SYSTEMATIC GENERATION OF BAYESIAN NETWORKS FROM SYSTEMS SPECIFICATIONS

Michael Borth, Hermann von Hasseln
DaimlerChrysler Research and Technology
Michael.Borth@DaimlerChrysler.com, Hermann.v.Hasseln@DaimlerChrysler.com

Abstract: The usefulness of Bayesian network technology for expert-systems for diagnosis, prediction, and analysis of complex technical systems has been shown by several examples in the past. Yet, diagnosis systems using Bayesian networks are still not being deployed on an industrial scale. One reason for this is that it is seldom feasible to generate networks for thousands of systems either by manual construction or by learning from data.
In this paper, we present a systematic approach for the generation of Bayesian networks for technical systems which addresses this issue. We use existing system specifications as input for a domain-dependent translation process that results in networks which fulfil our requirements for model-based diagnosis and system analysis. Theoretical considerations and experiments show that the quality of the networks in terms of correctness and consistency depends solely on the specifications and translation rules and not on learning parameters or human factors. We can significantly reduce time and effort required for the generation of Bayesian networks by employing a rules-based expert system for generation, assembly and reuse of components. The resulting semi-automatic process meets the major requirements for industrial employment and helps to open up additional application scenarios for expert systems based on Bayesian networks.

Key words: knowledge-based design methodologies, Bayesian networks, knowledge acquisition, component reuse, expert-systems for diagnosis

1. INTRODUCTION

Given the well-known advantages of Bayesian networks for diagnosis systems, we at DaimlerChrysler Research and Technology are naturally considering their use for next-generation systems with predictive diagnosis and system analysis functionality. Recently, we have seen extensive work on

---

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35602-0_35
M. A. Musen et al. (eds.), *Intelligent Information Processing*

modelling techniques that allow the construction of Bayesian networks for complex systems and their application in several domains, including ours: dynamic technical systems of high complexity. We can safely assume that it is possible to construct a useful Bayesian network for a given system, but this is immaterial as long as the resources needed to do so for all required systems are unreasonable high. This is clearly the case for the existing procedures, considering the thousands of (sub-) systems and their interworking as prevalent at DaimlerChrysler.

This determines the starting point for our paper. After we describe the requirements for the Bayesian networks and the processes for their generation, which result from our application scenario, we briefly review the existing procedures for the generation of networks. This defines the need for a new process with improved efficiency. We introduce our solution and propose a system architecture for its realisation. Finally, experiments and theoretical considerations illustrate the advantages of our approach and point to open research questions.

2. BAYESIAN NETWORKS FOR ADVANCED DIAGNOSIS SYSTEMS

Diagnosis has the purpose to infer the state of a system and often to provide the information needed to repair it. Thus, diagnosis is part of a more general process that determines the requirements for the diagnosis system. The process of building a diagnosis system is for the most part independent of this and has its own requirements. In this chapter, we introduce requirements of both types relevant to our work and cite their consequences.

2.1 Advanced Diagnosis Systems

DaimlerChrysler uses diagnostics for highly complex systems in the performance of a variety of tasks. These include the monitoring of systems in operation in order to detect failures, predictive diagnosis to enable maintenance prior to the failure or diagnosis and decision support during repair. These tasks result in several requirements for a diagnosis system [1]:

- Accomplishment of the diagnostic task.
- Efficiency: minimal computational costs.
- Reasonable diagnostic process.
- Utilisation of different sources of information.

Additionally, we expect an expert system to help us analyse the technical system itself. The identification of critical components or possible compensational behaviour is necessary to enhance product quality.

2.2 Bayesian Networks for Diagnosis

Even though no diagnostic technique is best geared to solve all diagnosis tasks with respect to all criteria [2], we will focus solely on Bayesian networks in this paper. Their general suitability as core technology for expert systems for diagnosis is well documented (see e.g. [3], [4], [5]) and needs no further illustration. However, there are several approaches on how to use Bayesian networks in the best possible way, and our design decisions must reflect the requirements of our application scenario.

First, we have to decide between model-based or symptom-based diagnosis systems. This determines what we have to model: which variables, states, etc. We use model-based systems, because they account for combinations of faults and compensational behaviour and allow predictive diagnosis and system analysis. Secondly, we have to decide how to build the Bayesian networks. The complexity of our technical systems and their modular, component-based assembly strongly suggests the use of object-oriented (OO) Bayesian networks [6].

2.3 Network Generation

We have a number of additional requirements for the network generation. Most importantly, this process must be highly efficient, since the use of Bayesian networks on an industrial scale requires networks for thousands of systems and their individual components.

