



University of British Columbia
Vancouver, Canada



University of Applied Sciences
Offenburg, Germany

A Method for Assessing the Trustworthiness of an Entity by Cooperating Authorities

Master Thesis

Communication and Media Engineering

Period: March 2005 - August 2005

Mathias Kohler, 165456

Wiesenstrasse 61-1
79312 Emmendingen
Germany

Supervisors:

Prof. Dr. Konstantin Beznosov
beznosov@ece.ubc.ca

Dept. of Electrical & Computer Engineering
University of British Columbia
Vancouver, BC, Canada V6T 1Z4



Prof. Dr.-Ing. Daniel Fischer
daniel.fischer@fh-offenburg.de

Dept. of Electrical and Computer Engineering
University of Applied Sciences
77652 Offenburg, Germany



Abstract

In this thesis a Model of Trust based on Bayesian Networks is introduced. The model determines the trustworthiness of a new or hitherto unknown entity by utilizing information exchanged with cooperative authorities.

Results from different trust models using individual evaluation techniques to assess the trustworthiness of an entity can only be utilized using specific conversion methods or not at all. However, utilizing results from other authorities is always needed if an authority cannot evaluate an entity on its own. This would be the case, for instance, if no previous interactions with an entity have taken place and hence no experience is available for an evaluation of an entity's trustworthiness.

This trust model addresses this problem and introduces a method to enable information exchange with cooperative authorities to assess a new or unknown entity, even if they use different evaluation techniques.

The method is based on the exchange of trust relations. This means, rather than using absolute values reflecting the degree of confidence in an entity, this model makes use of a relative notion of trust. An entity is hereby seen as being equally or more trustworthy than another entity.

The trust relations are modelled using the structure and inference algorithms of Bayesian Networks. By finally evaluating them, a new or unknown entity's trustworthiness is assessed.

Declaration of Authorship

I declare in lieu of an oath that the Master Thesis submitted has been produced by me without illegal help from other persons. I state that all passages which have been taken out of publications of all means or unpublished material either whole or in part, in words or ideas, have been marked as quotations in the relevant passage. I also confirm that the quotes included show the extent of the original quotes and are marked as such. I know that a false declaration will have legal consequences.

Vancouver, 31st August 2005

Mathias Kohler

Acknowledgements

I am particularly thankful to my supervisor, Prof. Dr. Konstantin Beznosov, for his continuous support throughout this project and for his insightful discussions and valuable advice.

My special thanks also go to Prof. Dr.-Ing. Daniel Fischer for his supervision of this project on the part of the Hochschule Offenburg.

My appreciation also goes to my fellow members of the LERSSE Group for our active discussions and in this way contributed opinions and suggestions.

Last but not least, I would like to thank my parents for their confidence in my abilities and their constant support throughout this endeavour.

Abbreviations

BG	Backward Graph
BN	Bayesian Network
CPT	Conditional Probability Table
FG	Forward Graph
ntw	not trustworthy
tw	trustworthy

Table of Contents

Abstract.....	I
Declaration of Authorship.....	II
Acknowledgements.....	III
Abbreviations	IV
Table of Contents.....	V
1 Introduction	1
2 Motivation	2
3 Related Work.....	5
3.1 Trust.....	5
3.2 Trust and Reputation Models.....	5
3.3 Bayesian Networks.....	6
4 Model Abstraction and Generalization	8
5 Goals and Approach.....	11
6 The Authorities' Relation Graph.....	13
6.1 How the Initial Graph is built	13
6.2 Notion of Distance	14
7 Merging Graphs	16
7.1 Merging Process.....	16
7.2 Handling Cycles	19
7.2.1 Collapsing cyclic parts.....	19
7.2.2 Deleting edge(s)	21
7.2.3 Deleting node(s)	22
7.3 Handling Graph Recommendations	22
7.3.1 Merging Multiple Graphs	22
7.3.2 Giving Graph Recommendations	23
8 Developing the structure of Bayesian Networks	24
8.1 An Introduction in Bayesian Networks	24
8.2 A Bayesian Network for Trust Evaluation	26
8.2.1 Backward Graph.....	27
8.2.2 Forward Graph	30
8.3 Dynamic Conditional Probability Values for the Bayesian Networks.....	33

8.3.1	Filling the CPTs dynamically.....	33
8.3.2	CPTs for nodes with multiple parental nodes	36
9	Determination of the Total Order	45
10	Application Example.....	50
11	Review and Future Work.....	54
12	Conclusions.....	57
13	References	58

1 Introduction

In this thesis a Model of Trust based on Bayesian Networks is introduced. The model determines the trustworthiness of an entity by utilizing information exchanged by cooperating authorities.

In open distributed computer systems trust is an issue which is demanding more attention than ever. Especially with the expansion of the internet and the ease of setting up temporal ad-hoc networks, it is critical to know which entities in a network are trustworthy and which entities are less or not trustworthy.

To evaluate the degree of confidence in an entity a lot of research has been done in recent years and many different trust models have been developed. Every model introduces its own method for judging an entity according to its capability for certain tasks or even to assess its trustworthiness in general.

With the increasing variety of trust evaluation methods comes also a greater variety of different evaluation techniques. In consequence, the value reflecting the degree of confidence in a certain entity might vary from trust model to trust model. Some of them use probabilities between 0.0 and 1.0 to express the belief in an entity. Others use integer numbers or just separate with categorizations such as "very untrustworthy", "untrustworthy", "trustworthy", and "very trustworthy."

In situations where an assessment for a certain entity from other authorities is needed, it is difficult to utilize results from different trust evaluation methods. Either specific formulas for conversion are needed or it is not possible at all.

The goal of this work is to provide a trust model to address this problem. It enables authorities to utilize recommendations from cooperative authorities even if they use different trust evaluation techniques.

The approach of this work is using relative trust relations. Rather than handling absolute values reflecting an absolute belief in an entity, this work models trust as a relative notion. It reflects the belief in an entity relative to other entities. In particular, an entity is stated to be equally or more trustworthy relative to one or more other entities.

The trust relations are modelled with the use of Bayesian Networks, a widely used knowledge based system. The structure, a directed acyclic graph, and all initial values are developed within this work. The functionality of Bayesian Networks, which are basically the inference algorithms, provides the method to compute the trustworthiness of a given entity.

The remainder of this work is organized as follows. In chapter 2 the motivation of this thesis is described on an introductory example. Chapter 3 deals with related work. Chapter 4 and 5 define the modelled system in greater detail and specify the goal and approach of this work. Chapter 6 introduces the relation graph this model is build on, followed by Chapter 7, 8 and 9 which describe step by step the approach of this work. Chapter 10 is reserved to exemplify the models application. Chapter 11 gives a critical review about the work and shows what future work might be possible to extend the model. Finally, chapter 12 concludes this work.

2 Motivation

The motivation will be explained in the following chapter using a simple example for a person Carol.

Carol might believe some people with regards to what they say about security engineering more than other people she knows. One of her friends could be an expert in that subject and she would believe this person (referred to as F) on what he says about security engineering. However, there might be another security expert (G) she might believe even more than her friend, F.

To illustrate this example, let's represent the set of people Carol knows within a circle as shown in Figure 1. Further, let's represent her confidence in F and G as a relation graph. In this case, it is $F \rightarrow G$, meaning that Carol trusts G about Security Engineering at least as much as F. The graph is also shown in Figure 1.

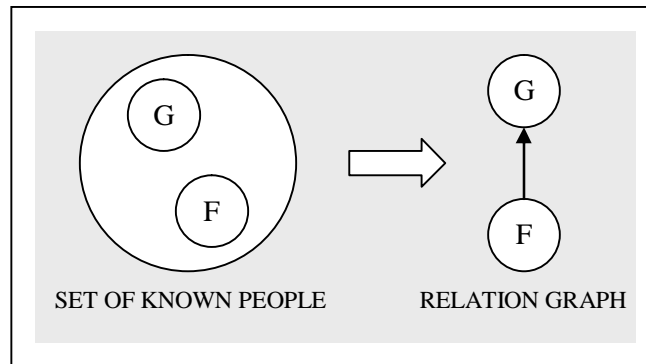


Figure 1: Considered the people Carol knows, she trusts person G at least as much as person F.

Assume furthermore at a security conference will be a speaker (referred to as S) whom Carol does not know yet. She is going to attend the security conference and wonders whether she should listen to the speech of S. Besides the topic, it is of interest for her how trustworthy the speaker and so his content might be. Since S is unknown to Carol it is not possible for her to evaluate how trustworthy the speaker is, nor is it possible to insert S in her relation graph (see Figure 2).

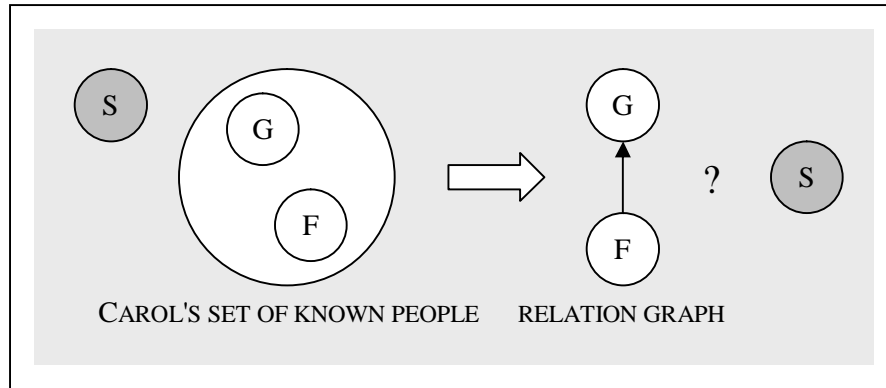


Figure 2: Person S is unknown to Carol. Hence it is not possible to insert S in her relation graph.

A colleague of Carol (referred to as A) knows the speaker and she could ask him how trustworthy what he thinks is S. But since the notion of trust is subjective, if her colleague tells Carol that S is 'trustworthy', it still remains unclear what 'trustworthy' means to her colleague compared to what it means to Carol herself.

Nevertheless, Carol's colleague might be able to build his own relation graph with the people he knows. Suppose he introduces Carol to his own view of specialists in security engineering. He also knows Carol's friend F and brings additionally one more expert (X) in the field of security into play. Assume the colleague's opinion on the expertise of these people is (in the ascending order of trust) X, F, S. The relationship graph is shown in Figure 3.

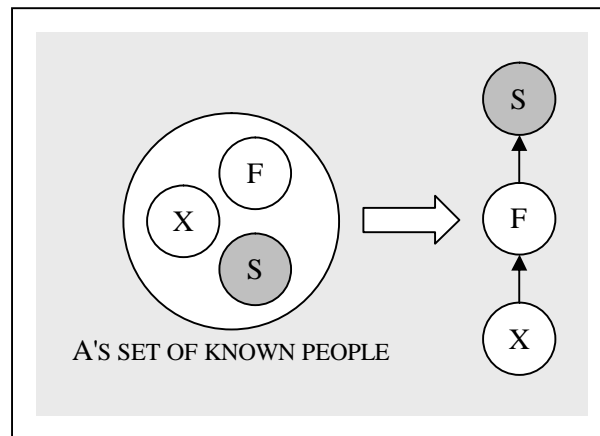


Figure 3: A's confidence in X, F, and S.

If Carol decides to take her colleague's list of security experts into account, then she might want to extend her own relation graph with the relations of her colleague's graph and "merge" somehow her own with her colleague's graphs.

In consequence, this step should make it possible to evaluate the trustworthiness of S compared to other people in the relation graph because their representative vertices will then be connected in the same graph.

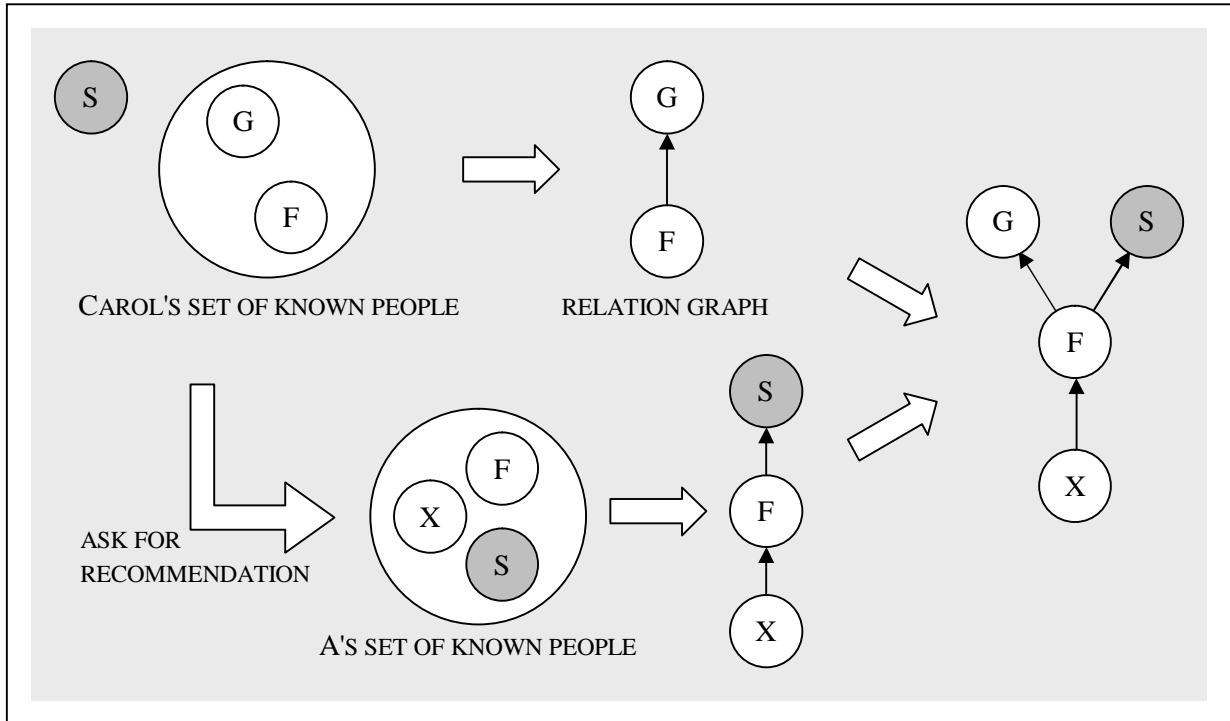


Figure 4: To insert speaker S in her relation graph, Carol asks A for recommendation and merges the two relation graphs.

At this point, Carol can already state S is more trustworthy than F and X since there is a direct and transitive connection between them. But, the question for Carol, illustrated in Figure 4, remains which of the two experts G or S she should trust more. It cannot be resolved at this stage since there is no direct or transitive relation between the two. The thesis will address this question later in this paper and develop a method to transfer such a relation graph into a graph of total order. This will provide then an assessment of how trustworthy speaker S is compared to all persons in the relation graph.

3 Related Work

This chapter represents work related to the model presented in this thesis. The first section covers the general aspects of trust, which plays a major role in this work. It lists the main points how other authors and researchers interpret the notion of trust.

The second section presents different trust and reputation models. It provides a general idea about recent developed models and gives the reader a more global picture about the context within which the model in this work is placed.

The third and final section in this chapter gives an account of related work on Bayesian Networks. It lists papers which provide a description for understanding Bayesian Networks and also papers which were used for this work.

3.1 Trust

In open distributed computer systems, trust receives more and more attention. Trust is often defined as a trusting party's belief in characteristics like reliability, honesty and competence of a trusted party [Wang, 2003], sometimes as subjective or even elusive notion [Abdul-Rahman, 2000]. Summarized, trust is seen as a degree of confidence that a specific entity acts or behaves in an explicit context in a certain way.

Several researchers characterize or use trust as context specific element [Yahalom, 1993], [Abdul-Rahman, 2000] or state that trust should always be based on knowledge or prior experiences [Jøsang, 1996].

In [Shi, 2004] and [McKnight, 1996] the authors expand this characterization and differentiate between three different types of trust: *Situational Trust*, *General Trust* and *Basic Trust*¹. The first type is the trust in another entity derived from all past experiences within a particular context or subject. The second and third types denote a context independent notion of trust and refer therefore to a broader view of belief in the expected behaviour of other agents.

In this work will focus mainly on the first type, the notion of trust based on experience and within a particular subject or context.

3.2 Trust and Reputation Models

A number of researchers deal with trust and reputation models. These models are usually built to help make decisions or to support the evaluation process of assessing the trustworthiness of different entities in distributed systems. In [Shi, 2004] a trust model is introduced for helping users and machines in decision-making based on prior experiences. In their model the possible outcome of the execution of an action is calculated with the aid of probability distribution.

¹ McKnight et al. [McKnight, 1996] use different terms to introduce these types of trust. In particular: *Interpersonal Trust*, *Impersonal Trust* and *Dispositional Trust*.

Abdul-Rahman and Hailes propose in [Abdul-Rahman, 2000] a Trust Model which assists agents to identify trustworthy agents. The trust evaluation in their model relies on prior experience with a certain agent within a specific context as well as on reputational information gathered in form of recommendations from other agents. In their model, the degree of trust in other agents is subjectively categorised into a set of *Trust Degrees*: "Very Trustworthy", "Trustworthy", "Untrustworthy" and "Very Untrustworthy." The proposed trust model deals also with the trustworthiness of recommender agents and introduces the notion of "semantic distance." It indicates the deviation between the agent's current opinion in another agent's trustworthiness and the recommender's proposal. The more the recommendation differs the less it influences the agent's opinion.

A model for trust decision and reputation management in multi-agent networks is suggested by Yu and Singh [Yu, 2002]. Their approach is based on the 'Dempster-Shafer Theory' which handles the notion of belief in a way that a 'lack of belief does not [necessarily] imply disbelief' [Yu, 2002]. They introduced the term *total belief* which describes the level of belief in another agent. It is calculated by combining the *local belief* (which refers to an agent's belief in another agent from direct interactions) and recommendations from other agents. By introducing *TrustNet*, Yu and Singh provide agents with the ability to combine and evaluate testimonies (recommendations) from several other agents.

In general, the evaluation process (whether an agent or entity is trustworthy) is done by analysing prior experiences and/or utilizing recommendations from others. It is important to consider the trustworthiness of the recommender entities as well as handling conflicting recommendations [Shi, 2004]. Also the aspect that trust has dynamic and non-monotonic characteristics ('additional information at a later time may increase or decrease [the] degree of trust in another [entity]' [Abdul-Rahman, 2000]) should be considered.

3.3 Bayesian Networks

Bayesian Networks are used in a variety of fields such as diagnosis [Heckerman, 1992], language understanding [Charniak, 1989], risk prediction and others. A good overview with references is given by Heckerman [Heckerman, 1995] in his paper.

