Model-based Probabilistic Diagnostics of Cyber-Physical Systems

GUUS GRIEVINK, University of Twente, The Netherlands

With model-based diagnostics, a machine is modelled with the intention to trace back the cause of a particular machine failure. Where traditionally this is done with logical statements, a probabilistic approach can be used by translating a diagnostic problem into a Bayesian network.

The research in this paper expands on the work of the TNO's Embedded Systems Institute (ESI) which, among other things, researches the Bayesian approach to model-based diagnosis. In the Bayesian models of ESI, a uniform prior probability is assumed on the health states of components. However, this does probably not hold in practice, as some components might be more likely to fail than others.

The research tries to determine whether including probabilistic information reflecting the real-world scenario improves the diagnostic capabilities of Bayesian diagnostic models. A simple theoretical system consisting of water pipes is used to achieve this. A network containing probabilities of component failure reflecting the used data is compared to a uniform model. Experiments on the test data show that the adjusted model, on average, performs better than the uniform model. However, other observations indicate that the used structure for the Bayesian network might not be optimal for including probabilities of component failure from real-world data. Suggestions are made to mitigate these issues, which are up for future research.

Additional Key Words and Phrases: Model-based diagnosis, Cyber-physical systems, Consistency-based diagnosis, Bayesian diagnostic problem, Probabilistic inference

1 INTRODUCTION

Cyber-physical systems act both in the physical world as well as the software domain. A typical example of this is an industrial production machine equipped with sensors. In industries where such systems are used, these machines are desired to experience the least possible downtime. An important aspect of this is fault diagnostics: being able to trace back a malfunctioning to a specific component. As cyber-physical systems become more complex, so does the task of tracing back machine failures. Once a specific component fails, it often results in a chain reaction of other components behaving abnormally before the malfunctioning is detected [2]. Next to this, on-site technicians often do not have all the knowledge about the system to trace back the problem themselves [4].

To deal with these problems, several automated fault detection and diagnostic methods have been developed. Each at a different state of technical maturity [6]. For the purpose of this research, we will look at a hybrid model-based method which is used by TNO's Embedded Systems Institute (ESI) in collaboration with several industrial companies, such as Cannon and ASML [1, 2]. This approach uses a probabilistic approach to **model-based diagnostics** or **diagnosis** (**MBD**).

TScIT 37, July 8, 2022, Enschede, The Netherlands

The main idea behind model-based diagnostics is that before a particular machine or piece of equipment is employed in a production setting, most of the knowledge available to find faults or defects is design knowledge, e.g. about the structure and organisation of the machine and how it normally functions. Only much later data will become available about actual faults occurring in the practical deployment of the machine. Such data can be employed to improve a particular model used for diagnostics to reflect the frequency with which they occur. Thus, in the early state of model-based diagnostics, the emphasis will be on exploiting design knowledge. The traditional method used in this case is known as **consistency-based diagnostics** or **diagnosis** (CBD) [5]. Although CBD is usually seen as a kind of logical reasoning, it can be mapped to the multivariate probabilistic representation of Bayesian networks [9]. The advantage of Bayesian networks as a formalism of model-based diagnosis is that in principle uncertain, probabilistic knowledge about the occurrence of faults can be integrated and also learnt from data, which is the aim of the present research.

This paper first states the research question in section 2. In section 3, relevant literature is discussed. The methodology in section 4 describes the steps that were taken to answer the research question. Results from these experiments are described in section 5 and are followed by discussion and conclusions in section 6.

2 PROBLEM STATEMENT

In the work of Barbini et al. [2] at ESI(TNO), the input to our research, the prior probability of health status of each component has a uniform distribution. This means that the assumption is made that all components are equally likely to fail. However, this assumption is not likely to hold in practice, as not all components are equally robust. Information about the relative likeliness of failure of each component might have a beneficial effect on the diagnoses. This leads to the following research question:

Does taking into account probabilistic information reflecting the real-world situation of component failures increase the accuracy of diagnostic models?