This requirement alone proscribes the manual construction of the networks using knowledge engineering, since this approach requires a combined effort of knowledge engineers, diagnosis experts, and system experts. There are several methodologies that reduce the manual effort, e.g. by supporting the construction of models from a knowledge base [7] or by assisting the experts with a standardized process. Yet, we found that it is simply not feasible to depend on human experts in our scenario, especially since the interactions between different components of a technical system can be so complex that an expert is unable to understand and use all the available information. This loss of information will detract from the quality of the diagnosis, as will any mistake made during the construction of a network. Since we cannot rely on humans to work without errors, we have to test the constructed networks with respect to all the sensible system states and modes of operation. This process is extremely expensive and may even not be

feasible. Learning the networks from data is also impractical. The necessary data simply does not exist for most systems, especially not prior to the real-life application. Even if we assume the existence of adequate data, we cannot rely on models generated by machine learning without extensive testing (see above). This adds to the overall effort for data preparation and optimisation of learning parameters. The resulting amount of necessary time and manpower is not acceptable.

To summarize, we need a new approach for network generation that results in object-oriented Bayesian networks which fulfil our requirements for model-based diagnosis and system analysis. The new process has to be largely independent of manual work to guarantee efficiency and faultlessness. Thus, it must be semi-automatic at the least and has to use inputs other than those of human experts.

3. SYSTEMANTIC GENERATION FROM SYSTEM SPECIFICATIONS

The specifications of the systems are a natural choice for such an input. Embedded in the process of system design and construction, they provide the necessary information in high quality. This chapter describes how we turn this information into Bayesian networks suitable for diagnosis systems.

3.1 Technical Specifications

Specifications come in many forms and typically consist of several documents, including diagrams, simulation models, formula, data, etc. They describe at least the following aspects:

- Design information: components and their composition.
- Functional information: functional effects, operation of the system.

Most likely, additional data is available about the quality of the components (e.g. the mean time between failures) or can be obtained from domain experts. Since Bayesian networks are insensitive to imprecision in the probabilities [8], small quantitative errors in this process pose no real problems. Experience shows that a suitable question strategy prevents qualitative errors [9]. Knowledge about the domains of the technical systems complements the specifications. For example, a specification of an electric system will not specify that a faulty component breaks the electric circuit, since this is a fundamental fact based on the laws of physics.

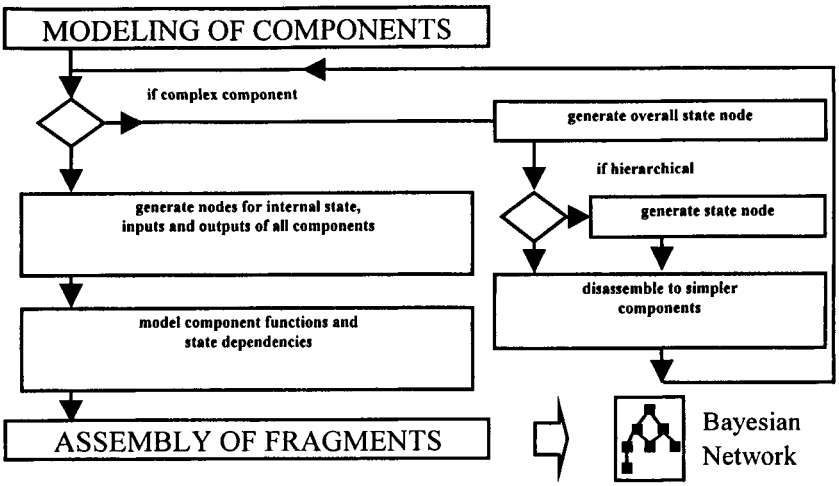

Figure 1: Systematic generation

3.2 Systematic Generation

Whenever a system specification conforms to the requirement that it comprises the information set out above, we can build a Bayesian network using the process outlined in figure 1.

The process results in a Bayesian network that is a valid description of the system. We neither omit any relevant information, nor add anything that is not in the specification. Thus, the entirety of the components, their functionalities, and interactions are represented.

3.2.1 Translation Rules

The translation of the relevant elements of the specification into network fragments and the subsequent assembly of the fragments follow rules which determine how the available information must be used. The generation of such sets of domain-dependent rules is a fundamental part of our approach. Their use must ensure the consistent and correct translation of the specifications. Our approach to guarantee this is to use only rules which represent principles of object-oriented design as well as physical laws and facts to model component functionality, interaction and assembly.