Charniak [Charniak, 1991] introduces Bayesian Networks in a very basic manner. He demonstrates that Bayesian Networks can be explained without juggling with too many formulae. It is very helpful if the reader wants to get just enough knowledge about Bayesian Networks to be able to estimate whether they might be useful for his or her own work.

A more advanced introduction into Bayesian Networks is given by Morawski in his paper [Morawski, 1989]. He mainly focuses on the explanation of the evaluation method for Bayesian Networks introduced by Judea Pearl [Pearl, 1988].

Bayesian Networks play a major part in this work to finally evaluate the trustworthiness of an entity. Besides the basic structure of Bayesian Networks and their common inference algorithms, for example the Message Passing Algorithm (for tree-based structures) introduced by Pearl [Pearl, 1988] or the clustering algorithm by Lauritzen & Spiegelhalter [Lauritzen, 1990] for any type of directed acyclic graphs, this work makes use of the extended version of the noisy OR-gate model.

The original noisy OR-gate model was introduced in [Pearl, 1988] and it is widely used in Bayesian Networks. It deals with specifying the initialization values for the 'Conditional Probability Tables' (CPTs) in a Bayesian Network. Usually these values are determined by either analysing huge data sets with sample data or by eliciting an expert in a particular domain. The noisy OR-gate model introduced by Pearl offers an alternative to compute the conditional probabilities needed for the networks CPTs if only limited information is available.

Henrion built on Pearl's developed noisy OR-gate model and extended it by introducing in [Henrion, 1989] the *leaky* noisy OR-gate. Bayesian Networks model in an abstract way the relation between causes and their effects. The original noisy OR takes only the actually present causes into account, implying that if no cause is present no effect will occur. Henrion extended this by introducing a *leak probability*. It models the fact that an effect occurs for a certain probability even if no cause is present.

4 Model Abstraction and Generalization

In this work, trust is seen as the belief in another person's or system's trustworthiness in being better or more reliable than others according to a specific subject (e.g. providing better quality in carrying out certain instructions or offering higher reliability in executing transactions).

In general and abstract terms, the system consists of several elements which will be defined in the following paragraphs.

A trust relationship always involves two parties [Jøsang, 1996], the trusting party and the trusted party. First, the definition for the trusted party in the system which is called *entity* is given. The definition for the trusting party which is called *trust authority* will be introduced later in this section.

Definition 1. Entity: *An entity is a distinct object such that*

- (a) *one or more trusting parties have a certain degree of confidence in its abilities, capabilities, or reliabilities according to a specific subject, and*
- (b) *if $E = \{e_1, e_2, \dots\}$ is a set of entities then " $e_i, e_j : e_i = e_j$ implies $i = j$.*

In [Yahalom, 1993] the authors state that there are several tasks (trust classes) where an entity can be trusted in and 'it might be very reasonable to trust an [entity] with respect to some tasks and not necessarily with respect to others.' In other words, the direct confidence in a certain entity should be related to a specific subject or *context*. This aspect will be defined in the following definition.

Definition 2. Context: *A context is the subject which the trusting party's confidence in an entity is related to. Let $C = \{c_1, c_2, \dots\}$ be a set of contexts such that " $c_i, c_j : c_i = c_j$ implies $i = j$.*

So far an *entity* as trusted party and the *context* where the belief of a trusting party is relying on are defined. To draw a parallel between the example which is given in section 2 and the definitions above, the people in the example are the distinct entities, and the subject of Security Engineering is the context.

The next step is to define the link between two entities. It stands for the trusting party's (subjective) degree of confidence in those entities and expresses its belief which of those entities is more trustworthy compared to each other. When it was claimed in the example in section 2 that person D is more trustworthy than person F (on context C), the relation was informally denoted with an arrow ($F \rightarrow D$). Formally described, this relation is as follows.

Definition 3. Trust Relation: *The trust relation between two entities expresses the trusting authority's degree of confidence in these entities and indicates the authority's belief which of these entities is more trustworthy. It is denoted by $e_1 \leq_c^A e_2$, meaning e_2 is no less trustworthy than e_1 , whereby*

- (a) $e_1, e_2 \hat{I} E$ with $e_1 \neq e_2$,
- (b) $c \hat{I} C$, and
- (c) A being the trusting authority².

In later trust evaluations, the notion of distance between two entities will be introduced. It represents how trustworthy one entity is with respect to another. The value range is $[0.0, 1.0] \in \mathbb{R}$. If, for example, the distance between two entities is 0.0 the two entities are equally trustworthy. The greater the distance gets the more trustworthy is one entity relative to the other. The definition for the notion of distance is given in the next paragraph. Its calculation is mainly application dependent and will be discussed later in this work.

Definition 4. Distance: *Let $e_1, e_2 \hat{I} E$ and let $e_1 \leq_c^A e_2$. The distance between the two entities represents the relative distance between the degree of confidence in e_1 and the degree of confidence in e_2 such that*

- (a) *its range is $[0.0, 1.0] \hat{I} \mathbb{R}$.*

This paper uses relation graphs representing the relations between entities. The following definition describes the structure of such graphs.

Definition 5. Context-based Relation Graph: *Let $D = (E, R, c)$ be a directed acyclic graph with*

- (a) $E = \{e_1, e_2, \dots\}$ being the set of vertices representing the entities,
- (b) $R = \{r_1, r_2, \dots\}$ being the set of edges representing the trust relations
 $R \hat{I} \{E \times E \mid (e_m, e_n) \circ e_m \leq_c^A e_n \text{ with } m \neq n\}$,
- (c) $d(r_i)$ being the distance between two entities, and
- (d) $c \hat{I} C$.

Along with *entities*, *trust relations* and *context-based relation graphs*, the last element in the system is the trusting party called *trust authority*. Every authority develops and maintains its own relation graphs. By setting corresponding trust relations in a graph, an authority can express its degree of confidence not only between two entities but among all entities in the graph - always with respect to the context the graph is related to.

² see Definition 6

Definition 6. Trust Authority: A trust authority is the trusting party maintaining one or more relation graphs such that

- (a) $D = \{g_1, g_2, \dots\}$ is the set of graphs the authority maintains,
- (b) " $g_i, g_j : g_i.c = g_j.c$ implies $i = j$, and
- (c) $c \hat{I} C$.

The graphs considered in this paper are directed acyclic graphs in contrary to directed cyclic graphs. The reason is as follows: Assume a cyclic graph D with a set of vertices $V(D) = \{A, B, C\}$, a set of edges $E(D) = \{(A, B), (B, C), (C, A)\}$ and assigned distances to every relation (see Figure 5). From a mathematical point of view the graph definition is fine. However, the trust relation would be seen in the following manner:

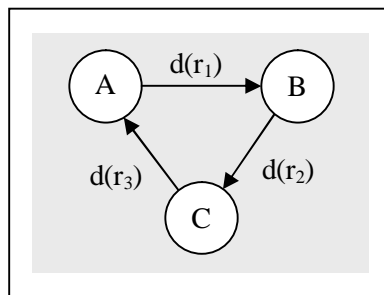


Figure 5: A Cyclic graph where A would be less trustworthy than B, B less than C, and C less than A.

There are edges from A to B and from B to C. Consequently the authority trusts B at least as much as A and it trusts C at least as much as B. Thus, it can also be stated: *A is equally or less trustworthy than C*. Still, there is a further edge from C to A representing *C is equally or less trustworthy than A*. This statement is obviously in contrary to the previous statement³ which illustrates the complication with cyclic parts in the relation graphs. In consequence acyclic graphs are used to represent relation graphs.

³ Except for the case that A, B and C are equally trustworthy.

5 Goals and Approach

In the last chapter the definitions which specify the framework of this thesis are introduced. In this chapter this framework will be used to specify the goals of this work and outline its approach.

The degree of trust in an entity is usually determined by analysing past experience with the entity. In a case where no previous interactions with an entity took place, this information is missing. In such a case most of the trust and reputation models introduce a method to interact with other authorities to exchange of information about the trustworthiness of the entity.

The problem the paper addresses is that this information exchange between authorities is only possible when the other authority uses the same or similar trust evaluation methods to determine the trustworthiness of the entity. This is because the value reflecting the degree of trust varies from trust model to trust model.

There exist many different possibilities to evaluate the degree of trust of an entity. They range from the use of very basic methods (e.g. calculating the ratio between the number of positive and the number of total interactions) to the use of highly sophisticated trust evaluation techniques. Some of them express the belief in an entity as probabilities. Others use integer numbers and yet others which distinguish between categorizations from "very untrustworthy" to "very trustworthy" as seen in the related work section.

In fact, to utilize evaluation results from other authorities using different evaluation techniques, either specific formulas for conversion are needed or it is not possible at all.

The goal of this work is to provide a trust model to assess the trustworthiness of a new or unknown entity. Hereby the model enables authorities to utilize recommendations from cooperating authorities independent of the trust model used by them.

The approach of this work is using relative trust relations. Rather than handling absolute values reflecting an absolute belief in an entity, this work models trust as a relative notion. It reflects the confidence in an entity relative to other entities. In particular, an entity is stated to be equally or more trustworthy relative to one or more other entities. All trust relations combined form a context-based relation graph, represented by a directed acyclic graph. The trustworthiness of an entity will then be determined by its integration in this relation graph.

The entire approach of this model is separated in three main parts.

The first part is the definition of an algorithm for utilizing trust relations from other authorities. This step is to integrate the new or unknown entities into the context-based relation graph of an authority.

The second part is to develop the structure and initialization values of the Bayesian network. This includes also the determination of the 'Conditional Probability Tables' which defines the probabilities used to evaluate a Bayesian Network.

The third part is to transfer the current partial ordered relation graph into a complete ordered graph by using the Bayesian network's structure developed in part 2. This finally provides the means to define completely the order of trust among all entities of a relation graph. Thus the trustworthiness of an entity to be assessed can be classified according to its position in the

total order. The total order is a hierarchical relation graph and the higher an entity's position in the graph, the more trustworthy is the entity in comparison to all other entities in the graph.

6 The Authorities' Relation Graph

In this chapter the authority's relation graph will be introduced. It is the main element in this work. The next section shows in detail how an authority's relation graph is built and on which values it is based. Further it explains aspects which are needed for the trust evaluation process (such as the notion of distance).

6.1 How the Initial Graph is built

This section describes how an authority builds an initial Context-based relation graph. It is assumed that every authority manages its own set of known entities. When an authority builds its *initial* relation graph, only these entities are taken into consideration. No recommendations from other authorities are considered at that point.

The starting point is that an authority has a set of known entities which have already been evaluated, meaning the degrees of trust in these entities are available. These degrees of trust should be expressed by a real number. It can be determined by using different trust models provided by other researchers or by using the very basic method of calculating the ratio between previously successful interactions (with that entity) divided by the total number of interactions. This gives the authority the freedom to choose an appropriate method for its own needs.

The authority might also have a set of entities which are new or unknown and therefore shall be evaluated.

The initial relation graph, which is a hierarchical ordered graph, is built using *only* the entities of the first set. Figure 6 illustrates this.

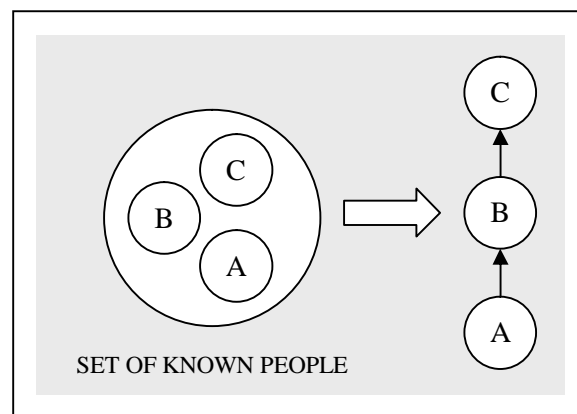


Figure 6: The known entities are used to build the initial relation graph.

The nodes of the graph represent the entities, the edges represent the relations. The set of nodes is given by the above-mentioned set of known entities. The relations are determined by

the following algorithm. It basically sorts the entities according to their trustworthiness and inserts the relations to form a hierarchical ordered graph.

ALGORITHM 1: BUILDINITIALRELATIONGRAPH(E)

Input: $E = \{e_1, e_2, \dots, e_n\}$ (set of evaluated entities)

Output: $G = (E, R)$ (directed graph)

```

1   sort(E, asc)           Ø Sorts E in ascending order according to value of ei
2   for i from 1..(n-1) do
3     R{ } ← edge(ei, ei+1)

```

Algorithm 1: Builds the initial relation graph

6.2 Notion of Distance

In this section the notion of distance is introduced. It qualifies every single relation previously inserted in the relation graph and is needed for the later trust evaluation.

Basically, the distance between two entities represents how much one entity is more trustworthy than another. The range of the distance's value is defined as $[0.0, 1.0] \in \mathbb{R}$. A minimal distance in a relation between two entities would consequently mean one entity is minimally more trustworthy than the other. In fact, if the distance is zero, the two entities would be equally trustworthy. The greater the distance gets, the more trustworthy is one entity relative to the other.

Relative is actually the key word in the last sentence. The distance is determined individually for every relation rather than in an absolute way. Its calculation uses the degrees of trust the authorities computed for every known entity. In other words, if the authority assigns, for instance, real values to express the degree of trust in an entity, these real values are used to calculate the relative distance.

The following formula can then be used to calculate the relative distance between two entities.

$$d_{mn} = 2 \cdot \left[\max \left(\frac{|e_m|}{|e_m| + |e_n|}, \frac{|e_n|}{|e_m| + |e_n|} \right) - 0.5 \right] \quad \text{for } n \neq m \quad (6.1)$$

In the formula, $|e_m|$ and $|e_n|$ are the values expressing the degree of trust an authority has in an entity.

At this point, two assumptions about the input values ($|e_m|$, $|e_n|$) are made.

1. If $|e_m| = |e_n|$ for $n \neq m$, the entities e_m and e_n are considered equally trustworthy.

2. The values are normalized that they scale linear to an entity's degree of trust. This means, for instance, if a value doubles, it reflects a double amount of trust in that entity.

To illustrate the distance calculation, assume an authority A manages a set of two entities. Be $S = \{e_1, e_2\}$ the set. Furthermore, assume A determines the belief in its entities using the ratio between an entity's successful transactions and total transactions. If, for instance, the ratio for entity $e_1 = 3/10$ and for entity $e_2 = 9/10$, the distance between the two entities would be

$$d_{12} = 2 \cdot \left[\max\left(\frac{3/10}{3/10+9/10}, \frac{9/10}{3/10+9/10}\right) - 0.5 \right] = 2 \cdot \left[\frac{3}{4} - 0.5 \right] = 1/2$$

The following algorithm calculates the distances for every given relation within a relation graph provided as input. Since generally the distance calculation itself is application dependent, the algorithm calls the function *CalcDistance* with two input values. In the above described situation, however, this function would simply return the value calculated by the formula (6.1) depicted previously.

ALGORITHM 2: CALCULATEDISTANCES(G)
Input: $G = (E, R)$ (Context-based relation graph)
Output: $G_{\text{out}} = (E, R)$ (weighted Context-based relation graph)

- 1 **for** each edge $r \in R$ **do**
- 2 $d(r) \leftarrow \text{CalcDistance}(|r.e_1|, |r.e_2|)$

Algorithm 2: Create Distances

A full initialized relation graph is the basis for the further approach of utilizing relation graphs from other authorities. The next chapters will go into the details of integrating recommendations from other authorities by merging two or more graphs. This is the first of three steps to evaluate the total order of a set of entities according to their trustworthiness.

7 Merging Graphs

Merging two or more graphs is one way an authority can integrate new and previously unknown entities into its relation graph. Relations between two entities which could previously not be resolved can be determined by uniting the authority's graph with that of another authority containing the relation of interest.

The following sections show how two relation graphs are merged. They also present techniques to handle cycles and assess them according to their advantage. The final section rounds the merging process off by addressing multiple merge processes.

7.1 Merging Process

The unification of two graphs itself is simple and straight forward. To express it in mathematical terms, two definitions are necessary: Let $G_1 = (E_1, R_1)$ be a directed graph where $E_1 = \{e_1, e_2, \mathbf{K}, e_p\}$ represents its finite set of vertices and $R_1 = \{(e_n, e_m), \mathbf{K}\}$ represents its finite set of edges. Let $G_2 = (E_2, R_2)$ be a second graph of the same type as G_1 . Formally expressed is the united graph $G_{1/2} = (E, R)$ the result of $E(G_{1/2}) = E_1 \cup E_2$ and $R(G_{1/2}) = R_1 \cup R_2$. Hence, the merge of two graphs ($G_{1/2} = G_1 \cup G_2$) is done by means of the unification of the sets of nodes and the sets of edges respectively. It unifies all (distinct) edges and nodes of two given graphs.

To draw an example, assume a directed graph D_1 is represented by its vertices $E(D_1) = \{A, B, C\}$ and edges $R(D_1) = \{(A, B), (A, C)\}$. A second directed graph D_2 is represented by $E(D_2) = \{B, C, D\}$ and $R(D_2) = \{(B, C), (C, D)\}$ respectively (see Figure 7).

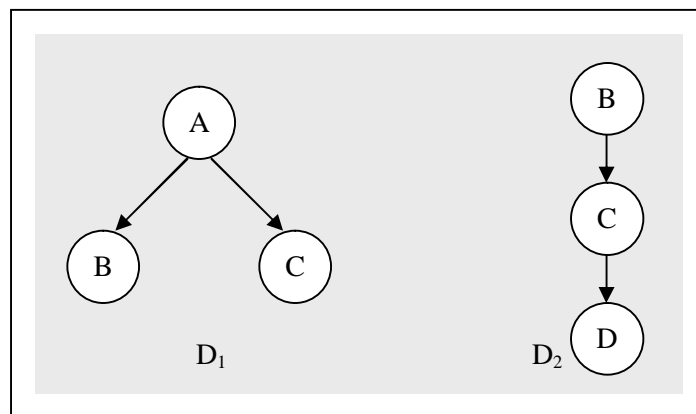


Figure 7: Two separate, independent graphs D_1 and D_2 .

According to the procedure described above, the result of the merge of the given two graphs is as follows. The vertices of the new graph $D_{1/2} = (E, R)$ are $E(D_{1/2}) = E(D_1) \cup E(D_2) = \{A, B, C, D\}$; the edges are $R(D_{1/2}) = R(D_1) \cup R(D_2) = \{(A, B), (A, C), (B, C), (C, D)\}$. The resulting graph is displayed in Figure 8.

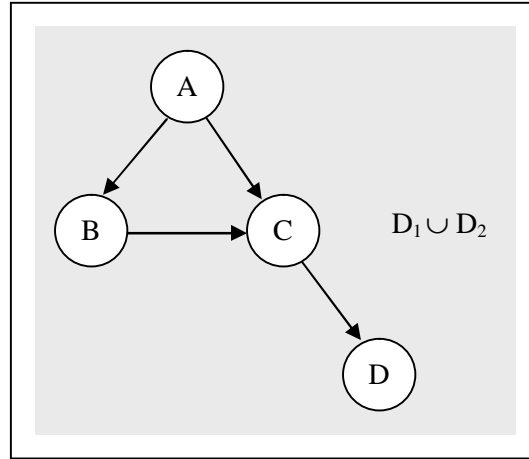


Figure 8: The result of the merging of the directed graphs D_1 and D_2 .

