was a direct test for the most probable value of the associated intermediate node. The explanation for the third intermediate node *Length* had three large arguments, where two of them were the same. The fourth and last argument was the argument that said the least about the most probable value of the associated intermediate node.

We translated the input of these two parts into a explanation, which is an interactive web page. Here the user is able to differentiate between three levels of explanation, where the third is the most detailed. We associated the actual found posterior probabilities with verbal expressions and visually showed the relation among the cluster by a Venn-diagram of the clusters for an associated intermediate node.

9 Discussion and Future Research

In this thesis we have presented a method that consists of several parts and is able to create an explanation of the reasoning of a Bayesian network. The final explanation showed us promising results, but it showed enough space for improvement as well. When we focus on the results of Chapter 7 we saw that the explanation of the node *Location* was very good, but the explanation of *Shape* and *Length* could be improved. It showed that we might want to extend our method by detecting if the presence of certain evidence causes us to have certain knowledge about the value of the associated intermediate node. It also showed that it is possible that we can create the same argument several times. Here we can argue that we might want to show just one of them. We could counter this argument by looking at the graphical representation in the Venn diagram, which shows that the way the arguments were constructed was different. Although we have two arguments that consist of the same evidence nodes the structures of the two clusters can be different.

During this thesis project we have done experiments on lots of randomly generated Bayesian network, but only on one real example, the Oesophageal cancer network. Therefore it is hard to say how this method performs on other Bayesian networks, but it is important to note that this method is just a heuristic. In Chapter 7 we have also used a patient with values for each of the evidence nodes. This is in fact the worst case, because there are no nodes to prune, which leads to clusters being very large.

When we take a look at the entire explanation, it consists of several methods and is built in a modular fashion. This means it is easy to interchange methods for completely different methods. In general our explanation is constructed by finding sets of intermediate nodes and creating clusters that form arguments. Finding intermediate nodes is now done by he Maximum-Flow-Minimum-Cut theorem, but there might be other possibilities, like for instance the theory behind articulation points or maybe even domain knowledge. But even if we do not change our method of finding intermediate nodes, we can come up with new weight-assignment functions, which could result in different intermediate nodes and therefore a completely different explanation. On the other hand we can construct clusters in a different fashion and not just take all descendants of a direct child of the intermediate node. Of course the calculation of the mutual information of a cluster showed to be a challenge on itself due to the complexity of the calculation. We have come up with a different method that uses mutual information, but it definitely showed different results in a significant part of the example cases. In further research we could investigated if there are better approximations for the calculation of the mutual information.

Further research can also be done on the explanation itself. The proposed interactive web page is just one of many ways to represent the found information to the user. In further research we can find out if other ways of presenting the information is more clear for an user We now chose a Venn diagram to help us explain the argument, but another graphical representation of a Bayesian network might be more clear or more detailed.

So our method for finding an explanation for the reasoning of a Bayesian network can be changed at several levels, were we can tweak the weights of the edges, but we can also decide to find the set of intermediate nodes in a completely different fashion. So for further research we can expand this current method by adjusting these methods or even try to use completely replace methods by new ones. This can result a better heuristic for finding intermediate nodes or creating clusters which could lead to a better explanation of the reasoning of a Bayesian network.

References

[1] M. Baker and T. E. Boult. Pruning Bayesian networks for efficient computation. *UAI '90 Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*, pages 225–232, 2013.

[2] E. Charniak and S. E. Shimony. Cost-based abduction and MAP explanation. *Artificial Intelligence*, 66(2):345–374, 1994.

[3] G. F. Cooper. The computational complexity of probabilistic inference using bayesian belief networks. *Artificial intelligence*, 42(2):393–405, 1990.

[4] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*, volume 2. MIT Press, Cambridge, 2001.

[5] T. M. Cover and J. A. Thomas. Entropy, relative entropy and mutual information. *Elements of Information Theory*, pages 12–49, 1991.

[6] K. M. D. Madigan and R. Almond. Graphical explanations in belief networks. *Journal of Computational and Graphic Statistics*, 6:160–181, 1997.

[7] F. Díez. *Sistema Experto Bayesiano para Ecocardiografía.* PhD thesis, Departamento Informática y Automática, UNED, Madrid, 1994.