Every domain has (surprisingly few) different types of components, like, for the electric domain, conductor, capacitors, transistors, ground, etc. Each has a specific functionality. We establish rules for the translation of each component type and its functionality into a fragment of a Bayesian network according to the process laid out in the next section. Additionally, we

formulate rules for the basic principles which determine the interaction between components, e.g. Kirchhoff's laws. These rules describe a simple pattern matching process that identifies what is connected to what, adopts the principle to the specific situation and generates the information needed for the translation.

3.2.2 Modelling of Components

We start the process by generating OO network fragments which represent the components of the system. The diagnostic task defines the appropriate level of detail, which, in turn, determines all subsequent steps.

A component itself can consist of several components. We model each sub-component of such complex components separately and generate a node representing the overall state of the complex component which is determined by the sub-components. A complex component is called a hierarchical component, if it has an additional state variable, which determines the states of all sub-components. We generate the node for such a state variable, too and follow this process recursively until we have broken down all complex components to simple components.

We determine the possible internal states of the components and their inputs and outputs. Inputs and outputs are interfaces between components or components and the environment. They transfer either physical quantities or information. The range of possible values of an interface defines the state-space of the according network nodes which we now construct. We generate nodes for the internal states and enter the apriori probabilities for all state variables, except for the overall sates.

Next, we model the state dependent functionality of the components, which are deterministic unless the specification is not detailed enough or the system includes random elements. The conditional probability distribution defined by $P(O=o|I_1=i_1,\ldots,I_n=i_n) = 1$ for $F(i_1,\ldots i_n) = o$, else $P = 0$ describes the function $F = F(I_1,\ldots, I_n) \rightarrow O$ for the input variables $I_1,\ldots, I_n$ and the output variable O [10]. Consequently, we insert connections from all the input nodes and from the state node to the output node and quantify the connections with the appropriate distribution.

Overall state variables of complex components and variables in functional groups directly depend on other state variables. We connect these nodes in accordance with existing causalities.

3.2.3 Assembly of Fragments

The design information in the specification describes how the components are connected to each other. These connections between outputs and inputs simply transfer physical effects or information. Accordingly, we generate 1:1 connections from output nodes to the appropriate input nodes. These connections guarantee that the nodes are in the same state.

We use the same method to connect the input and output nodes which represent the system environment with the components. This step completes the generation of the Bayesian network.

3.3 Modelling of Time

Whenever a specification does not rely solely on static information but describes dynamic effects, we have to model the flow of time. The modeling steps described above already account for this. They do not require that all the modeled entities belong to the same time-frame or that only static values are modeled and not the change of values. Thus, we can use either method to model the dynamics of the system. Depending on the specification we describe the dynamics in time-frames with conventional dynamic Bayesian networks (see [11], [12]) or changes over time, for which we use Arroyo's approach [5].

3.4 Properties of the Process

During the development of the process, we had to analyze several theoretical issues. The most important aspect, for the purpose of this paper, is that we can guarantee that the process always results in a valid Bayesian network. For example, we can show that the process never generates circles (which are not allowed in Bayesian networks) given that the flow of time was modelled in a cored way. Another important aspect is the determinism of the process. Once the required level of detail and the appropriate translation rules are selected, we face no additional design decisions. The tasks in every single step result directly from the information available, the rules, and the results of the preceding steps. The formal proof for these properties is outside the scope of this paper.

4. GENERATION OF NETWORKS BY AN AUTOMATED SYSTEM

The main purpose of the new process is that it be used in an automated system. We propose the system architecture set out below (figure 2).

4.1 Components as Basic Elements

Given the specifications and the required level of detail, the system starts by extracting all components and the information available about them. We use these components as basic elements for all successive steps. To avoid duplicated work, we store all the components in a library, realized as a database. If a component has already been modelled, the component library provides the respective network fragment.