The distances assigned to all relations are considered as follows. In general the distance value assigned to a relation is inserted into the unified graph combined with the edge representing the relation.

Consider again D_1 and D_2 as defined above. D_1 has two relations and their specified distances⁴ are $d_1(A, B) = d_1(A, C) = 0.3$. D_2 has also two relations and their corresponding distances are $d_2(B, C) = d_2(C, D) = 0.5$. The representing unified graph is shown in Figure 9 (a).

When two relation graphs are unified it is possible that both graphs contain the same relation but different distances assigned to it. In such a case the final distance in the unified graph is the average distance of both source relations.

Consider D_{2b} as identical to D_2 except that it has an additional relation $(A, C) \in R(D_{2b})$ with an assigned distance of $d_{2b}(A, C) = 0.5$. The final distance of the relation $(A, C) \in D_{1/2b}$ is then $d_{1/2b}(A, C) = \text{avg}(d_1(A, C), d_{2b}(A, C)) = \text{avg}(0.3, 0.5) = 0.4$. The corresponding graph is shown in Figure 9 (b).

⁴ Distances are denoted as $d_x(r_m)$. The index x denotes the relation graph's D_x index. If, for example, the distance's assigned relation $r_m \in D_1$, the distance would be denoted as $d_1(r_m)$.

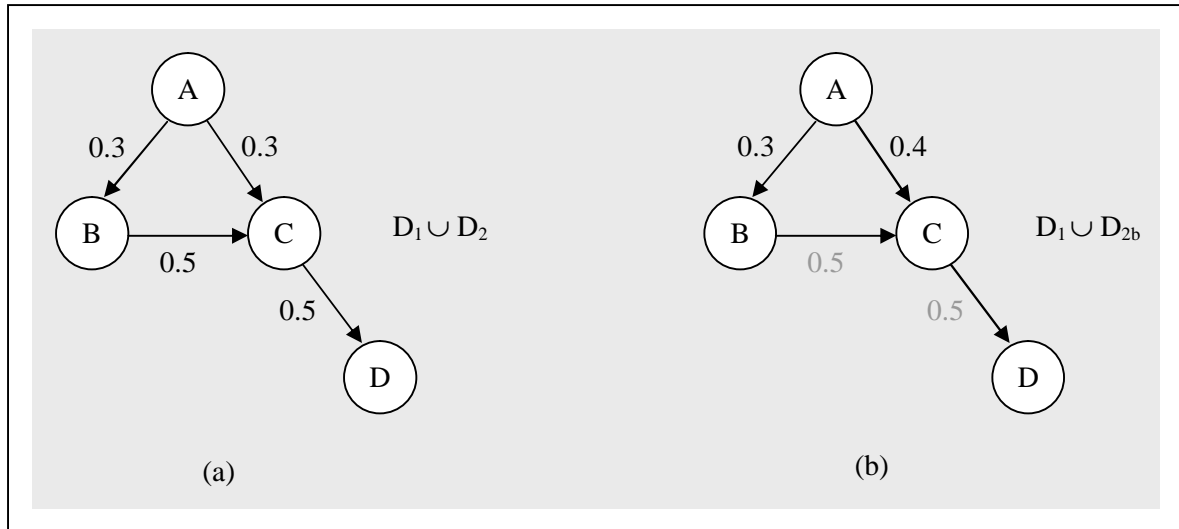


Figure 9: (a) Unified relation graph with assigned distances;
 (b) Unified relation graph with the graph D_1 and the modified graph D_{2b} as sources.

The corresponding algorithm for merging two directed graphs is described with Algorithm 3. It basically makes use of the first input graph as starting point and integrates all missing nodes and edges from the second input graph. The distances are handled accordingly.

ALGORITHM 3: MERGEGRAPHS(G_1, G_2)

Input: $G_1 = (E_1, R_1)$, $G_2 = (E_2, R_2)$ (two directed graphs)

Output: $G_3 = G_1 \cup G_2$

1	$G_3 \leftarrow \text{Copy}(G_1)$	\emptyset Make a copy of G_1 and save it as G_3
2	for each vertex $e \in E_2$	
3	do if $e \notin E_3$	
4	then $E_3[] \leftarrow e$	\emptyset add vertex e to the list of vertices E_3
5	for each edge $r_m \in R_2$	
6	do if $r_m \notin R_3$	
7	then $R_3[] \leftarrow r_m$	\emptyset add edge r_m to the list of edges R_3
8	$r_m.d \leftarrow d(r_m)$	\emptyset save the distance of edge r_m
9	else $r_m.d \leftarrow \text{avg}(d(r_m), d(r_n))$	\emptyset with $r_m \in R_2$ and $r_n \in R_3$

Algorithm 3: Merge Graphs

7.2 Handling Cycles

As previously stated, this paper uses directed acyclic graphs representing the trust relations between several entities instead of using cyclic graphs. This is because in presented notion of trust, the relations are transitive⁵ and therefore exclude cyclic relations. However, when merging two graphs using the procedure outlined above, cycles may appear in resultant graphs.

In the next paragraph a method is presented on how to deal with cyclic parts in graphs, followed by other considered alternatives. To illustrate the methods let $G = (E, R)$ be a directed sample graph with $E(G) = \{a, b, c, d, e, f, g, h\}$ and $R(G) = \{(a, b), (a, f), (b, c), (c, a), (c, d), (c, g), (d, e), (f, c), (f, e), (h, a), (h, c)\}$. The graph is shown in Figure 10.

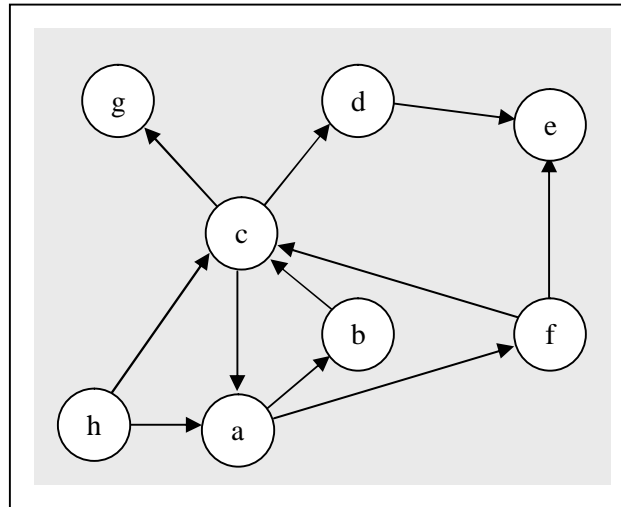


Figure 10: Graph $G = (E, R)$ with two cycles $\{a, b, c, a\}$ and $\{a, f, c, a\}$ forming a strongly connected component.

7.2.1 Collapsing cyclic parts

To deal with cycles actually means to transfer a cyclic graph into an acyclic graph, thus to dissolve the cyclic parts by modifying or replacing them properly. In the sample graph G the *strong component*⁶ $\{a, b, c, f\}$ is the cyclic part of the graph which therefore has to be dealt with.

⁵ If $e_1 \leq_c^A e_2$ and $e_2 \leq_c^A e_3$ is given, transitivity implies $e_1 \leq_c^A e_3$.

⁶ A *strongly connected component* [short: *strong component*] of a directed graph G is a maximal subgraph of G such that for every two distinct vertices in the subgraph, each vertex is reachable from the other. A graph can contain one or more strong components.

It is indeterminable which entity within a cycle is most trustworthy, since every one of them could be (as shown in section 4). Therefore, these entities are going to be treated as being equally trustworthy. This is done by collapsing cyclic parts to one node, hereby putting all the entities within a cycle in the same level of trustworthiness. In terms of graph theory, this modification is a transformation of a cyclic directed graph into an acyclic directed graph by building the *strongly connected component graph (SCCG)*.

To illustrate an example let $G = (E, R)$ be the original cyclic graph (Figure 10). To get the acyclic graph, the strong component $\{a, b, c, f\}$ is collapsed to one node 'abcf'. The resultant SCCG is $G' = (E', R')$ with $E'(G') = \{abcf, d, e, g, h\}$ and $R'(G') = \{(abcf, d), (abcf, e), (abcf, g), (d, e), (h, abcf)\}$ (shown in Figure 11).

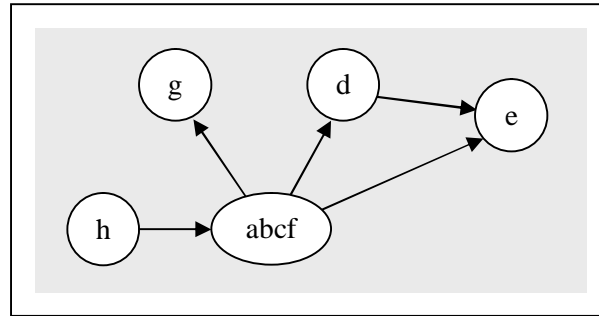


Figure 11: Strongly connected component graph of $G' = (E', R')$

If nodes outside of a strong component have multiple links to nodes inside a strong component, the links are combined to one *combined edge* pointing to the collapsed node. An example is node h in the sample graph G . This node has the edges (h, a) and (h, c) . Since the nodes a and c are collapsed in G' , h is linked to the collapsed node by the single connection $(h, abcf)$.

The distances when collapsing a graph are handled as follows. In a strong component the distances are reduced to zero. This corresponds to the assumption that all entities within a strong component are equally trustworthy. The distance for the combined edge is determined by averaging the distances from the connections to the former single nodes.

The algorithm for determining the strongly connected components in a given directed cyclic graph G is presented, for example, in [Cormen, 2001, p. 554] or [Manber, 1989, p. 231]. The following algorithm makes use of it and generates a directed acyclic graph which is used for further processing.

ALGORITHM 4: GENERATESCCG(G)*Input:* $G = \{E, R\}$ (cyclic directed graph)*Output:* $G' = \{E', R'\}$ (acyclic directed graph)

```

1   $G' \leftarrow$  create empty graph
2   $SC[ ] \leftarrow$  call StronglyConnectedComponents(G)  $\emptyset SC[ ] =$  list of strong components
3  for each  $sc \in SC$ 
4      create composite vertices  $cv$ 
5   $E'[ ] \leftarrow$  all vertices  $v \in E$  and  $\notin SC$ 
6   $E'[ ] \leftarrow$  all composite vertices  $cv$ 
7  for each vertex  $v \in E$  and  $\notin SC$   $\emptyset$  loops: create edges between
8      for each  $sc \in SC$   $v \notin SC$  and composite vertices
9          if edges  $r[ ] = (v, sc) \in R$  exist
10              $R'[ ] \leftarrow r' \leftarrow$  create ( $v, cv$ )  $\emptyset cv \in E'$  resulted from  $sc \in SC$  in
11             else if  $r[ ] = (sc, v) \in R$  exist step 4
12              $R'[ ] \leftarrow r' \leftarrow$  create ( $cv, v$ )
13              $d'(r') \leftarrow$  avg(  $d(r)$  )
14  for each  $sc_1 \in SC$   $\emptyset$  loops: create edges between
15      for each  $sc_2 \in SC$  and  $sc_1 \neq sc_2$  composite vertices
16          if edges  $r[ ] = (sc_1, sc_2) \in R$  exist
17              $R'[ ] \leftarrow r' \leftarrow$  create ( $cv_1, cv_2$ )
18              $d'(r') \leftarrow$  avg(  $d(r)$  )

```

Algorithm 4: Generate Strongly Connected Component Graph⁷

There are two further possibilities which were considered for dealing with cyclic parts of graphs. These additional methods are described in the following paragraphs along with reasons for not choosing them.

7.2.2 Deleting edge(s)

The first alternative option to deal with cycles is to remove one edge contributing to form the cycle. Choosing the edge is an arbitrary choice since deleting any of the edges of the cycle will break it.⁸

This method of deleting edges was declined for the reason of two main drawbacks:

By deleting an edge, one certainly makes a choice of the order of trust for the remaining vertices. If, for instance, the edge (c, a) in graph G is deleted, the order of trust among the

⁷ The algorithm for the procedure call 'StronglyConnectedComponents()' is depicted in [Cormen, 2001, p. 554] or [Manber, 1989, p. 231].

⁸ There are types of strongly connected components where possibly more than one edge has to be deleted to dissolve the cyclic part. However, in most cases it remains ambiguous which edges should be removed.

entities which were part of the cycle is now c, b, a (in descending order) with entity a as least trustworthy. However, by deleting the edge (a, b) the resulting order is different in that node a is now the most trusted entity. The order of trust in descending order in this case is: a, b, c .

The second consequence of removing ambiguous parts of the graph (edges in this case) is, it affects consistency. Hence, further analyses of the relation graph will necessitate the consideration and evaluation of several cases such as how far removing edge (a, b) instead of edge (c, a) would affect the final result.

7.2.3 *Deleting node(s)*

It is also possible to break a cycle by removing one or more nodes v within a cycle to interrupt it. Additionally every edge (v, x) or (x, v) previously connected to the node is deleted as well. This method too has drawbacks similar to those mentioned above thus equivocally affecting the order of trust relation of remaining entities, as well as the possibility of affecting the consistency of the result.

Furthermore, depending on which node of the cycle is chosen to be deleted (which is basically an arbitrary choice, too) the resultant graph might become disconnected.⁹ This is the case, for instance, by removing node c to interrupt the cycle $\{a, b, c, a\}$ in the sample graph G . In this case, node g would become disconnected. Relations between entities in disconnected graphs cannot be resolved.

7.3 Handling Graph Recommendations

The merging and utilizing of other authorities' graphs is an ongoing process and it is easily possible that multiple graph recommendations are utilized. This section deals with this issue and shows how to cope with it.

7.3.1 *Merging Multiple Graphs*

When multiple graphs have to be unified, it is possible that a certain relation (directed link between two entities) is recommended more than once from different authorities. In particular is such a case of interest, if different distances are associated with the recommended relations. As stated in section 7.1, the distance for the resulting relation is determined by averaging their sources and an exemplary algorithm for merging two graphs was given. The following algorithm extends Algorithm 3 to make it suitable for multiple merging processes.

⁹ A graph becomes disconnected when a *cut-vertex* is deleted. "A vertex v of a connected graph G is a cut-vertex of G if and only if there exist vertices u and w ($u, w \neq v$) such that v is on every u - w path of G ." [Chartrand, 1986]

```

ALGORITHM 5: MERGEMULTIPLEGRAPHS( $G[ ]$ )
Input:  $G[ ] = \{G_1 = (E_1, R_1), \dots, G_n = (E_n, R_n)\}$  ( $n$  directed graphs)
Output:  $G' = G_1 \cup \dots \cup G_n$ 

1   $G' \leftarrow$  create empty graph
2  for each graph  $G_i \in G[ ]$ 
3    for each vertex  $e \in E_i$  and  $e \notin E'$ 
4       $E'[ ] \leftarrow e$   $\emptyset$  add vertex  $e$  to the list of vertices  $E'$ 
5    for each edge  $r_m \in R_i$ 
6      if  $r_m \notin R'$ 
7         $R'[ ] \leftarrow r'_m \leftarrow r_m$   $\emptyset$  save edge  $r_m$  as  $r'_m$  and add it to  $R'[ ]$ 
8         $r'_m.d \leftarrow d(r_m)$   $\emptyset$  save the distance of edge  $r_m$ 
9         $r'_m.count \leftarrow 1$   $\emptyset$   $r'_m.count$  counts the amount of same relations
10     else
11        $r'_m.d \leftarrow r'_m.d + d(r_m)$ 
12        $r'_m.count ++$   $\emptyset$  increment counter
13   for each edge  $r' \in R'$ 
14      $d(r') \leftarrow r'_m.d / r'_m.count$   $\emptyset$  compute the average distance for every  $r'$ 

```

Algorithm 5: Merge Multiple Graphs

To maintain consistency all relation graphs considered for unifying should be merged in one single step using Algorithm 5. An inconsistency when merging several graphs in multiple steps comes from the calculation of the distances. The result is different if multiple steps are used.

7.3.2 Giving Graph Recommendations

Up to now, only authorities have been considered who ask other authorities for their recommendations. In some cases, however, an authority may also get requests for recommendations.

When authorities recommend trust relations by providing their own relation graph, it is critical that they only provide their *initial relation graph* as introduced in chapter 6. Otherwise, when recommending relation graphs that already contain parts from other authorities or those that have already been ran through the algorithm that collapses cycles, inconsistencies in the same sense as in section 7.3.1 are bound to arise.

This concludes the discussion about the first of three steps. The next chapter will introduce the transfer of the merged context-based relation graph into Bayesian Networks.

8 Developing the structure of Bayesian Networks

In this chapter the general structure for the Bayesian Networks (BNs) is introduced. This is the second of three steps on the way to compute the total order of the entities in a relation graph.

This chapter is divided into three main sections.

- 1) An Introduction in Bayesian Networks,
- 2) A Bayesian Network for Trust Evaluation, and
- 3) Dynamic Conditional Probability Values for Bayesian Networks.

The first section gives a basic introduction into BNs. It shows where BNs are used, explains its structure and components and presents an introductory example.

The second section shows the transfer of the relation graph into BNs. It defines the basic structure of the BNs and prepares the basis for the dynamic extension developed in section 3.

In the third section the dynamic parts of the BNs are developed. This extension transfers the in the previous section modelled basic BNs into a dynamic network for individual consideration of the relations in the network.

With the dynamic BNs as result from this chapter, the following chapter 9 will introduce the method to compute the total order.

8.1 An Introduction in Bayesian Networks

Bayesian Networks, also known as Bayesian belief networks, probabilistic networks, or causal networks provide a method of reasoning using probabilities.

BNs are used in a variety of applications. Their main use takes place in the area of (medicine) diagnosis (e.g. the Pathfinder Project [Heckerman, 1992]). Other areas are, for example, language understanding, risk prediction, or forecasting.

Bayesian Networks are built on the structure of directed acyclic graphs. In the next paragraph the nodes of such a network will be discussed. This will be followed by a discussion about the edges and a short example to make it clearer.

The nodes are *random variables*. Every node has its own set of *values* which represent the states a variable can adopt. In the simplest case they are binary which means there are two values for every node. For example, imagine a node represents the activities of a 'fire department.' The values might be *fire* or *idle*, reflecting 'the fire department is at a fire' or 'the fire department is idle.' However, generally a node's number of values is not limited. To give an example for multiple values, think of an extension of the fire department's node with a value *drill*, reflecting 'the fire department is having a drill.'