In this paper, a first step will be taken to answer this research question. For this, a theoretical, relatively simple, system will be evaluated along with artificially generated data. Although the conclusions drawn from experiments with this system may not directly generalise to real systems, they may provide interesting insights on how to apply probabilistic MBD to cyber-physical systems in the industry. Next to this, a simple theoretical system may help to assess the methodology that is provided in this research. Insights can finetune the methodology so that it can be employed in a real-life scenario in the future to answer the research question in its entirety.

3 BACKGROUND

3.1 Consistency-based diagnosis

The concept of MBD was developed to solve the fault diagnostics problem in a structured way [8]. Where a **system** SYS is modelled

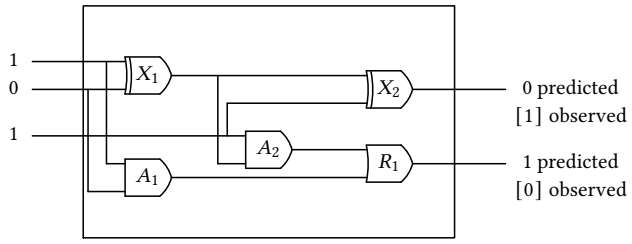


Fig. 1. Full adder with inputs and observed and predicted outputs. Here, Obs = $\{\text{in1}(X1) = 1, \text{in2}(X1) = 0, \text{in1}(A2) = 1, \text{out}(X2) = 1, \text{out}(R1) = 0\}$

as a **system description** SD and a list of **components** Comps. The system description defines the normal behaviour of components and how these components are connected in first-order logical sentences. In consistency-based diagnosis, given a specific input of the system, the output of the model is compared with the observed output of the system. A discrepancy between these two indicates a fault or defect in the system [5, 10].

A **diagnostic problem** can be seen as a system together with a **set of observation** Obs. A **diagnosis** D is a minimal set of components that, when behaving abnormally, explains the observation of a faulty system. Formally, a diagnosis D is defined as follows:

$$\text{SD} \cup \text{Obs} \cup \{\text{Ab}(c) \mid c \in D\} \cup \{\neg\text{Ab}(c) \mid c \in \text{Comps} - D\} \nvdash \bot \quad (1)$$

Where Ab is the abnormal predicate that indicates that a component c behaves abnormally (and thus ¬Ab indicates normal behaviour) and $\bot$ is false (the left-hand side of $\nvdash$ is consistent).

Example 3.1. Consider the logical circuit depicted in Figure 1, which represents a full adder, i.e. a circuit that can be used for the addition of two bits with carry-in and carry-out bits. This circuit consists of two AND gates ($A1$ and $A2$), one OR gate ($R1$) and two exclusive-OR (XOR) gates ($X1$ and $X2$); Comps = $\{A1, A2, X1, X2, R1\}$. The input and output symbols for the components c are denoted as $\text{in}(c)$ and $\text{out}(c)$.

The behaviour description SD, as provided in [10], consists of the following axioms:

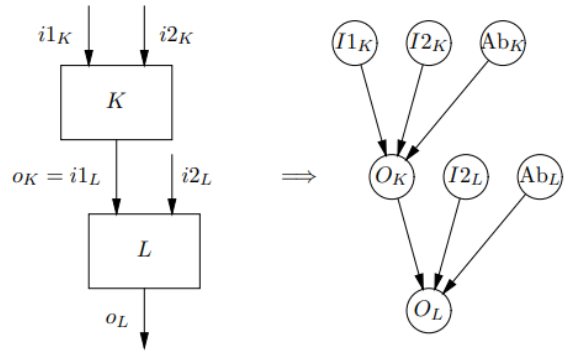
$$\begin{aligned}
\neg\text{Ab}(c) &\rightarrow \text{out}(c) = \text{and}(\text{in1}(c), \text{in2}(c)), \text{ for } c \in \{A1, A2\},\\
\neg\text{Ab}(c) &\rightarrow \text{out}(c) = \text{xor}(\text{in1}(c), \text{in2}(c)), \text{ for } c \in \{X1, X2\},\\
\neg\text{Ab}(c) &\rightarrow \text{out}(c) = \text{or}(\text{in1}(c), \text{in2}(c)), \text{ for } c = R1.
\end{aligned}$$

These logical rules describe the normal behaviour of each individual component (gate).