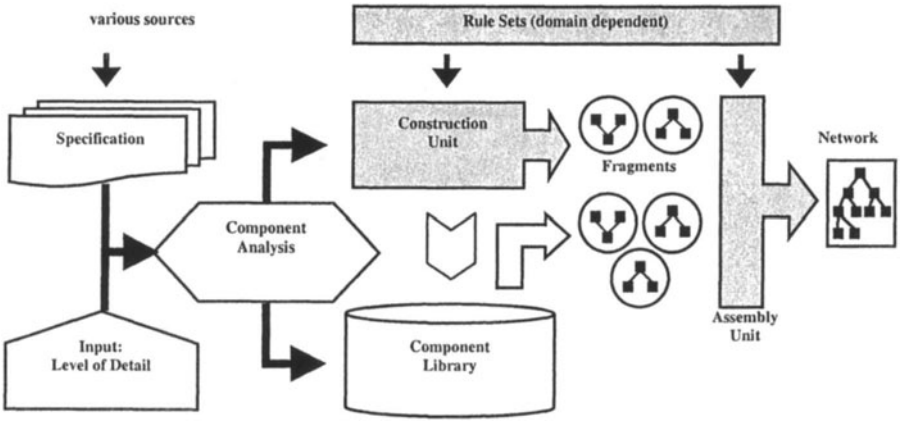

Figure 2: System for network generation (overview)

4.2 Rule-based System for Modelling and Assembly

A rule-based expert system is the heart of the system. It consists of a rule base, the construction unit, and the assembly unit. The rule base holds all translation rule sets and provides the other units with the rules for their tasks.

The construction unit builds the network fragments for new components. It can adapt fragments of known components, use templates for typical ones or construct solely on the basis of rules. The construction unit stores new fragments in the library, together with the component information and details about the construction process (useful for the adaptation mentioned above).

The assembly unit connects all the fragments in accordance to the specification. It constructs both the necessary connections between input and

output nodes and the direct dependencies between state variables. The result is an operational network, which maps all information available to the construction system.

4.3 Optimisation and Deployment

It is likely that the resulting network will benefit from optimisation, e.g. by a reduction of the number of nodes or simplifications. We can do this without any loss of functionality by eliminating redundant information and the use of common modeling tricks, as noisy-or gates. For example, we can merge nodes with 1:1 connections, which only transfer information. The basic optimisation procedures follow a small set of simple rules. This enables us to implement the optimization as a rule-based expert system, too.

Finally, we transcribe the generated network into a form suitable for the intended application, e.g. diagnosis software or a hardware realization. This is a standardized process which needs no additional input. We can generate data files for network modelling tools, precompiled networks for an inference engine, or code for a hardware realization [13].

5. EXPERIMENTAL EVALUATION

We tested the systematic generation of Bayesian networks from specifications to evaluate both the process and the resulting networks.

The experiments were representative for a wide range of both digital and analog technical systems and covered different types of diagnostic problems. We decided on the smallest replaceable units as an appropriate level of detail for modelling and translated the system specifications accordingly with domain dependent rules, which we formulated in advance. We tested the Bayesian networks on a predefined set of diagnostic tasks.

5.1 Examples

We will now present some details of our experiments. A complete description of even a single experiment is outside the scope of this paper.

We chose an electrical systems composed of several circuits with common ground as an example for digital systems. The system accepts user input at the power amplifiers, where it also provides diagnostic information. We had to derive the state of all components from the combination of the error codes and the user input. Thus, we had to deal with hidden mistakes, compensational behaviour and feedback circles. To solve this challenge, we

had to exploit that effects in electrical circuits are non-directional. Our translation rules for digital electronics account for this by constructing a bi-directional flow of information between the power amplifiers and ground. The electric system also demonstrated that it is indeed possible to work on different aspects of components separately and independently, which negates the need to anticipate every possible combination of functionality. For example, we built a plug as a hierarchical component, where one of the pins is also part of a functional group. These two aspects of the pin correspond to different types of effects, which we model in different nodes.

Our second example, a suspension system (an analogous damped oscillating system where the components interwork) demonstrates another issue: cyclical influences. They require the modelling of time, since the information available at a single moment is insufficient for a diagnosis. In the system specifications, differential equations describe changes over time, so we model the state changes, as set out in section 3. However, we found that both approaches break the cycle where the components influence each other, since any effect occurs later than the cause.

5.2 Results

Our experiments show that systematic generation of Bayesian networks works extremely well given suitable specifications and translation rules, which are the main influence factors for the quality of the results. The modelling of both components and the interactions between them can be done in an automated process by a rule-based expert system.

Our experiments verify that specifications are the ideal input for constructing a representation suitable for system analysis and model-based diagnosis. The resulting diagnosis systems outperformed alternative systems, especially for complex problems, such as multiple faults with compensational behavior. The system analysis works equally well, e.g. the identification of critical components or dependency analysis as described in [14]. The value of different pieces of information for the diagnosis process can be inferred using sensitivity analysis [15], but we have not looked into this in detail thus far.