Each possible value has a probability associated which expresses the degree of confidence in that value. For example, the probabilities for the fire department activity node might be

$P(\textit{fire}) = 0.9$ (strong belief that the fire department is at a fire), $P(\textit{idle}) = 0.0$ (no belief that the fire department is idle) and $P(\textit{drill}) = 0.1$ (small belief that the fire department is having a drill).¹⁰

The arcs in the BN specify the dependencies (probabilistic correlations) between the variables. The arrows always point towards the affected variable. In a very simple BN as shown in Figure 12, node E would have a direct influence in node T.

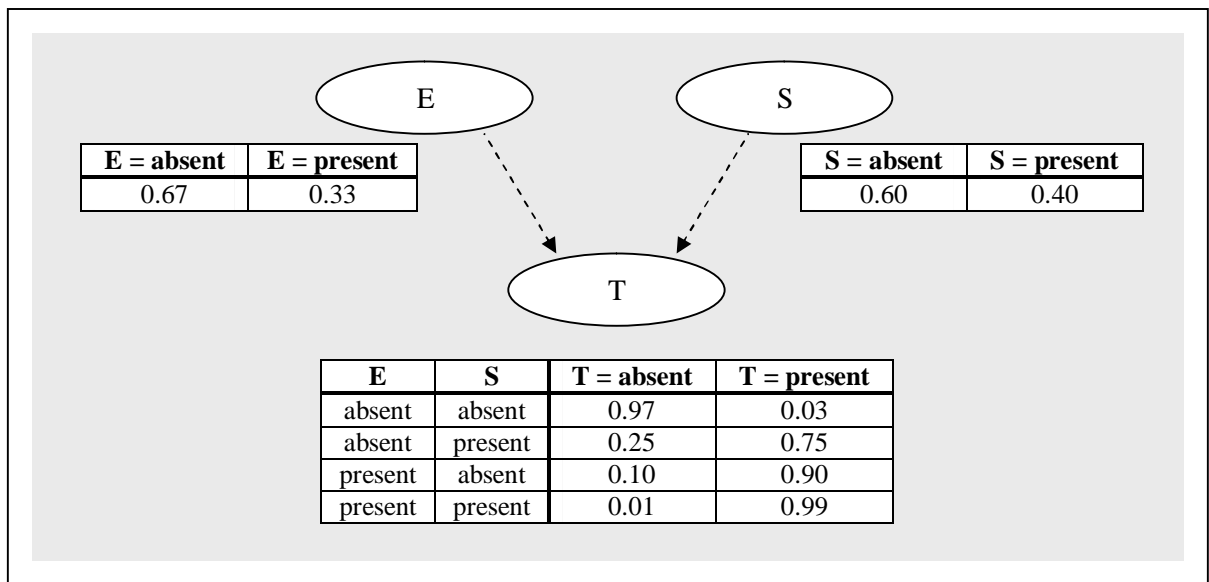


Figure 12: A Bayesian Network.

To give a short example, let node E stand for 'Event', node S for 'Snow' and node T for 'Traffic Jam'. All nodes can adopt the values *absent* and *present*, reflecting the possibility of the absence or presence of an event, snow and of traffic jam, respectively. The arrow from E to T expresses that the *absence* or *presence* of an event has a direct influence in the *absence* or *presence* of traffic jam. Also, the *absence* or *presence* of snow has a direct influence in the *absence* or *presence* of traffic jam.

How strong and in which way a node influences others is determined by 'Conditional Probability Tables' (CPTs). There is always one table for every node in a BN. The table lists the conditional probabilities, given for every combination of states of the node's parents in the network. For instance, the conditional probabilities for 'Traffic Jam' are shown in the lower table in Figure 12. The first line shows that if there is no event and no snow the probability that there will be a traffic jam is $P(\textit{Traffic Jam} = \textit{absent} \mid \textit{Event} = \textit{absent}, \textit{Snow} = \textit{absent}) = 0.03$.¹¹ The second line states the probability for a traffic jam if there is no event, but snow.

¹⁰ This example was taken from [Morawski, 1989].

¹¹ The probability-values are sample values and do not necessarily reflect a real-world scenario.

The third line represents the case if there is an event, but no snow. And the last line has the probability for a traffic jam if there is both, an event and snow.

If all needed (conditional) probability values are given, the BN can be fully evaluated and every node's degree of confidence in its states can be computed. "The basic computation on belief networks is the computation of every node's belief given the evidence that has been observed so far." [Charniak, 1991] In particular, Bayes' rule is used to calculate the belief at each node. A complete description of the process is also described in [Morawski, 1989] and [Neapolitan, 2004].

The evaluated BN for the example described above (also see Figure 12) is shown in Figure 13.

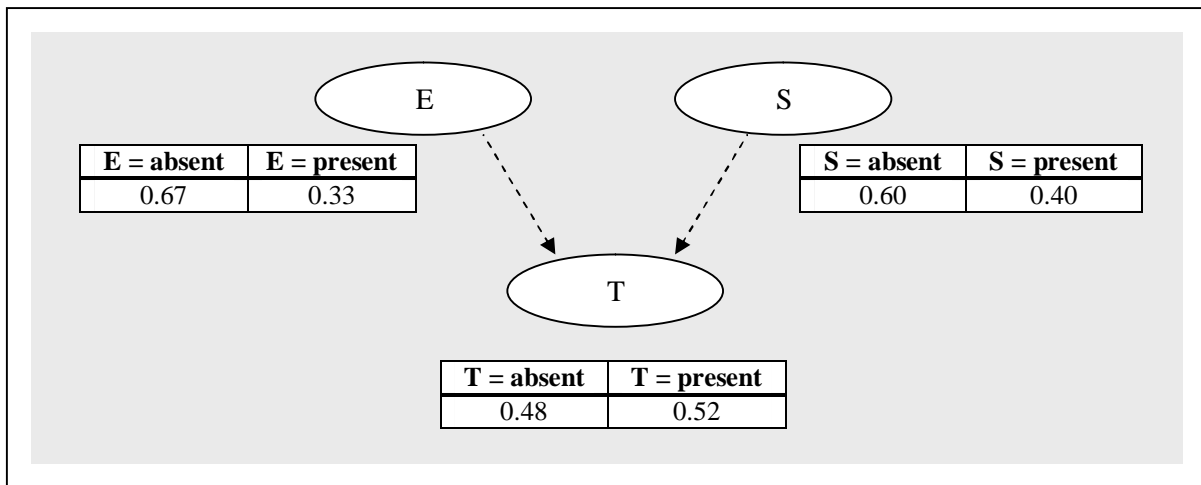


Figure 13: A complete evaluated BN. The probability values reflect the belief in every state of a variable.

In the rest of this paper, probability theory is used and the following formalization holds. Random variables (nodes) are generally denoted by a capital letter. $P(X = x)$ denotes the probability or probability distribution of the random variable X . Events are denoted by a small letter. Using the example above, the probability distribution of the random variable T (Traffic Jam) is $P(T = t)$ where t is either the event *absent* or *present*.

8.2 A Bayesian Network for Trust Evaluation

In this section and the following one will show how the Bayesian Networks are built to finally compute the total order of the entities. The model introduced in this thesis works with two Bayesian Networks. In the following these two structures are introduced. One is called the *Backward Graph*, the other the *Forward Graph*. In general, both networks are based on the structure of the *Context-based Relation Graph* defined in chapter 4.

For every knowledge-based network, the structure and the parameters which initialize the network are most important. The structure is for both networks (Backward Graph and Forward Graph) already given by the structure of the relation graph. The parameters have yet to be specified.

In BNs these parameters are gathered in the Conditional Probability Tables. The following diagram shows a roadmap to determine the CPTs.

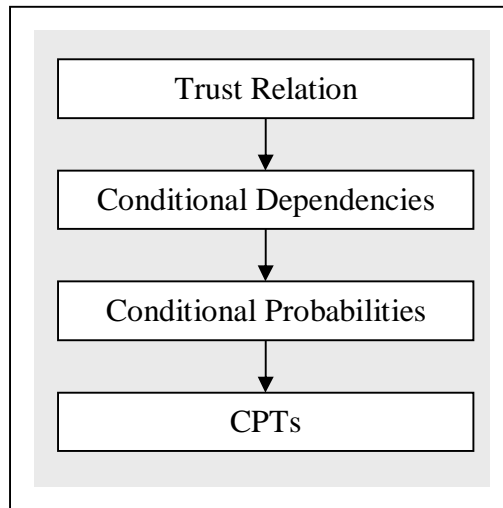


Figure 14: Roadmap to develop the CPTs.

The starting point is the Trust Relation specified in Definition 3. The examination of its trust statement "one entity is no less trustworthy than the other" will lead to the Conditional Dependencies. They specify the dependencies between two entities connected to each other by a trust relation. Then the Conditional Probabilities are determined which finally give the parameters for the CPTs.

In the next section, the basic CPTs for the Backward Graph are determined. Afterwards, the results are adapted for the Forward Graph.

8.2.1 Backward Graph

The first of the two Bayesian Networks is called the *Backward Graph*. It is based on the *Context-based Relation Graph* in the system introduced in chapter 4. This means

- 1.) the nodes in the relation graph are the nodes in the BN, and
- 2.) the edges in the relation graph are the arcs between the nodes.

For the Backward Graph, the edges of the relation graph are reversed. Figure 15 illustrates this transfer. Please note that lower letters are used for relation graphs and capital letters and dotted lines for BNs.

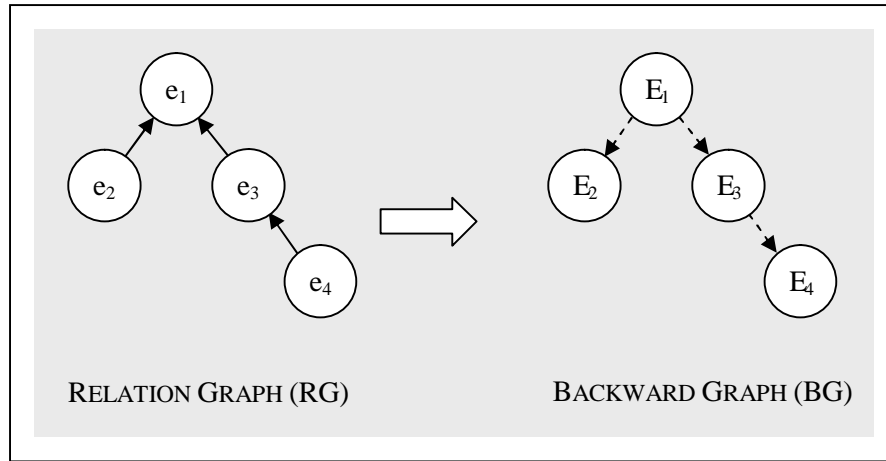


Figure 15: Transfer from Relation Graph to the Backward Graph which serves as the structure for the BN.

Every node in a BN has a set of values. The values reflect the states a node can adopt. It is important that the selected states are mutually exclusive and exhaustive. In case of this model, the nodes of the network represent the entities and every entity has the values *trustworthy* (tw) and *not trustworthy* (ntw). The states are mutually exclusive; no entity can be *trustworthy* and *not trustworthy* at the same time. The states are also exhaustive; for the final goal to evaluate the trustworthiness of an entity, the entities can either be *trustworthy* or *not trustworthy*, nothing else.

The BN has only nodes of the type 'Entity.' Therefore, there are only dependencies between 'Entities', represented by an arc between them ('Entity' → 'Entity'). At this point it is important to note that in BNs the direction of an arc plays a significant role. All conditional probability values in the CPTs are determined according to this direction.

Table 1 shows an exemplary CPT of node E_1 for a Relation Graph (RG) and its Backward Graph (BG). The contents of the table show that all probabilities are conditional dependent on entity E_2 which is the parental node of E_1 .

<p>RG</p>	<p>BG</p>		$E_1 = tw$	$E_1 = ntw$
	$E_2 = tw$	P($E_1 = tw \mid E_2 = tw$)	P($E_1 = ntw \mid E_2 = tw$)	
	$E_2 = ntw$	P($E_1 = tw \mid E_2 = ntw$)	P($E_1 = ntw \mid E_2 = ntw$)	

Table 1: CPT of node E_1 for a relation with two entities.

The goal is to determine the values for the conditional probabilities shown in Table 1. To specify these probabilities the *Conditional Dependencies* are determined and the probability values are derived from them.

Conditional Dependencies express in words in which way one entity is dependent on its connected parental entity or entities. Consider a relation between entity e_1 and entity e_2 such as shown beneath Table 1 ($e_1 \leq_c^A e_2$).

To determine the Conditional Dependencies for the relation between e_1 and e_2 it is to examine how the child node is conditional dependent from its parental node. It is therefore assumed E_2 is either in its state *not trustworthy* or in its state *trustworthy*. Both cases are depicted in Figure 16.

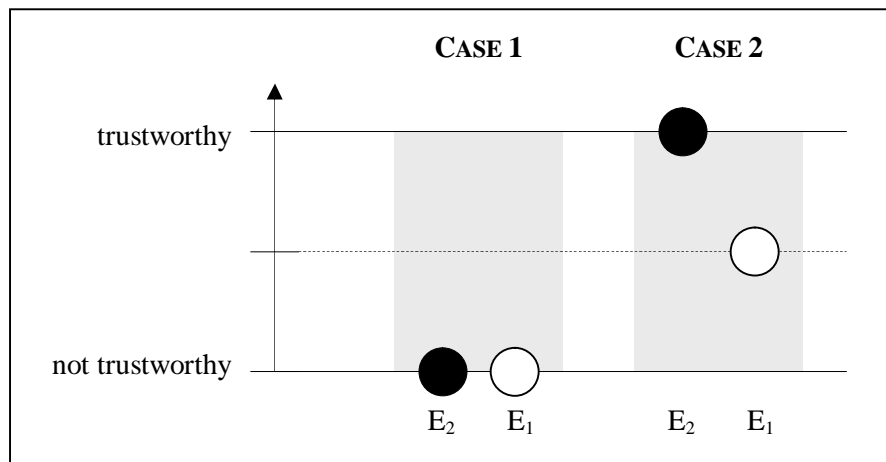


Figure 16: The conditional dependencies graphically illustrated for a relation between two nodes E_1 and E_2 in a Backward Graph.

In Case 1, E_2 is assumed to be *not trustworthy*. The original relation states E_1 is equally or less trustworthy than E_2 . In this case E_2 is already *not trustworthy* which concludes E_1 can only be *not trustworthy*, too. The conditional probability that E_1 is *not trustworthy* given E_2 is *not trustworthy* is therefore $P(E_1 = \text{ntw} \mid E_2 = \text{ntw}) = 1.00$. According to probability theory then $P(E_1 = \text{tw} \mid E_2 = \text{ntw}) = 0.00$.

In Case 2, E_2 is assumed to be *trustworthy*. Again, the original relation states E_1 is equally or less trustworthy than E_2 . If E_1 is 'equally trustworthy' then E_1 would be *trustworthy*, too. If E_1 is 'less trustworthy', E_1 could be anything between trustworthy and not trustworthy. Therefore is the conditional probability that E_1 is *trustworthy* given E_2 is *trustworthy*: $P(E_1 = \text{tw} \mid E_2 = \text{tw}) = 0.50$. Accordingly, $P(E_1 = \text{ntw} \mid E_2 = \text{tw}) = 0.50$.

Table 2 summarises these conclusions.

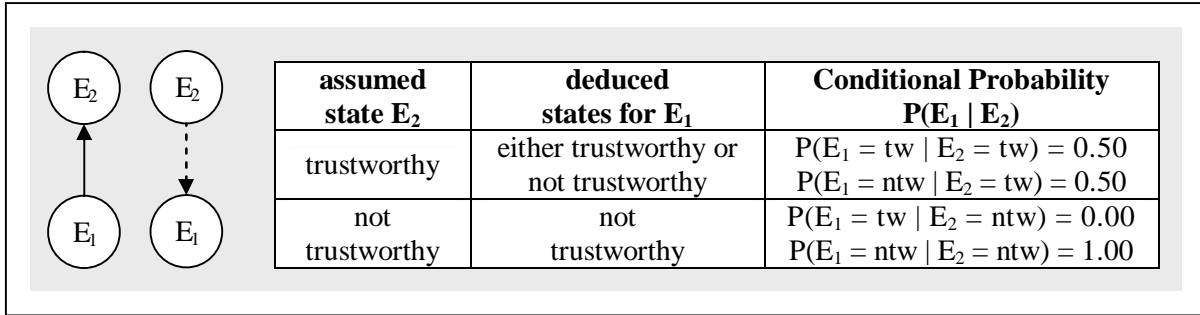


Table 2: Conditional Dependencies between two exemplary entities E_1 and E_2 .

Figure 17 shows the Bayesian network with its CPTs. The root probabilities for node E_1 are set to $P(E_1 = tw) = 0.50$ and $P(E_1 = ntw) = 0.50$ since the trustworthiness of a root node cannot be derived at this point. They will be considered in chapter 9.

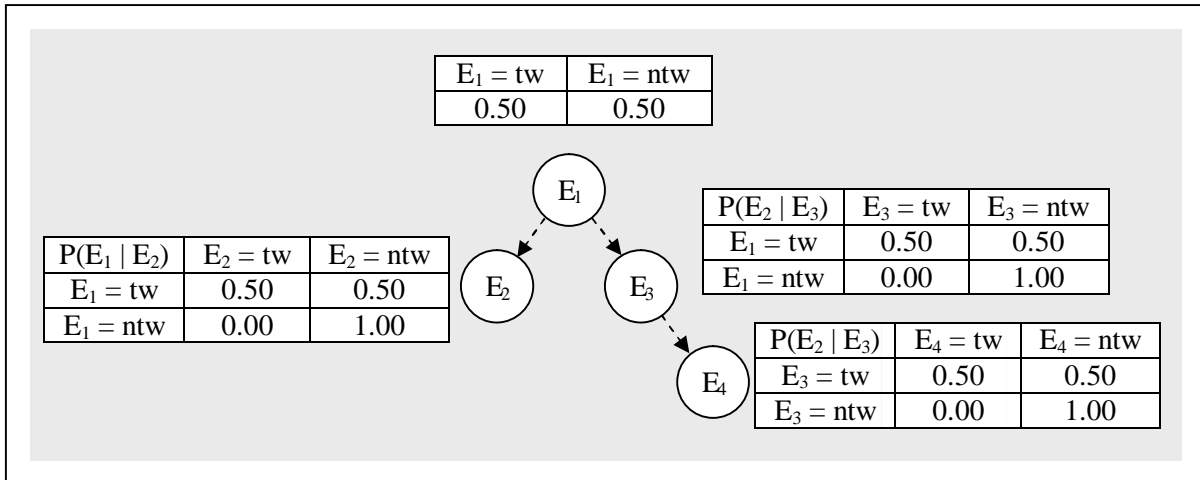


Figure 17: A Bayesian Network with its CPTs.