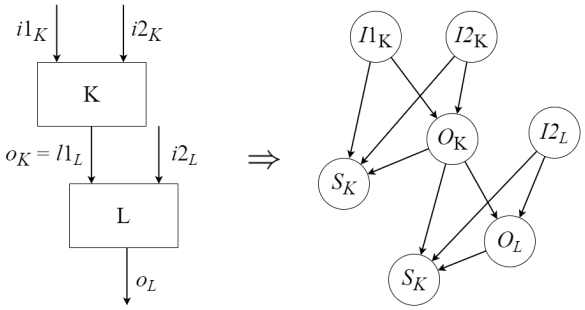
The component connections are described as follows:

$$\begin{aligned}
\text{out}(X1) &= \text{in2}(A2) & \text{out}(X1) &= \text{in1}(X2)\\
\text{out}(A2) &= \text{in1}(R1) & \text{in1}(A2) &= \text{in2}(X2)\\
\text{in1}(X1) &= \text{in1}(A1) & \text{in2}(X1) &= \text{in2}(A1)\ .\\
\text{out}(A1) &= \text{in2}(R1)
\end{aligned}$$

With the observations Obs as indicated in Figure 1 it is clear that when assuming the empty diagnosis, $D = \emptyset$ — all components are behaving normally — Equation (1) will give an inconsistency, as also indicated in the figure (predicted and observed outputs differ). There are multiple solutions for the diagnostic problem in this case.



(a) Traditional approach with abonormality vertices



(b) Approach with state vertices

Fig. 2. Two methods of mapping of a simple system model with two components K and L to a Bayesian network

For example, $D = \{X1\}$, $D' = \{X2, R1\}$, and $D'' = \{X2, A2\}$ are (minimal) diagnoses.

3.2 Mapping to a Bayesian Network

To add a probabilistic aspect to consistency-based diagnosis, a logical diagnostic problem can be mapped to a Bayesian network. A Bayesian network is a directed acyclic graph that links random variables together as arcs between vertices [11]. Each arc between two connected vertices represents a (set of) conditional probability distributions.

There are different ways for translating a diagnostic problem into a **Bayesian diagnostic problem**. Two of these will be highlighted. One of these is the traditional method created by Srinivas [12] and the other is a more recent adaptation by ESI(TNO). The latter one is used for this research, but since it is based on the traditional method both will be expanded upon.

3.2.1 Traditional approach. The traditional method stems from the work of Srinivas in 1994 [12]. In Figure 2a, a simple system has been translated into a Bayesian network following this method.

Each input and output of a component are modelled as vertices. A component is modelled as a vertex node that is the child of all its input vertices. Note that one of the inputs of component L is the output of component K and thus the output vertex of K is directly linked to the output vertex of L. Next to these, per component,

an extra vertex Ab is added as the parent of the output vertex. This vertex corresponds with the abnormal predicate in MBD and similarly indicates whether the component behaves abnormal or not.