[8] M. Druzdzel. *Probabilistic Reasoning in Decision Support Systems: From Computation to Common Sense.* PhD. Thesis, Carnegie Mellon University, 1993.

[9] M. Druzdzel. Qualitative verbal explanations in Bayesian belief networks. *Artificial Intelligence and Simulation of Behaviour Quarterly*, 94:43–54, 1996.

[10] M. Druzdzel and M. Henrion. Using scenarios to explain probabilistic inference. *Working Notes of the AAAI-90 Workshop on Explanation*, pages 133–141, 1990.

[11] M. Druzdzel and M. Henrion. Efficient reasoning in qualitative probabilistic networks. *Proceedings of the 11th Conference on Artificial Intelligence*, pages 548–553, 1993.

[12] M. Druzdzel and M. Henrion. Qualitative reasoning and decision technologies. *Working Notes of the AAAI-90 Workshop on Explanation*, pages 451–460, 1993. Citeseer.

[13] C. Elsaesser. Verbal expressions for probabilty updates. How much more probable is much more probable. *Uncertainty in Artificial Intelligence*, 5:387–400, 1990.

[14] J. Gamez. *Inferencia Abductiva en Redes Causales.* PhD thesis, University of Granada, 1998.

[15] GeNIe. https://dslpitt.org/genie/.

[16] M. Henrion. Some practical issues in constructing belief networks. *Uncertainty in Artificial Intelligence*, 3:167–173, 1989. Seattle, WA, USA.

[17] M. Henrion and M. Druzdzel. Qualitative propagation and scenario-based approaches to explantions of probabilistic reasoning. *Proceedings of the 6th conference on Uncertainty in Artificial Intelligence*, pages 17–32, 1990.

[18] HUGIN. http://www.hugin.com/.

[19] J. Koiter. *Visualizing Inference in Bayesian Networks.* MSc Thesis, Delft University, 2006.

[20] C. Lacave and F. J. Díez. A review of explanation methods for Bayesian networks. *The Knowledge Engineering Review*, 17(02):107–127, 2002.

[21] J. Oosterhoff and W. R. van Zwet. *A Note on Contiguity and Hellinger Distance*. Springer, 2012.

[22] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publisher, 1988.

[23] J. Pearl. *Causality: Models, Reasoning and Inference*. Cambridge University Press, 2000.

[24] N. Pennington and R. Hastie. Explanation-based decision making: Effects of memory structure on judgement. *Journal of Experimental Psychology: Learning, Memory and Cognition*, 2:521–533, 1993.

[25] E. Santos Jr. On the generation of alternative explanations with implications for belief revision. *Proceedings of the Seventh conference on Uncertainty in Artificial Intelligence*, pages 339–347, 1991. Morgan Kaufmann Publishers Inc.

[26] P. Sember and I. Zukerman. Strategies for generating micro explanantions for Bayesian belief networks. *Elsevier*, pages 295–305, 1990.

[27] P. Sember and I. Zukermann. Strategies for generating micro explanations for Bayesian belief networks. *Uncertainty in Artificial Intelligence*, 5:295–305, 1990. Elsevier.

[28] S. E. Shimony. *A probabilistic framework for explanation*. Brown University, Department of Computer Science, 1991.

[29] E. Shortliffe. *Computer-based Medical consultations: MYCIN*. Elsevier, 2012.

[30] H. Suermondt. *Explanation in Bayesian Belief Networks*. PhD. Thesis, Stanford University, 1992.

[31] H. Suermondt and G. Cooper. An evaluation of explanations of probabilistic inference. *Computers and Biomedical Research*, 26:242–254, 1993.

[32] L. van der Gaag and J. Meyer. Informational independence: models and normal forms. *International Journal of Intelligent Systems*, 13:83–109, 1998.

[33] L. van der Gaag and S. Renooij. Probabilistic reasoning. *Lecture Notes*, University Utrecht, 2013.

[34] L. van der Gaag, S. Renooij, C. Witteman, B. Aleman, and B. Taal. Probabilities for a Probabilistic Network: A Case-study in Oesophageal Cancer. *Artificial Intelligence in Medicine*, 25:123–148, 200.

[35] M. Wellman. Fundamental concepts of qualitative probabilistic networks. *Artificial Intelligence*, 44:257–303, 1990.

[36] M. Wellman. Graphical inference in qualitative probabilistic networks. *Networks*, 20:687–701, 1990.