This concludes the basic structure for the first Bayesian network, *Backward Graph*.

8.2.2 Forward Graph

The second graph also uses the *Context-based Relation Graph* defined in chapter 4 as basis. Compared to the Backward Graph, however, all arc directions remain. It is called the *Forward Graph*. Please see Figure 18 for illustration.

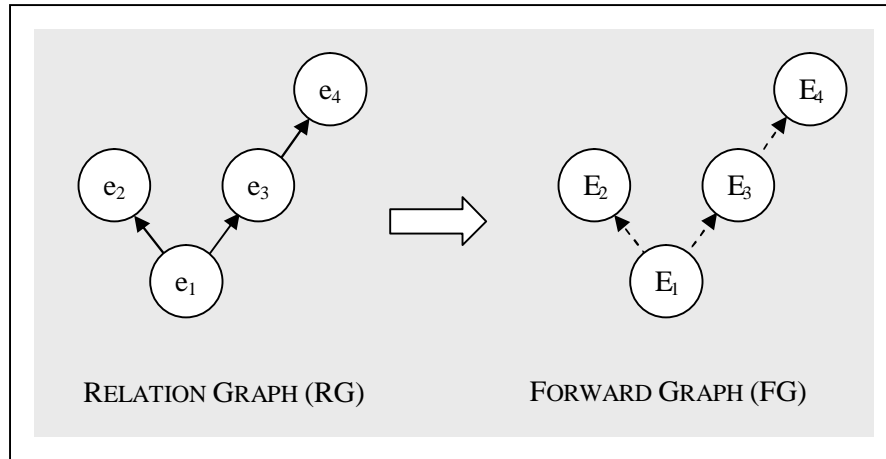


Figure 18: Transfer from Relation Graph to the Forward Graph which serves as structure for the BN.

Also for this graph the goal is to specify the values for the conditional probabilities in the CPTs. For a relation between two entities e_1 and e_2 such that $e_1 \leq_c^A e_2$, a CPT is given with Table 3. Consider the Forward Graph shown beneath the table. This time, E_2 is the child node and the probabilities have to be specified depending on its parental node, E_1 .

	$E_2 = tw$	$E_2 = ntw$
$E_1 = tw$	$P(E_2 = tw \mid E_1 = tw)$	$P(E_2 = ntw \mid E_1 = tw)$
$E_1 = ntw$	$P(E_2 = tw \mid E_1 = ntw)$	$P(E_2 = ntw \mid E_1 = ntw)$

Table 3: CPT for node E_2 for a relation with two entities.

To specify the conditional probabilities, again the *Conditional Dependencies* are determined considering the two cases. This time node E_1 is assumed to be either *trustworthy* or *not trustworthy*. Figure 19 illustrates both cases.

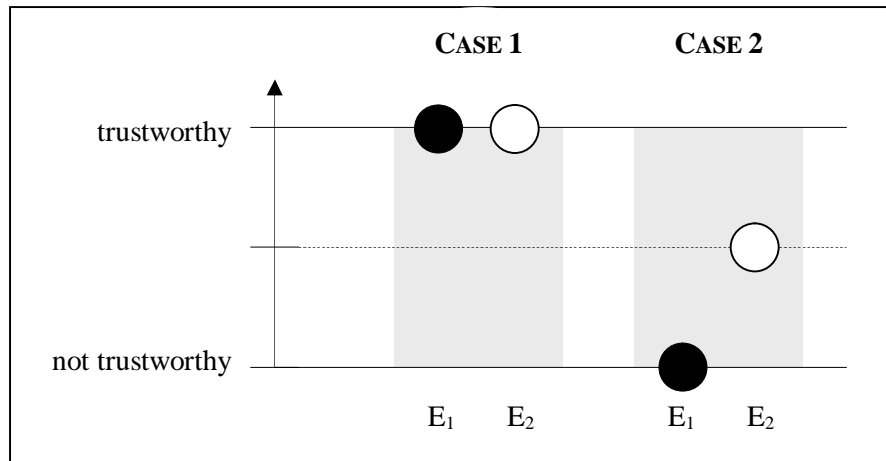


Figure 19: The conditional dependencies graphically illustrated for a relation between two nodes E_1 and E_2 in a Forward Graph.

In the first case node E_1 is assumed to be trustworthy. Since E_2 is, according to the relation graph, equally or more trustworthy than E_1 and E_1 is already assumed to be *trustworthy*, E_2 must be *trustworthy* as well. Therefore is E_2 concluded to be *trustworthy* given E_1 is *trustworthy*: $P(E_2 = tw | E_1 = tw) = 1.00$. Consequently is $P(E_2 = ntw | E_1 = tw) = 0.00$.

In the second case E_1 is assumed to be not trustworthy. E_2 with the condition to be equally or more trustworthy can either be *trustworthy* or *not trustworthy*. The conditional probability for E_2 to be *trustworthy* given E_1 is *not trustworthy* is therefore $P(E_2 = tw | E_1 = ntw) = 0.50$. And thus $P(E_2 = ntw | E_1 = ntw) = 0.50$.

For the Forward Graph, Table 4 summarises the conclusions.


	assumed state E_1	deduced states for E_2	Conditional Probability $P(E_2 E_1)$
	trustworthy	trustworthy	$P(E_2 = tw E_1 = tw) = 1.00$ $P(E_2 = ntw E_1 = tw) = 0.00$
not trustworthy	either trustworthy or not trustworthy	$P(E_2 = tw E_1 = ntw) = 0.50$ $P(E_2 = ntw E_1 = ntw) = 0.50$	

Table 4: Conditional Dependencies between two exemplary entities E_1 and E_2 .

Figure 20 shows the Bayesian network with its CPTs. Also for the Forward Graph are the root probabilities for node E_1 set to $P(E_1 = tw) = 0.50$ and $P(E_1 = ntw) = 0.50$ since the trustworthiness of a root node cannot be derived at this point. They will be considered in chapter 9.

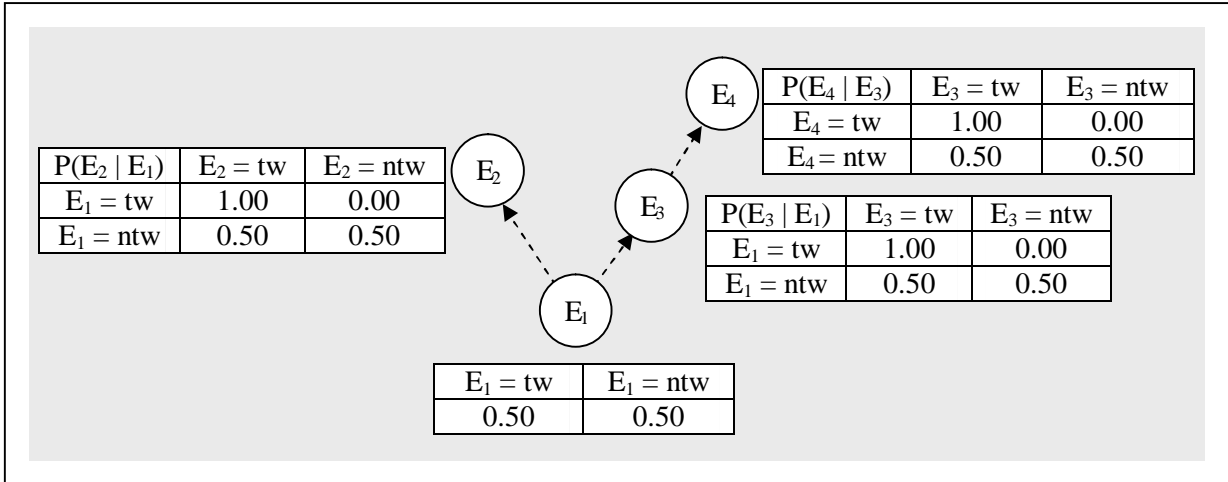


Figure 20: A Bayesian Network with its CPTs.

The CPTs for the networks in Figure 17 and Figure 20 are generic and thus every relation is treated equally. However, not every relation in the system is the same. In Definition 4 the notion of distance was specified and explained throughout in section 6.2. This notion will be used to extend the above described method to determine the CPTs with a more individual and dynamic approach. This is discussed in the next section.

8.3 Dynamic Conditional Probability Values for the Bayesian Networks

Up to now, the generic structures of the two BNs have been defined. The way to make this system dynamic is by determining the values for the CPTs according to an individual, relation-dependent aspect.

In this section it is discussed how the CPTs and the accompanying conditional probabilities can be specified to meet a relation's conditions dynamically. This issue will be discussed using the Backward Graph as subject. Afterwards, the dynamic CPTs for Forward Graphs will be specified by deriving the results developed next.

8.3.1 Filling the CPTs dynamically

The connecting link between generic and dynamic CPTs is the notion of distance between two entities which was introduced with Definition 4. A dynamic CPT is expressed by individual conditional probabilities for every relation, dependent on the distance between two entities.

Please recall Figure 16 presented in section 8.2.1 for a Backward Graph with the two nodes E_1 and E_2 . The relation was $e_1 \leq_c^A e_2$. In 'Case 1' E_2 's state was assumed to be *not trustworthy*. The conclusion was E_1 can only be *not trustworthy*, too. In 'Case 2' E_2 's state was assumed to be *trustworthy*. E_1 was derived to be either *trustworthy* or *not trustworthy*.

In the next step, the two cases above are revised by considering the conditional dependencies related to the distance between the two entities.

In Case 1, E_2 is assumed to be not trustworthy. Compared to section 8.2.1, E_1 remains to be not trustworthy. Even in the best case for E_1 (minimal distance to E_2), E_1 can never get trustworthy. Thus, E_1 remains not trustworthy given E_2 is not trustworthy ($P(E_1 = \text{ntw} | E_2 = \text{ntw}) = 1.00$).

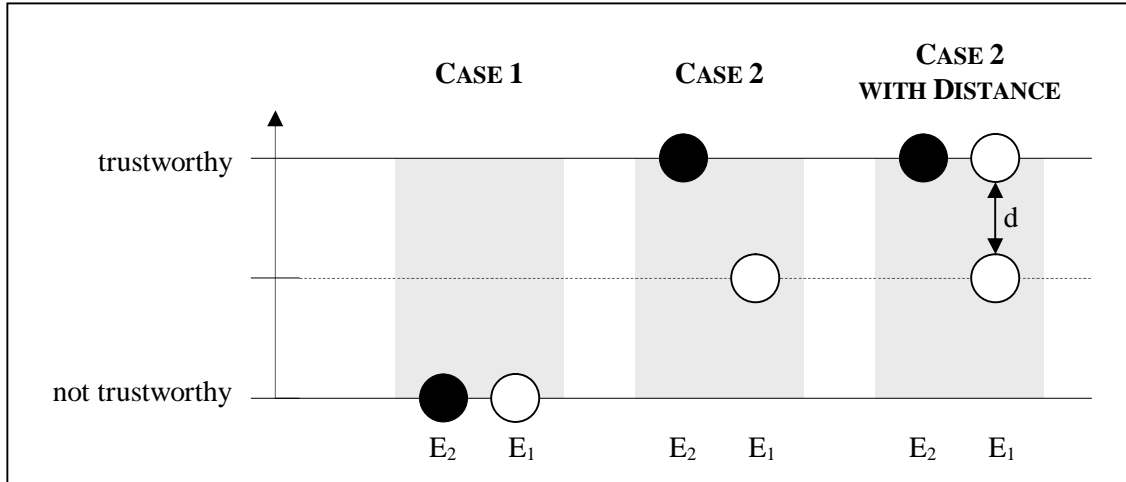


Figure 21: The dynamic conditional dependencies graphically illustrated for a relation between two nodes E_1 and E_2 in a Backward Graph.

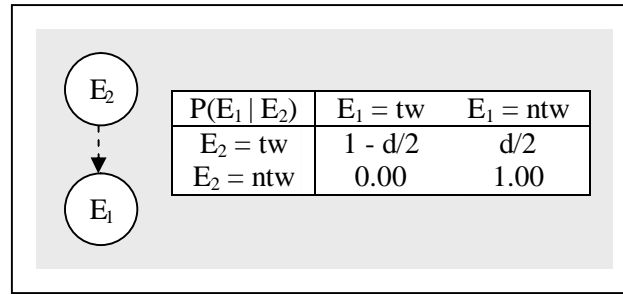
In Case 2, E_2 is assumed to be trustworthy. Using distances gives a measure to determine the trustworthiness of E_1 . If the distance is minimal (both entities are equally trustworthy) and E_2 is trustworthy, E_1 is consequently trustworthy as well. The greater the distance between the two of them gets, the smaller gets the probability for E_1 to be trustworthy. If the distance is maximal, E_1 is independent from the state of E_2 and can either be trustworthy or not trustworthy. Please see Figure 21 (Case 2 with Distances) for illustration.

To put it in numbers, with a minimal distance ($d = 0.0$) the conditional probability $P(E_1 = \text{tw} | E_2 = \text{tw}) = 1.0$. With a maximal distance ($d = 1.0$) the conditional probability $P(E_1 = \text{tw} | E_2 = \text{tw}) = 0.5$. Hence, the conditional probabilities range between the borders 0.5 and 1.0, dependent on the distance. The values between these borders are assumed to be proportional to the distance d .

The following formula defines the conditional probability for a relation with two entities in the *Backward Graph* given node E_2 is trustworthy:

$$P(E_1 = \text{tw} | E_2 = \text{tw}) = 1 - d/2. \quad (8.1)$$

Table 5 shows the current dynamic CPT for a node E_1 with one parental node (singly connected node) for a Backward Graph, given the relation $e_1 \leq_c^A e_2$.



$P(E_1 E_2)$	$E_1 = tw$	$E_1 = ntw$
$E_2 = tw$	$1 - d/2$	$d/2$
$E_2 = ntw$	0.00	1.00

Table 5: Dynamic CPT for singly connected nodes in the Backward Graph.

The dynamic CPTs for the *Forward Graph* are very similar to those of the Backward Graph. However, in the Forward Graph, node E_2 is conditional dependent on node E_1 .

Please recall Figure 19 presented in section 8.2.2. The relation was $e_1 \leq_c^A e_2$.

In Case 1, E_1 is assumed to be trustworthy. Compared to section 8.2.2, E_2 remains to be trustworthy. Even in the worst case for E_2 (minimal distance to E_1), E_2 can never become not trustworthy. Thus, E_2 remains not trustworthy given E_1 is trustworthy ($P(E_1 = tw | E_2 = tw) = 1.00$).

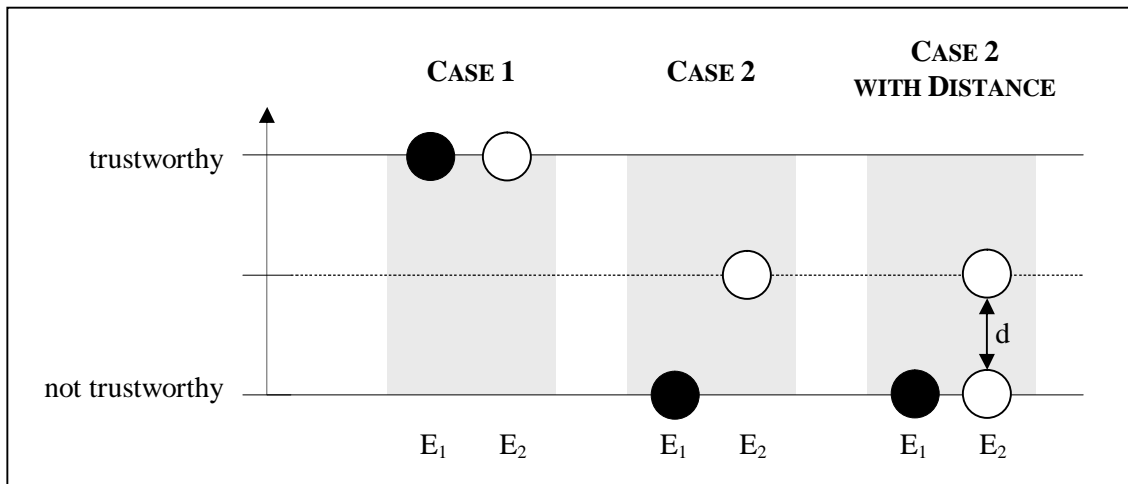


Figure 22: The dynamic conditional dependencies graphically illustrated for a relation between two nodes E_1 and E_2 in a Forward Graph.

In Case 2, E_1 is assumed to be not trustworthy. If the distance is minimal (both entities are equally trustworthy) and E_1 is not trustworthy, E_2 is consequently not trustworthy either. The greater the distance between the two of them gets, the greater the probability for E_2 to be trustworthy. If the distance is maximal, E_2 is independent from the state of E_1 and can in consequence be either trustworthy or not trustworthy. Please see Figure 22 (Case 2 with Distances) for illustration.

To put it in numbers, with a minimal distance ($d = 0.0$) the conditional probability $P(E_1 = \text{ntw} \mid E_2 = \text{ntw}) = 1.0$. With a maximal distance ($d = 1.0$) the conditional probability $P(E_1 = \text{ntw} \mid E_2 = \text{ntw}) = 0.5$. Hence, the conditional probabilities range between the borders 0.0 and 0.5, depending on the distance. The values between these borders are assumed to be proportional to the distance d .

The following formula defines the conditional probability for a relation between two entities in the Forward Graph given node E_1 is not trustworthy:

$$P(E_1 = \text{tw} \mid E_2 = \text{ntw}) = d/2. \quad (8.2)$$

Table 6 summarizes the values for the conditional probabilities for the current CPT assigned to a node with one parent (singly connected node) in a Forward Graph, given the relation $e_1 \leq_c^A e_2$.

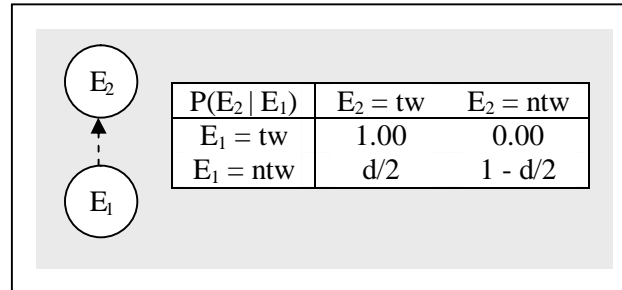


Table 6: Dynamic CPT for singly connected nodes in the Forward Graph.

At this point, the dynamic CPTs for both BNs, the *Forward Graph* and the *Backward Graph* are developed. This concludes the discussion about dynamic CPTs for singly connected entities.

In the next section the CPTs for nodes with multiple parental nodes are discussed.

8.3.2 CPTs for nodes with multiple parental nodes

This section deals with the CPTs in the BNs in case of multiple parental nodes. A sample relation graph and its Backward Graph illustrating the situation are shown in Figure 23.