3.2.2 Approach based on state vertex. The method which is currently used by ESI(TNO) models the in- and outputs of the components similar to the traditional approach. The main difference is that the abnormality vertices are replaced by **state vertices**, as can be seen in Figure 2b. The state vertex is the child of all in- and output vertices of the corresponding component and defines a specific state for each combination of in- and outputs. The possible values of the state variable are the different behaviours that the pipe can exhibit, similar to fault modes as defined by Barbini et al. [1]. As an illustration, the abnormality vertices of the traditional approach could be mimicked by assigning the values Normal and Abnormal to the state variable. Where the state Normal indicates that all in- and outputs match the expected behaviour and Abnormal if this is not the case. The benefit of the state vertex lies in the fact that multiple fault modes can be included in a single variable, as will come apparent from the case in the methodology. In addition, the state vertices support enforcing extra dependencies between the connected variables.

3.2.3 Forming a diagnosis. With the logical diagnostic problem mapped to a Bayesian diagnostic problem, probabilistic inference methods can be used to derive the state (i.e., whether the component behaves correctly) of components to form a diagnosis [2].

Before such derivation can take place, first the evidence (i.e., known states of inputs and specific outputs) should be included, which is analogous to the observations in MBD. With probabilistic inference methods, the posterior probabilities of each of the state vertices can be calculated. Then for a set of state vertices S the diagnosis

$$D = \arg\max_{s} P(S = s \mid \text{Evidence}) \tag{2}$$

Note that the same method can be used for the traditional structure of a Bayesian diagnostic model. For that case, instead of the state vertices, the variables of the abnormality vertices are maximised.

4 METHODOLOGY

Within this paper, a simple theoretical system is used as context for the experiments. In the following sections, both this system as well as the evaluation methods are described. This system, including a Bayesian network representing the system, was provided by ESI(TNO). Next to a system and model, data is needed to perform the experiments. The decision was made to use artificially generated (synthetic) data, as it is quickly available and serves well as a first step to develop the appropriate methods before switching to real data.

The programming language Python was used to work with the Bayesian networks and to perform the experiments. For the Bayesian networks specifically, the pyAgrum package was used. This is a Python package that wraps the functionalities of the C++ aGrUM library, which implements many functionalities regarding working with graphical models [7].

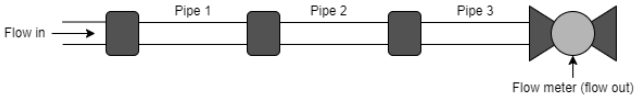


Fig. 3. System of connected water pipes

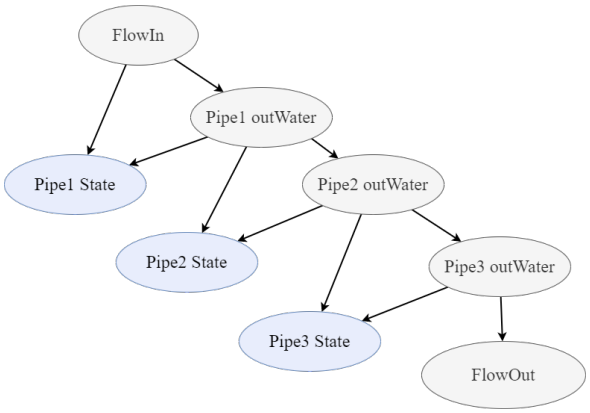


Fig. 4. Waterpipes system mapped to a Bayesian network

4.1 System

The system used consists of three water pipes connected in series. Water flows into the system and, after flowing through all pipes, is measured by a flow sensor at the end. A diagram of this system is provided in Figure 3. Within this system, the water flow is discredited in three states: NormalFlow, LowFlow and NoFlow. Each pipe has one flow coming into it ($inWater$) and one flow going out of it ($outWater$), if pipe x behaves normally $inWater_x = outWater_x$. Alternatively, if pipe x has a leak, the outgoing flow will be one level lower than the incoming flow:

$$inWater_x = \text{NormalFlow} \wedge \text{Leak}_x \implies outWater = \text{LowFlow}$$

and

$$inWater_x = \text{LowFlow} \wedge \text{Leak}_x \implies outWater = \text{NoFlow}$$

For a MassiveLeak, all water will escape the pipe:

$$\text{MassiveLeak}_x \implies outWater_x = \text{NoFlow}$$

Using the method to map a system to a Bayesian diagnostic problem as described in section 3.2.2, the system in Figure 3 can be translated to the Bayesian network as displayed in Figure 4. Within this diagnostic problem, the flow into the system and out of the system are known values and the states of each pipe are to be derived (i.e., the diagnosis).