These positive results show that the translation rules used performed well. Our experiments confirm that the rules must be adequate for both the domain and the level of detail. Hence, we have to construct a set of rules for every combination that we anticipate (a small number). Currently, we select the correct rule set for a given system manually, but we aim at achieving an automatic selection via classification or case-based reasoning.

The correctness of the rules is not self-evident, since they were manually constructed and thus prone to error. As the rules represent basic principles, such as physical laws, the generation of an incorrect rule is unlikely. Far

more problematic are incomplete rule sets, which fail to cover existing situations. Such mistakes crop up during the translation process and force a revision of the rule set. A related problem is that the rules are not unequivocal. Often, there are several approaches for the construction of a rule set. Each approach leads to consistent rules, but the resulting networks may differ. It is essential to realize the impact of these modelling decisions on the diagnostic system. The problem is that we cannot deduce them from the system specification. We need additional information on how to interpret the system behaviour. This best illustrates the importance of the rule set. We can anticipate and eliminate such problems if we involve experts for diagnosis during the rule generation.

6. DISCUSSION AND DIRECTIONS FOR FUTURE RESEARCH

Theoretical considerations, exemplary implementation, and experimental evaluation show that our approach for a systematic generation of Bayesian networks from specifications of technical systems works. The resulting networks fulfil the requirements for both diagnosis and system analysis and outperform most alternative diagnosis systems. The process for network generation can be used manually or realized within a semi-automatic system. Such systems meet our requirements for generating Bayesian networks on an industrial scale. In this context, we have to consider the effort needed to build such a system. Even though we see no alternative within our application scenario, others may decide that the advantages of an extensive automation do not prevail over the costs of the necessary components.

We hope that future research will help to reduce these costs. For example, it should be possible to simplify the rule generation and, thus, the building of the expert system by exploiting the hierarchical dependencies between different levels of detail. Another open research question is network optimisation. Currently, we use only elementary mechanisms to simplify the networks. We expect better results from more complex approaches, e.g. pattern-based search-and-replace operations.

7. CONCLUSION

The main contribution of this work is the introduction of a systematic method for the efficient and accurate generation of Bayesian networks for complex technical systems. An implementation of this process in a semi-

automatic system allows the generation of networks even for high numbers of systems. This, in turn, allows the industrial employment of the process, since only minimal efforts are required from human experts. We are confident that this will help to open up additional application scenarios for knowledge-based expert systems using Bayesian networks.

REFERENCES

1. A. Heinzelmann: Produktintegrierte Diagnose komplexer mobiler Systeme, Ph.D. Thesis, University of Paderborn, 1998.
2. S. Iwanowski: Technical Diagnosis - Chances and Limitations. DaimlerChrysler Technical Report, 1997.
3. D. Heckerman, M. Wellman: Real World Applications of Bayesian Networks. In *Communications of the ACM*, 38(3), 25-26, 1995.
4. C. Skaaning, F.V. Jensen, U. Kjaerulff: Printing System Diagnosis - A Bayesian Network Application, In *Proceedings of the 9th International Workshop on Principles of Diagnosis*, 1998.
5. G. Arroyo-Figueroa, L. Sucar: A Temporal Bayesian Network for Diagnosis and Prediction. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, 1999.
6. D. Koller, A. Pfeffer: Object-oriented Bayesian Networks. In *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence*, 1997.
7. K. Laskey, S. Mahoney: Network Fragments: Representing Knowledge for Construction Probabilistic Models Networks. In *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence*, 1997.
8. M. Henrion: Why is diagnosis using belief networks insensitive to imprecision in probabilities?. In *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence*, 1996.
9. C. Skaaning: Knowledge Acquisition for Bayesian Networks in Diagnosis, Hewlett-Packard Research Report, 1998.
10. S. Srinivas: Building diagnostic models from functional schematics, Stanford University Report No. KSL 94-15, 1994.
11. U. Kjaerulff: A computational scheme for reasoning in dynamic probabilistic networks. In *Proceedings of the 8th Conference on Uncertainty in Artificial Intelligence*, 1992.
12. U. Lerner, R. Parr, D. Koller, G. Biswas: Bayesian Fault Detection and Diagnosis in Dynamic Systems. In *Proceedings of the 16th National Conference on Artificial Intelligence*, 1997.
13. A. Darwiche, G. Provan: Query DAGs. In *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence*, 1996.
14. L. Portinale, A. Bobbio: Bayesian Networks for Dependency Analysis. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, 1999.
15. K. Laskey: Sensitivity Analysis for Probability Assessments in Bayesian Networks. In *Proceedings of the 7th Conference on Uncertainty in Artificial Intelligence*, 1991.