For the CPT of node E_x , the conditional probabilities for all possible combinations of states of its parents must be given. For instance, node 'Traffic Jam' in the small example in section 8.1 had two parental nodes; each of them with the two states 'present' and 'absent'. The CPT for 'Traffic Jam' therefore provided conditional probabilities for every possible combination of these two states.

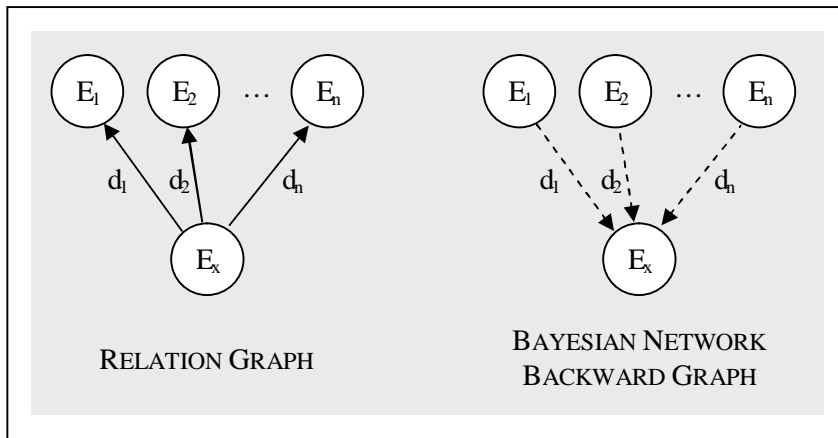


Figure 23: Backward Graph: Node E_x with multiple parental nodes.

So far, only the probabilities which were conditioned on the states of *one* predecessor node had to be determined. The next step is to determine the probabilities for a child node given n parental nodes.

Assume such a node with n parental nodes as shown in Figure 23 (right). Every node has two states (trustworthy and not trustworthy). For a CPT for node E_x , 2^n probability values have to be specified since all possible combinations of the parental nodes' states have to be considered. The problem at this point is that only the conditional probabilities for single relations are known, rather than the probabilities conditioned on every possible combination of the parental nodes' states.

This problem is known and was addressed by Pearl in 1988 by introducing the noisy OR-gate model [Pearl, 1988]. It uses, in such cases, only the individual conditional probabilities to determine the 2^n probabilities for the CPT. Figure 24 illustrates this.

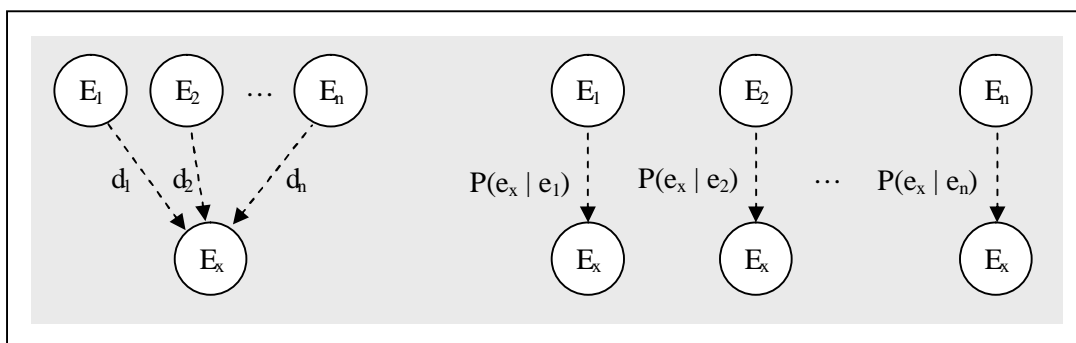


Figure 24: Rather than determining the conditional probabilities for every possible combination of the states of n parental nodes, the noisy OR-gate model uses the conditional probability for every single relation.

Before launching into a description of how the noisy OR-gate model is used in this work, it is worth discussing the basics of it in more detail.

The noisy OR-gate, widely used in Bayesian Networks, approximates the conditional probability distributions such that fewer parameters are required. The original noisy OR-gate introduced by Pearl, models the case in which every variable in the relation has only two states. Usually these values are called binary and reflect the states *absent* and *present* or *false* and *true*, respectively (see also the example in section 8.1).

In BNs the relation between two variables are defined by *cause* and *effect*. Usually, the parental node causes the effect reflected by the child node. In the example in section 8.1, for instance, the cause, snow, produced (for a certain probability) the effect, traffic jam.

Given a set X of n binary variables as causes (see Figure 25), the model from Pearl assumes that every variable can cause an effect independent of other causes. In other words, the sole presence of the cause X_i (e.g. snow) is enough to produce the effect Y (e.g. traffic jam).

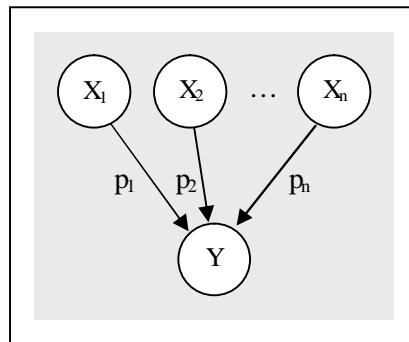


Figure 25: The effect node Y has n binary variables as causes.

The probability of causing an effect is usually termed by p_i which is the conditional probability that an effect Y will occur, given the sole presence of X_i . In [Neapolitan, 2004] p_i is referred to as *causal strength*.

To derive a complete CPT for a set of n parental nodes ($X = \{X_1, X_2, \dots, X_n\}$) the following formula is given by Pearl. Hereby, let S be a set of indices such that $i \in S$ if and only if $X_i = \text{true}$.¹²

$$p(Y = \text{true} \mid X = x) = 1 - \prod_{i \in S} (1 - p_i) \quad (8.3)$$

¹² The formalization is partly taken from [Neapolitan, 2004].

The noisy OR-gate as such regards only causes which are *present*. It consequently models that the absence of all causes at no time produces a presence of Y.

Henrion extended in his work [Henrion, 1989] the noisy OR-gate by additionally modeling that an effect Y can spontaneously occur even if all causes are *absent*. He introduced the concept of a *leak probability* p_0 .¹³ The extended *leaky noisy OR-gate* formula is given by:

$$p(Y = true | X = x) = 1 - (1 - p_0) \prod_{i \in S} \frac{(1 - p_i)}{(1 - p_0)} \quad (8.4)$$

The noisy OR-gate model is used for the model introduced in this thesis. It provides a method to use known conditional probabilities to approximate the values for a complete CPT if multiple parental nodes are given.

Henrion's extension is also used in this thesis. In particular, the leak probability corresponds in this system to the case in which an effect node E_x is independent from its predecessors. This is the case if all distances are maximal. In probability theory it holds, if event A is independent from event B, $P(B | A) = P(B)$. The conditional probabilities in case of independence between E_x and its predecessors is $P(E_x = e_x | E_n = e_n) = 0.5$. Thus, for this system $p_0 = 0.5$ holds.

Using the leaky noisy OR-gate as given, tests showed that the trustworthiness was dependent on the number of predecessors, rather than on the distance between the nodes. If the number of predecessors a node E_x has increased, its trustworthiness increased or decreased as well, dependent on the use of the Forward or Backward Graph. Such behaviour could have easily been used to manipulate results.

As a consequence the following requirement for the BNs was introduced. To describe it, let $E = \{E_1, E_2, \dots, E_n\}$ be the set of random variables, and let $e = \{e_1, e_2, \dots, e_n\}$ be a set of values of the variables in E. Further, let the nodes in E be the direct predecessors of E_x (see Figure 24). Then the requirement for both, the Backward and Forward Graph is:

The number of direct predecessor nodes E must not influence the trustworthiness of the direct successor node E_x .

This implies that if, for instance, *all* distances $d_n = 0.00$ and *all* parental nodes are equally trustworthy, E_x is as trustworthy as its parental nodes, regardless of the potency of set E.

As described above, the noisy OR-gate only takes the conditional probabilities into account where the predecessor nodes' states $X_i = true$, or respectively $E_i = trustworthy$ in the Backwards Graph. By empirical study it was found that weighting the CPT values according to the number of influencing probabilities achieves the specified requirement.

For illustration, Table 7 shows which *individual* conditional probabilities for the calculation of $P(E_x = tw | E = e)$ ¹⁴ in case of three predecessors are considered. Let S be a set of indices such

¹³ For further details please see [Henrion, 1989].

¹⁴ In case of three predecessors, $P(E_x = tw | E = e) = P(E_x = tw | E_1 = e_1, E_2 = e_2, E_3 = e_3)$.

that $i \in S$ if and only if $E_i = tw$ and $\neg S$ be a set of indices such that $j \in \neg S$ if and only if $E_j = ntw$.¹⁵ Further, let $|S|$ be the potency of S .

$E_1 = e_1$	$E_2 = e_2$	$E_3 = e_3$	$ S $	$P(E_x = tw \mid E_1 = e_1, E_2 = e_2, E_3 = e_3)$
tw	tw	tw	3	$P(E_x = tw \mid E_1 = tw), P(E_x = tw \mid E_2 = tw), P(E_x = tw \mid E_3 = tw)$
tw	tw	ntw	2	$P(E_x = tw \mid E_1 = tw), P(E_x = tw \mid E_2 = tw)$
tw	ntw	tw	2	$P(E_x = tw \mid E_1 = tw), P(E_x = tw \mid E_3 = tw)$
tw	ntw	ntw	1	$P(E_x = tw \mid E_1 = tw)$
ntw	tw	tw	2	$P(E_x = tw \mid E_2 = tw), P(E_x = tw \mid E_3 = tw)$
ntw	tw	ntw	1	$P(E_x = tw \mid E_2 = tw)$
ntw	ntw	tw	1	$P(E_x = tw \mid E_3 = tw)$
ntw	ntw	ntw	0	---

Table 7: Using the noisy-OR gate, only those conditional probabilities are considered for $P(E_x = tw \mid e_1, e_2, e_3)$ where the conditioned predecessor $E_i = trustworthy$ (Backward Graph). The table lists which probabilities are considered in each case. $|S|$ is the number of considered conditional probabilities.

The leaky noisy OR-gate formula extended with the weighting factor is given for a *Backward Graph* with formula (8.5).

$$P(E_x = tw \mid E = e) = \frac{|S|}{|S| + |\neg S|} \cdot \left[1 - 0.5 \prod_{i \in S} \frac{1 - P(E_x = tw \mid E_i = tw)}{0.5} \right] \quad (8.5)$$

Since $P(E_x = tw \mid E_i = tw) = 1 - d_i/2$ (see formula (8.1)), $P(E_x = tw \mid E_i = tw)$ can be substituted. The result is formula (8.6).

$$P(E_x = tw \mid E = e) = \frac{|S|}{|S| + |\neg S|} \cdot \left[1 - 0.5 \prod_{i \in S} d_i \right] \quad (8.6)$$

The following table shows as an example the conditional probabilities $P(E_x = e_x \mid E = e)$ calculated with (8.6). It is the CPT for a successor node in a Backward Graph as shown in Figure 24 with three parental nodes, E_1 , E_2 , and E_3 . The assumed distances are $d_1 = 0.4$, $d_2 = 0.8$, and $d_3 = 0.5$.

¹⁵ The formalization is partly taken from [Neapolitan, 2004].

E_1	E_2	E_3	$ S $	$P(E_x = tw e_1, e_2, e_3)$	$P(E = ntw e_1, e_2, e_3)$
tw	tw	tw	3	0.92	0.08
tw	tw	ntw	2	0.56	0.44
tw	ntw	tw	2	0.60	0.40
tw	ntw	ntw	1	0.27	0.73
ntw	tw	tw	2	0.53	0.47
ntw	tw	ntw	1	0.20	0.80
ntw	ntw	tw	1	0.25	0.75
ntw	ntw	ntw	0	0.00	1.00

Table 8: Conditional probabilities for a Bayesian network as shown in Figure 24 with 3 parental nodes.

Previous formula (8.6) is solely for calculating CPTs for the Backward Graph. The formula for the Forward Graph is introduced next. An exemplary Forward Graph is shown below in Figure 26.

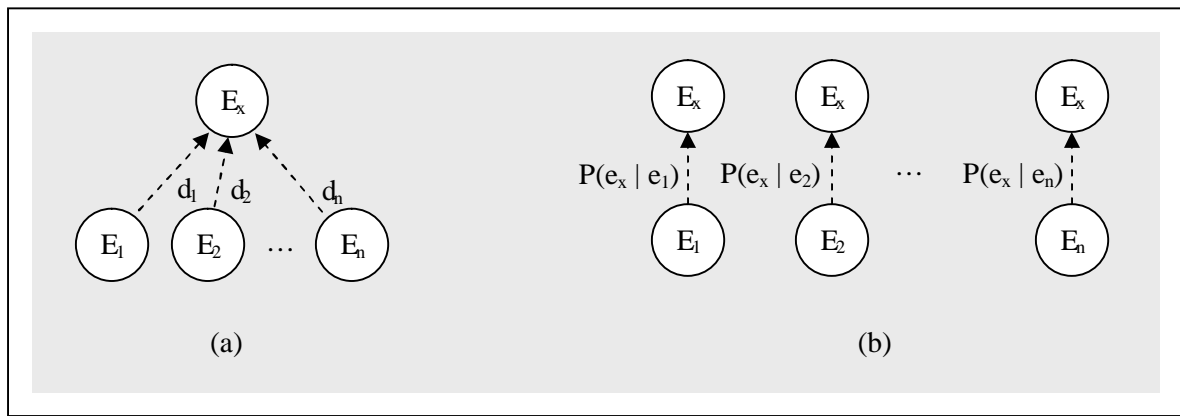


Figure 26: (a) Forward Graph with a single node E_x and multiple predecessors. (b) The graph's division in its single relations.

Also for the second BN, the leaky noisy-OR gate model is used. The following formula (8.7) is given for the *Forward Graph*.

$$P(E_x = ntw | E = e) = \frac{|\neg S|}{|S| + |\neg S|} \cdot \left[1 - 0.5 \prod_{i \in \neg S} \frac{P(E_x = tw | E_i = ntw)}{0.5} \right] \quad (8.7)$$

Since $P(E_x = tw | E_i = ntw) = d_i/2$ (see formula (8.2)), $P(E_x = tw | E_i = ntw)$ can be substituted. The result is formula (8.8).

$$P(E_x = ntw | E = e) = \frac{|\neg S|}{|S| + |\neg S|} \cdot \left[1 - 0.5 \prod_{i \in \neg S} d_i \right] \quad (8.8)$$

The following table shows the conditional probabilities $P(E_x = e_x | E = e)$ calculated with (8.8). It is the CPT for node E_x as shown in Figure 26 with three predecessor nodes E_1 , E_2 , and E_3 . The assumed distances are $d_1 = 0.4$, $d_2 = 0.8$, and $d_3 = 0.5$.

E_1	E_2	E_3	$ \neg S $	$P(E_x = tw e_1, e_2, e_3)$	$P(E = ntw e_1, e_2, e_3)$
tw	tw	tw	0	1.00	0.00
tw	tw	ntw	1	0.75	0.25
tw	ntw	tw	1	0.80	0.20
tw	ntw	ntw	2	0.47	0.53
ntw	tw	tw	1	0.73	0.27
ntw	tw	ntw	2	0.40	0.60
ntw	ntw	tw	2	0.44	0.56
ntw	ntw	ntw	3	0.08	0.92

Table 9: Conditional probabilities for a Bayesian network as shown in Figure 26 with 3 parental nodes.

This concludes the discussion about the development of the two Bayesian Networks, Forward Graph and Backward Graph.

The following diagram summarizes the discussion of the two networks.

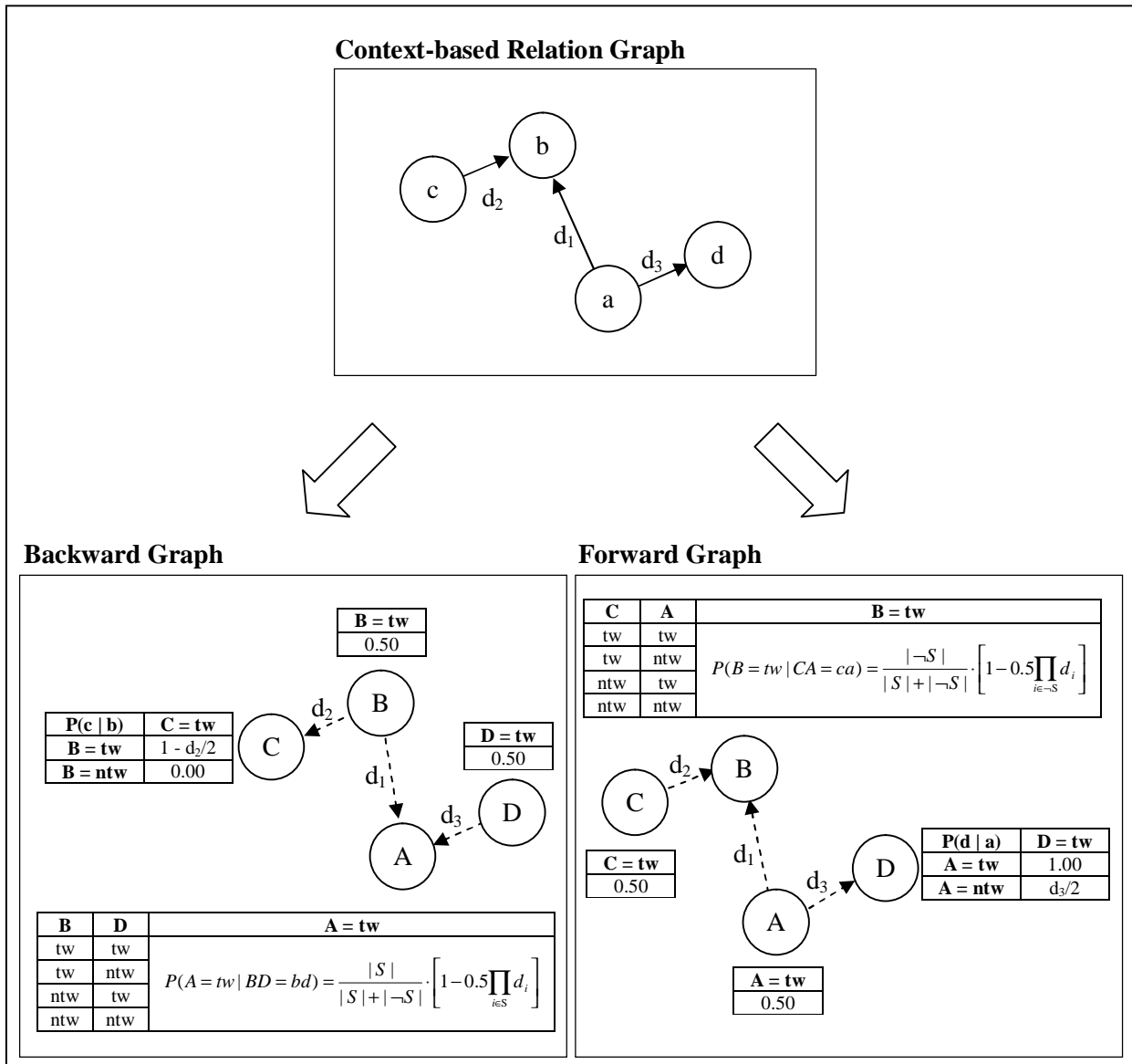


Figure 27: Summarized development of the dynamic Backward and Forward Graphs.

The algorithm to transfer a Relation Graph into two Bayesian Networks is given with Algorithm 6. It creates both BNs using the structure of the relation graph given as input and computes the CPTs for every node. Hereby the formulas (8.1) and (8.6) are used to calculate the CPTs of the Backward Graph and the formulas (8.2) and (8.8) to calculate the CPTs for the Forward Graph.

```

ALGORITHM 6: TRANSFERRELATIONTOBNS(G)
Input:  $G = (E, R)$  (Context-based Relation Graph)
Output: 2 Bayesian Network models (BG and FG), each over the set of variables E
1   BG ← create Bayesian Network using structure of G, but reversing edges
2   for each variable  $v \in BG$ 
3     if  $v.parents[] = \emptyset$  then set a-priori probabilities to 0.5
4     else if  $v.parents[].count = 1$ 
5       compute CPT using formula (8.1)
6     else compute CPT using formula (8.6)
7   FG ← create Bayesian Network using structure of G
8   for each variable  $v \in FG$ 
9     if  $v.parents[] = \emptyset$  then set a-priori probabilities to 0.5
10    else if  $v.parents[].count = 1$ 
11      compute CPT using formula (8.2)
12    else compute CPT using formula (8.8)

```

Algorithm 6: Transfer Relation Graph to Bayesian Networks

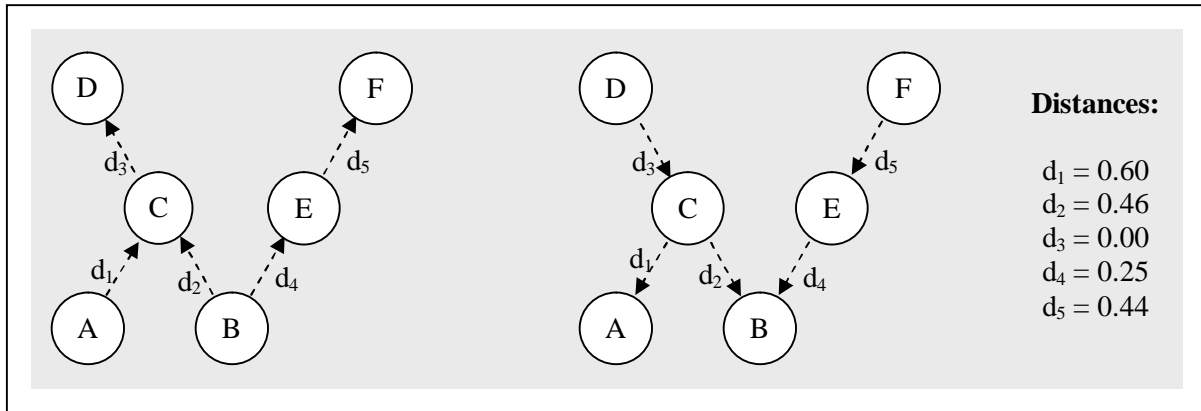
Since now both network structures are known, it is possible to continue with step 3 to compute the total order of the nodes in the relation graph.

In the following chapter the procedure of determining the total order will be described using the Bayesian Networks developed in this chapter.

9 Determination of the Total Order

In this chapter it will be shown how the total order of the entities connected in one context-based relation graph are finally determined.

Assume two BNs as shown in Figure 28 (for a set of entities $X = \{A, B, C, D, E, F\}$). Also, assume the CPTs of all nodes are already initialized with the conditional probabilities according to the given distances.



**Figure 28: Two Bayesian Networks originated from the same relation graph.
 (a) Backward Graph; (b) Forward Graph**

The first step is to *evaluate* the two Bayesian Networks using *Propagation Algorithms*. They are provided, for example, by Pearl (Message Propagation Method, [Pearl, 1988]) or by Lauritzen & Spiegelhalter (Clustering Method, [Lauritzen, 1990]).

These algorithms compute the belief in every single state of a random variable. Hence, after evaluating the BNs, the values $P(X_i = \text{trustworthy})$ and $P(X_i = \text{not trustworthy})$ are computed for every entity in each graph. Hereby $P(X_i = \text{tw}) + P(X_i = \text{ntw}) = 1.00$. Figure 29 shows the evaluated networks and the calculated beliefs for each node.¹⁶

¹⁶ For clarity only $P(E_i = \text{tw})$ is depicted for each node.

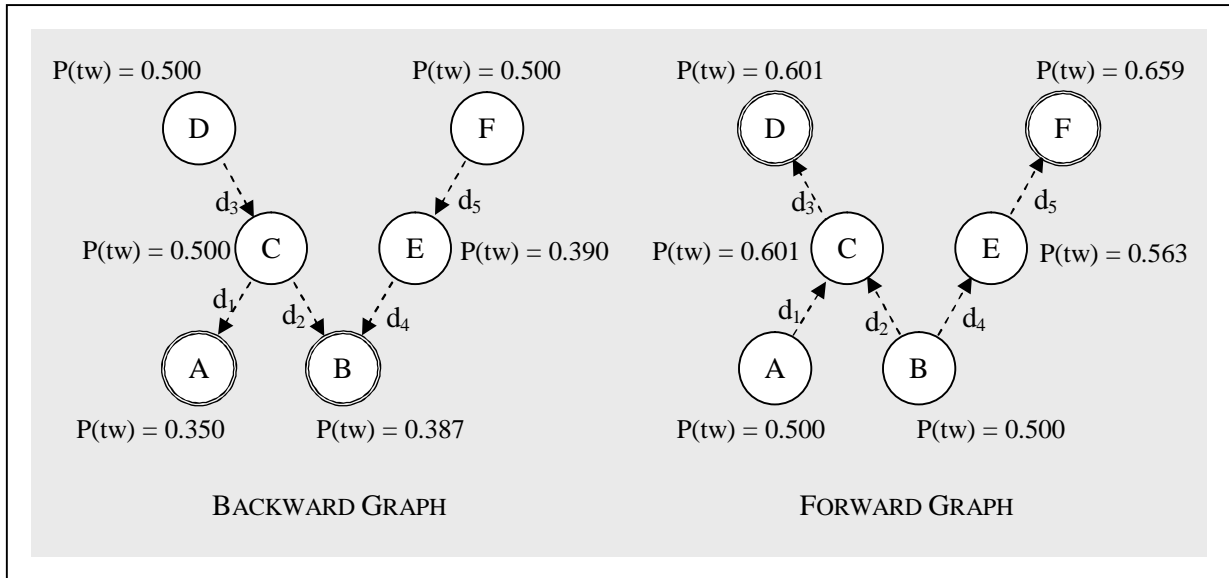


Figure 29: The two evaluated Bayesian Networks.

It is not possible to compute the total order by just evaluating one of the two networks. The reason is that the root probabilities for both networks are unknown. This model works solely on the basis of trust relations and their distances between two entities. Therefore the trust values or beliefs for single entities which would be needed to initialize the root nodes are not available.

Hence, to compute the total order of the set of entities X , a second step is necessary. The approach is to merge the results of the two BNs into one.

Both networks have *root nodes* which are the nodes that have no incoming edges. And both networks have *local sinks* which are nodes that have no exiting edges. In the networks in Figure 29 the local sinks are marked by a double line.

By initializing the root nodes of both networks with the probability of $P(\text{trustworthy}) = 0.5$, all entities serving as root nodes are set on the same level of trust.

Metaphorically speaking, the root nodes of each BN are building the ground level (see Figure 30). In case of an evaluated Forward Graph, all beliefs are equal or greater than 0.5 (see Figure 29, Forward Graph). The beliefs of the local sinks (node D and F) are the greatest. This allows, again metaphorically speaking, the evaluation of the 'peaks' of the network.

In case of the Backward Graph, all beliefs are equal or smaller than 0.5. The beliefs of the local sinks (node A and B) are the smallest (see Figure 29, Backward Graph). This allows the evaluation of the 'depth of the roots' of the network.

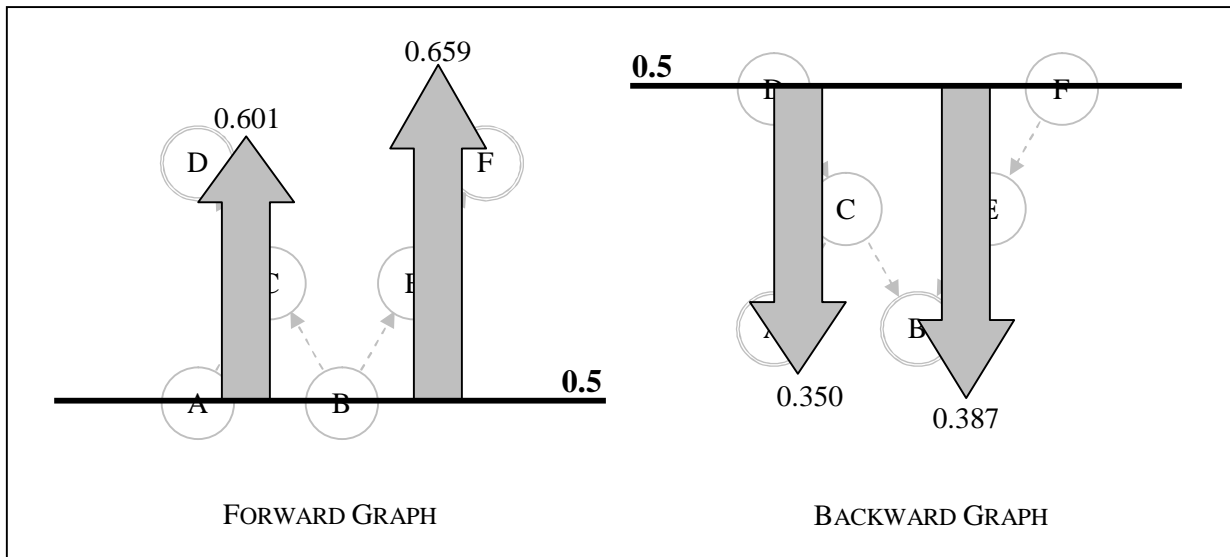


Figure 30: The root nodes are initialized with the same value. By evaluating the Forward Graph, the peaks of the network are calculated. By evaluating the Backward Graph, the roots of the network are calculated.

Putting these beliefs (the peak values from the Forward Graph and the root values from the Backward Graph) together into one graph 'balances the network', with every node evening out at a certain level.

In other words, to 'merge' the results of the two networks, the beliefs for variable A and B from the Backward Graph are used as *new* root probabilities in the Forward Graph. The Forward Graph is instantly re-evaluated. The beliefs of the local sinks D and F in the Forward Graph are entered as likelihood values ('virtual evidence' [Morawski, 1989]) in the re-evaluated Forward Graph.

The final step is to sort the computed values in ascending order to get the *total order* of the entities. Figure 31 summarizes the final steps in principal.

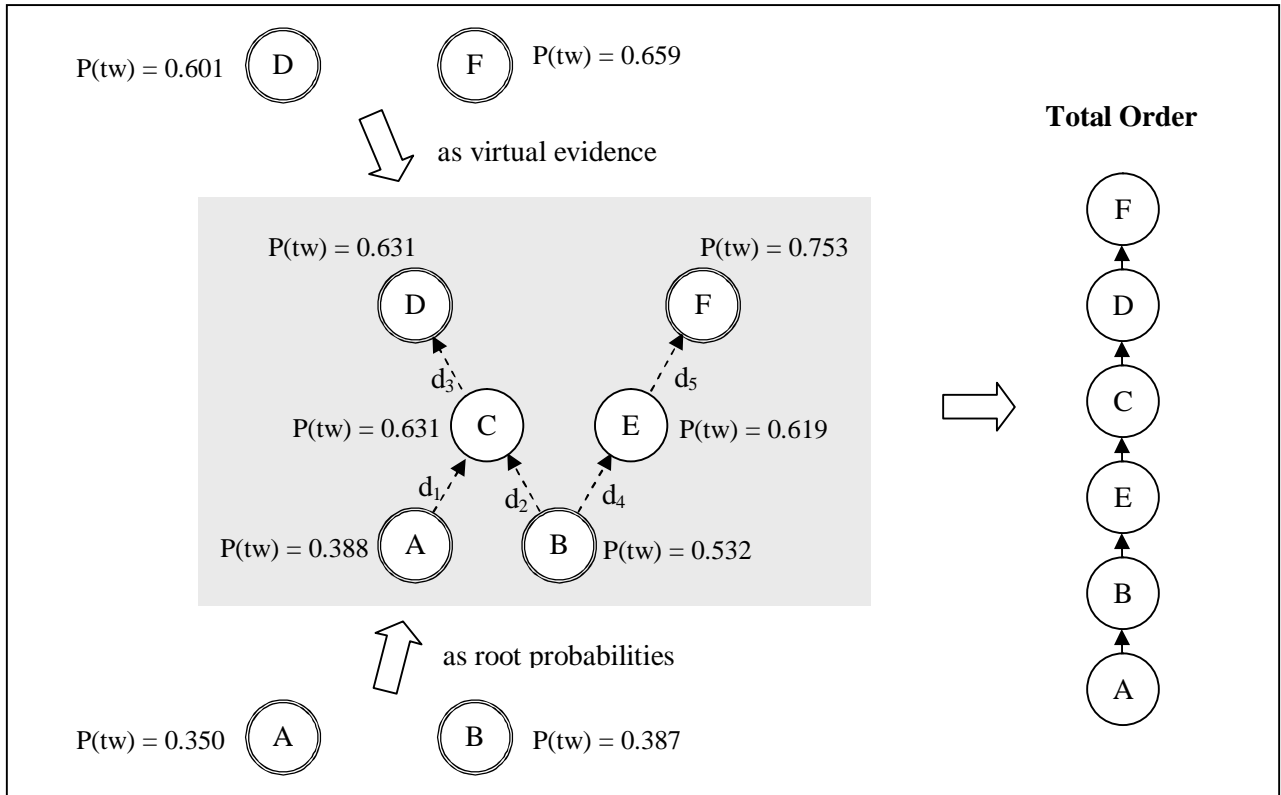


Figure 31: Last steps to compute the Total Order.

The following Algorithm 7 defines the determination of the total order with the input of the two BNs (Backward and Forward Graph) generated by Algorithm 6.

ALGORITHM 7: COMPUTETOTALORDER
Input: 2 BNs (1 x Forward Graph (FG), 1 x Backward Graph (BG))
Output: Total Order of set of entities E

1	Evaluate(FG)	∅ using interference algorithm
2	Evaluate(BG)	
3	BN ← create graph using structure and CPTs of FG	
4	BN.root_probabilities[] ← BG.local_sinks[]	∅ initialize BN's root probabilities
5	Evaluate(BN)	
6	BN.local_sinks[] ← FG.local_sinks[]	∅ insert virtual evidence
7	Update(BN)	∅ using interference algorithm
8	Sort(BN.E[], asc)	∅ sort entities according to their beliefs

Algorithm 7: Compute Total Order

This concludes the determination of the total order. In the next chapter a case scenario is given. It illustrates how the total order can be used to finally assess the trustworthiness of an entity.

10 Application Example

This chapter describes a scenario which will illustrate the application of the introduced model.

Assume Carol wants to buy a book of her interest and as a modern internet user she decides to buy it online. She finds that the book is offered by a number of online stores, sellers at Amazon's marketplace and sellers at eBay.¹⁷ However, only three of them (referred to as Seller S_1 , S_2 and S_3) offer it for a reasonable price. Due to increasing fraud in online transactions, Carol is unsure from which of the online sellers she should buy the book.

Seller S_1 acts over Amazon's marketplace. Amazon's marketplace is a platform where anybody can sell books. After a transaction, the buyer can rate the seller by giving a rating between 1 and 5, where 1 represents a "negative feedback", 3 a "neutral feedback" and 5 a "positive feedback." The final rating for a seller reflects the summarized feedbacks given over the last 12 months and is presented by a real number between 1.0 and 5.0.

Seller S_2 offers the book on eBay. eBay is an online auction platform. Similar to Amazon, after a completed transaction the buyer can rate how successful a transaction was by giving a "positive", "neutral" or "negative" feedback. The final rating for a seller reflects the percentage of his or her received positive feedbacks.

Seller S_3 runs a small online book shop. Ratings for online shops and services can be found, for example, on websites summarizing customer reports.¹⁸ In this example, a rating system is assumed where a rating between 0 "very bad" to 5 "very good" for a certain shop can be given. The final rating for a shop results in the average value of all ratings cast for a shop.

To evaluate which of the three sellers is most trustworthy, Carol decides to build three hierarchical relation graphs.

The first graph G_{Amazon} is based on all sellers on Amazon's marketplace that offer her desired book. She uses a seller's transaction feedback for the hierarchical ordering.

The second graph G_{eBay} is based on all current sellers on eBay offering the book. Also here, Carol uses a seller's percentage of positive feedbacks to determine the hierarchical order among them.

The third graph G_{shops} is the hierarchical ordered relation graph based on a number of book shops. Carol found them rated on a customer report website, along with the shop from which she is considering buying the book.

The three graphs with all selected sellers and shops are shown in Figure 32. Every node shows its rating beneath it. The three sellers considered for purchase are marked by a double-lined node.

¹⁷ Amazon is a registered trademark of Amazon.com, Inc.; eBay is a registered trademark of eBay Inc.

¹⁸ An in Europe well established customer report side is, for instance, www.ciao.com.

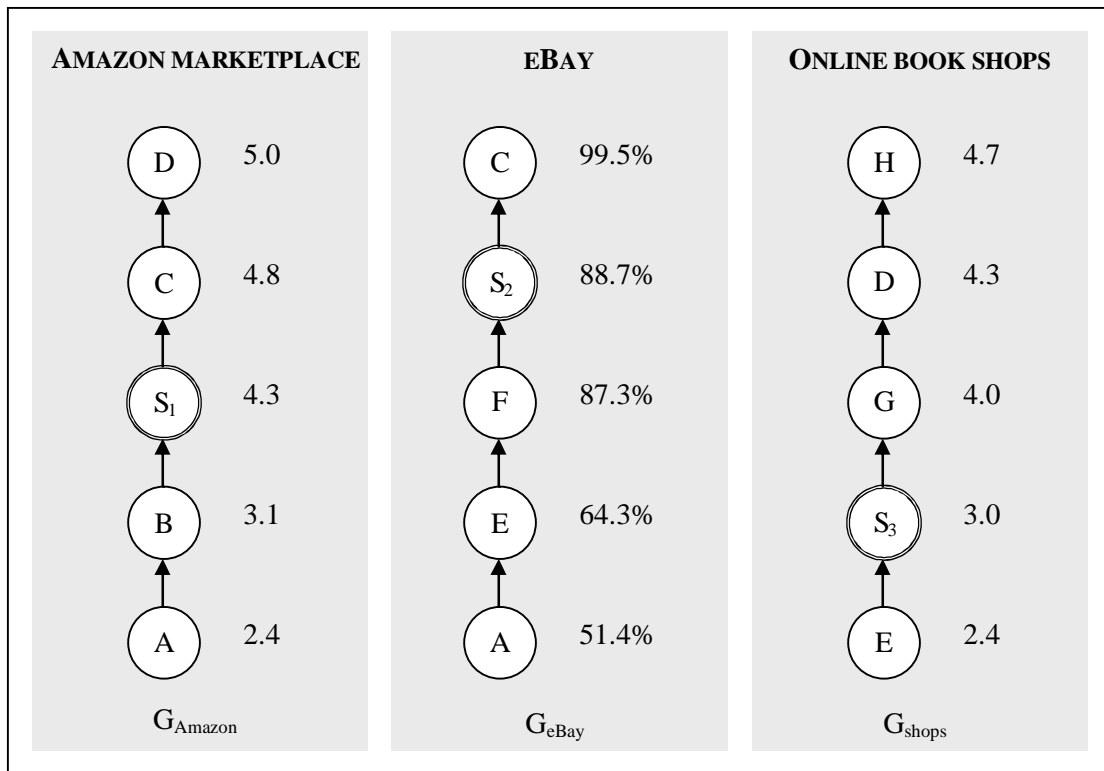


Figure 32: Three hierarchical relation graphs.

The next step is to calculate the distances using Algorithm 2. This is followed by a complete merge of the three graphs to unify the considered sellers into one relation graph.

A merge of the three given relation graphs is only realizable because some shops or sellers are operating on more than one platform. For instance, seller "A" sells his books on Amazon marketplace as well as on the eBay platform, or seller "D" sells on Amazon marketplace and also has his own online shop.

Since there are more than two graphs involved at this point, Algorithm 5 introduced in chapter 7.3.1 is used for the graph merging process. This algorithm allows the merging of several graphs in one step.

The merged graph with all computed distances is shown in the following Figure 33.

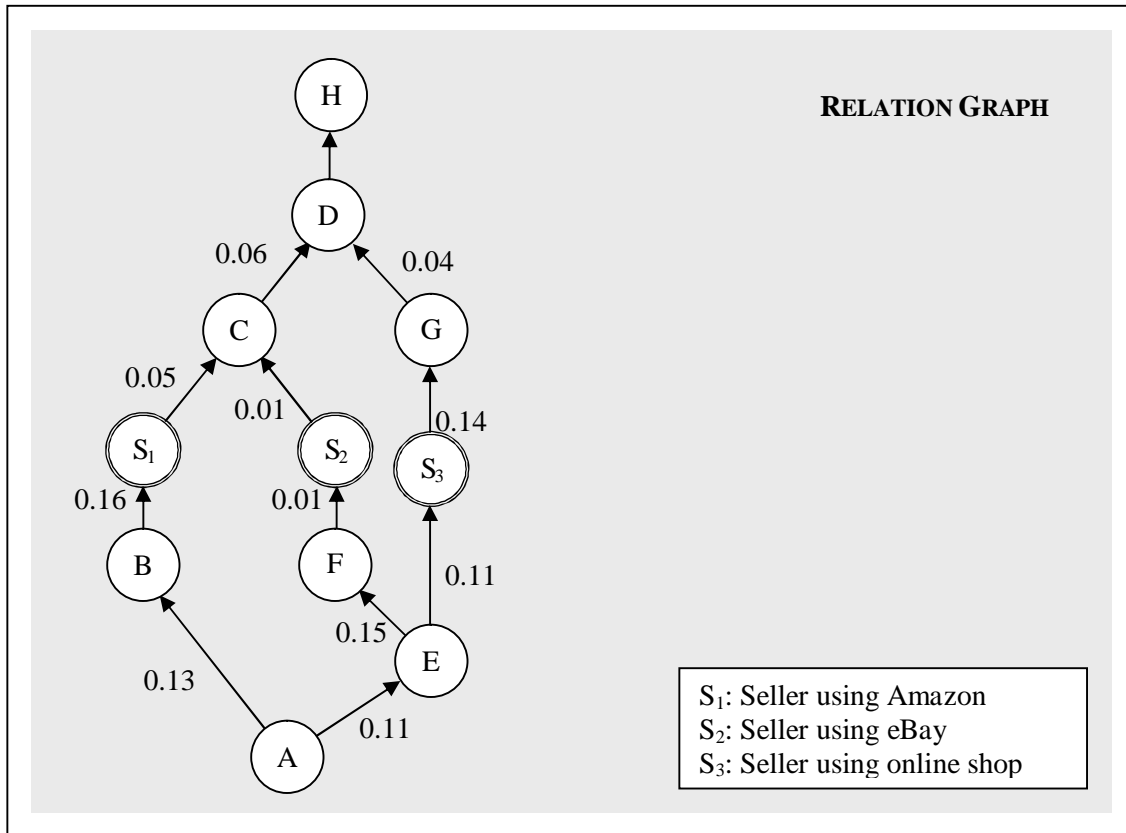


Figure 33: Merged relation graph.

After the merge process, the relation graph might contain cyclic parts. Algorithm 4 handles cyclic parts by collapsing them. The result is a directed acyclic graph. Since Carol's merged relation graph has no cyclic parts it is already in its desired structure for further processing.

The next step is to form the Bayesian network structures. In particular, the Backward and the Forward Graph including their CPTs must be built. Algorithm 6 transfers the relation graph into the two BNs. To calculate the conditional probability values for the CPTs the formulas introduced in chapter 8.3.1 and 8.3.2 are used. The formulas (8.1) and (8.2) are used to compute the CPTs for nodes with one predecessor. The (8.6) and (8.8) are used to populate CPTs for nodes with multiple predecessors.

Using the structures of both BNs (Backward and Forward Graph), the total order of all given entities (sellers in this case) are computed. This will give Carol the needed information as to which of her three selected sellers is the most trustworthy. Algorithm 7 lists the final steps to compute the total order.

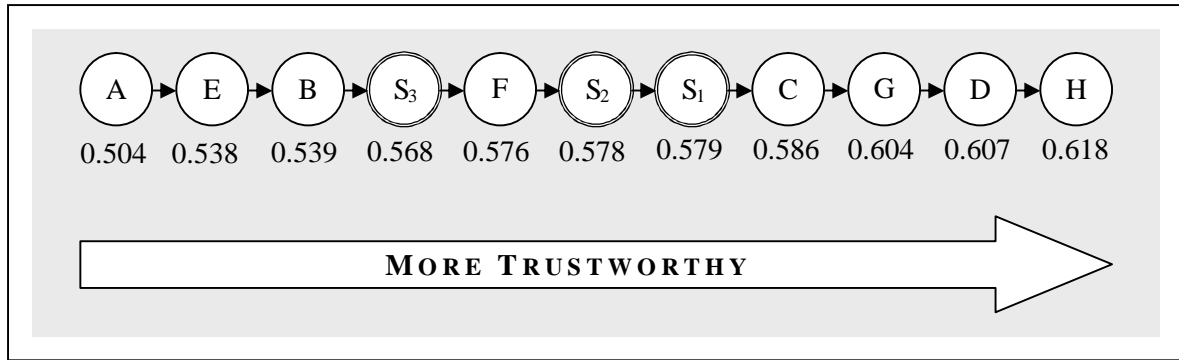


Figure 34: The computed Total Order of all given sellers according to their trustworthiness.

The results of Algorithm 7 are the belief values for every node. In ordered sequence they give the total order of the sellers. Figure 34 shows all nodes in ascending order along with their computed values. The values show that some of the sellers can almost be considered equally trustworthy (e.g. 'E' and 'B').

For Carol the result shows that according to the given information by the input relation graphs, seller S_1 or S_2 might be the right choice.

11 Review and Future Work

This thesis presents a model of trust which allows the assessment of the trustworthiness of a new or hitherto unknown entity. For this purpose, the model uses recommendations from cooperative authorities.

Unlike other trust models, the model in this work does not consider each entity individually, but determines its trustworthiness based on information about its trustworthiness in relation to other, known entities. This allows utilizing recommendations from various authorities for the process of assessing an entity, even if their methods of assessing an entity's degree of trust are incomparable among each other.

For the evaluation of relations, the model uses the structure and inference algorithms of Bayesian Networks. The Bayesian Networks are commonly used as probability models for decision making. In this thesis, as an already established model for processing probabilities, they make it possible for one to consider all relations relevant to an entity, within a connected network and thus in a total context.

By this means one gets an assessment of an entity's trustworthiness based on all relations between the individual entities involved.

To evaluate the ability of the presented model, it was tested. In the process, individual, experimentally derived relation graphs with various structures were used. The input values were chosen in such a way that the results were known in advance. Thus it was possible to compare the known results with those gained by using the model.

On the whole, the expected results were confirmed. However, there have been cases (in different networks) when comparing the expected total order with the resulting total order, differences occurred. The essentials shall be summarized in an example.

Figure 35 shows an example network, illustrating the expected order as well as the resulting total order. The values shown in the graph reflect the beliefs held for each entity. They are used to calculate the distances between every two entities connected to each other.

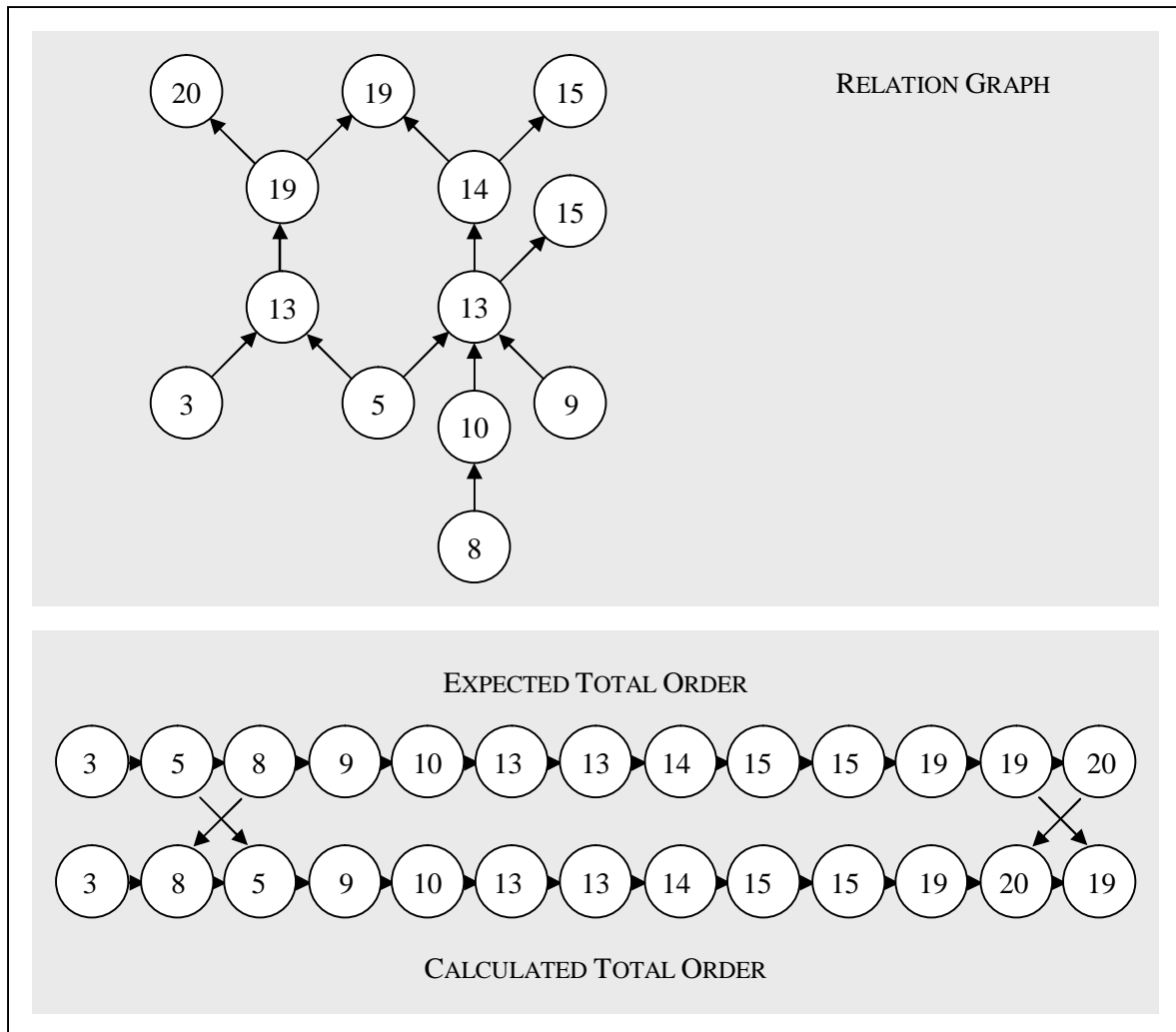


Figure 35: Example network, illustrating the expected and computed total order.

The graphic illustrates the mentioned differences. They are reflected by the changes regarding an entity's place if the calculated total order is compared with the expected total order. This introduces a certain aspect of uncertainty in the results.

Due to the small number of trials carried out so far, it is, however, not possible to finally assess the extent of the uncertainty. The assessment of the extent of differences which possibly occur in a network could be analysed by carrying out further trials. A random generation of the structure of the relation graph as well as a random number-generation of the input values would be means to proceed.

As perspectives and suggestions for a continuation of this work, two major aspects may be significant for further research:

- a) Developing a simulation environment for a systematic trial of the model

- b) A further development of the model may take the trustworthiness of the authorities – and by this means, of their recommendations – into consideration. It is conceivable to that the procedure developed and presented in this work might be employed in the assessment of an authority's trustworthiness as well.

12 Conclusions

In distributed computer systems, trust plays an increasing role. With the expansion of the internet in particular, it appears increasingly important to be able to assess the degree of trustworthiness of potential partners of interaction (or entities in general).

The thesis in hand developed a model for the assessment of the trustworthiness of hitherto unknown entities by means of utilizing recommendations from cooperative authorities. The model allows this especially in situations when the enquired authorities are using different trust classifications.

In analysing the trustworthiness of an entity, Bayesian Networks are used to model the entity's dependencies within the network and to calculate the trustworthiness by means of inference algorithms.

In the model's field of application, for instance in the 'assessment of an entity's reliability in the execution of transactions', the model as presented in this thesis may make a contribution to choosing the appropriate partner of interaction by providing a method to assess the entity according to its ability for a certain task.

In general provides the model which is developed on a scientific basis also a very good foundation for further research activities to determine the degree of trustworthiness of an entity by using Bayesian Networks.

13 References

- [Abdul-Rahman, 2000] Abdul-Rahman, A., Hailes, S.: *Supporting Trust in Virtual Communities*. In IEEE 2000 Proc. of the 33rd Hawaii International Conference on System Sciences, 2000
- [Charniak, 1989] Charniak, E., Goldman, R.: *A Semantics for Probabilistic Quantifier-Free First-Order Languages with Particular Application to Story Understanding*. In Proc. of 11th International Joint Conference on Artificial Intelligence, pp. 1074-1079, 1989.
- [Charniak, 1991] Charniak, E.: *Bayesian Networks without Tears*. In AI Magazine 12 (4), pp. 50-63, 1991
- [Chartrand, 1986] Chartrand, G., Lesniak, L.: *Graphs & Digraphs, Second Edition*. Wadsworth, Inc., California, 1986
- [Cormen, 2001] Cormen, Thomas H., et. al.: *Introduction to Algorithms, Second Edition*. The MIT Press, Massachusetts, 2001
- [Heckerman, 1992] Heckerman, D., Horvitz, E., Nathwani, B.: *Towards Normative Expert Systems: Part I. The PATHFINDER Project*. Methods of Information in Medicine, pp. 90-105, June 1992
- [Heckerman, 1995] Heckerman, D., Mamdani, A., Wellman, M. P.: *Real-world applications of Bayesian networks*. Communications of the ACM, Volume 38, Issue 3, pp. 24-26, Mar. 1995
- [Henrion, 1989] Henrion, M.: *Some Practical Issues in Constructing Belief Networks*. In Uncertainty in Artificial Intelligence 3, pp. 161-174. North Holland, New York, 1989.
- [Jøsang, 1996] Jøsang, A.: The right type of trust for distributed systems. Proceedings of the 1996 workshop on New security paradigms, pages 119-131, 1996.
- [Lauritzen, 1990] Lauritzen, S. L., Spiegelhalter, D. J.: *Local computations with probabilities on graphical structures and their application to expert systems*. In Readings in uncertain reasoning, pp. 415-448, 1990.
- [Manber, 1989] Manber, Udi: *Introduction to Algorithms: A Creative Approach*. Addison-Wesley, Massachusetts, 1989
- [McKnight, 1996] McKnight, D., Chervany N.: *Meanings of Trust*. Technical Report MISRC 96-04, Management Information Systems Research Center, University of Minnesota, 1996

- [Morawski, 1989] Morawski, P.: *Understanding Bayesian Belief Networks*. In AI Expert, Volume 4, pp: 44-48, May 1989
- [Neapolitan, 2004] Neapolitan, R. E.: *Learning Bayesian Networks*. Pearson Prentice Hall, 2004
- [Pearl, 1988] Pearl, J.: *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.
- [Shi, 2004] Shi, J., Bochmann, G., and Adams, C.: *A Trust Model with Statistical Foundation*, Workshop on Formal Aspects in Security and Trust (FAST), 18th IFIP World Computer Congress 2004, pages 169-181, August 2004
- [Wang, 2003] Wang, Y., Vassileva, J.: *Bayesian Network-Based Trust Model*. In IEEE 2003 Proc. of the International Conference on Web Intelligence, 2003
- [Yahalom, 1993] Yahalom, R., Klein, B., and Beth, Th.: *Trust Relationships in Secure Systems - A Distributed Authentication Perspective*. In Proc. 1993 IEEE Computer Society Symposium on Research in Security and Privacy, pages 150-164, 1993
- [Yu, 2002] Yu, B., Munindar, P. Singh: *An Evidential Model of Distributed Reputation Management*. In Proceedings of the first international joint conference on Autonomous agents and multiagent systems, pages 294-301, July 2002