4.2 Probability distributions

In a Bayesian network, each vertex has a conditional probability table (CPT) associated with it. This CPT defines conditional probabilities regarding its state given the states of its parent vertices. There are two types of vertices of which it is useful to analyse the CPTs: the pipe and the state vertex.

Table 1. CPT for the state vertex of a pipe

		Pipe State		
Pipe inWater	**Pipe outWater**	**Normal**	**Leak**	**MassiveLeak**
Normal Flow	**NormalFlow**	1	0	0
	LowFlow	0	1	0
	NoFlow	0	0	1
LowFlow	**NormalFlow**	1	0	0
	LowFlow	1	0	0
	NoFlow	0	1	0
NoFlow	**NormalFlow**	1	0	0
	LowFlow	1	0	0
	NoFlow	1	0	0

4.2.1 State CPT. As explained earlier in section 3.2.2, the state of a component c is modelled as a vertex with all in- and output vertices of c as parents. For the Waterpipes system specifically, the state of a pipe x is modelled as a vertex with parents $inWater_x$ and $outWater_x$.

The CPT of a pipe, as displayed in Table 1, defines which label should be associated with the behaviour of the pipe. The conditional probabilities map a specific fault mode to each combination of $inWater$ and $outWater$, corresponding to the description as in section 4.1. For example, $P(\text{State} = \text{Leak} \mid inWater = \text{NormalFlow}, outWater = \text{LowFlow}) = 1$. Since all components in the system are similarly behaving pipes, the CPT of all state vertices share the same conditional probabilities.

4.2.2 Pipe CPT. In Figure 4, it can be seen that each pipe is modelled as a flow out of the pipe $outWater$ with one flow coming into the pipe $inWater$. Since this structure is the same for every pipe, the CPT of each pipe is also similar such that it reflects $P(outWater \mid inWater)$. Different CPTs associated with this conditional probability are displayed in Table 2. As an illustration, from the CPT in Table 2a it follows that when the $inWater$ = NormalFlow, there is an equal likelihood that $outWater$ is either Normal-, Low- or NoFlow. Furthermore, if the $inWater$ = LowFlow, there is an equal likelihood that $outWater$ is either Low- or NoFlow while the probability of NormalFlow is 0. In other words, this models the behaviour that besides the $inWater$, there can be no extra water entering the pipe. This same modelling choice can be seen in the row that represents the case that $inWater$ =NoFlow.

In these CPTs, the probability that the pipe malfunctions is reflected. In Table 2b, the uniform distribution of Table 2a has been replaced, reflecting a distribution that assigns a certain, non-uniform, likelihood of the pipe failing. Within this adjustment, a probability of 0.9 is given to both cases where the pipe behaves normally (i.e., $inWater = outWater$). For the case where $inWater$ = NormalFlow, a probability of 0.07 is given to the state LowFlow and a probability of 0.03 to NoFlow, in other words, this reflects that a normal leak is more likely than a massive leak. For the case where $FlowIn$ = LowFlow the probability of NoFlow is 0.1, the accumulation of the two previous probabilities. In Tables 2c and 2d, other distributions that reflect a higher probability of failure are reflected.

Table 2. CPT distributions for different likelihoods of pipe failure

(a) Uniform distribution

	Pipe outWater		
Pipe inWater	**NormalFlow**	**LowFlow**	**NoFlow**
NormalFlow	1/3	1/3	1/3
LowFlow	0	0.5	0.5
NoFlow	0	0	1

(b) Low probability of failure

	Pipe outWater		
Pipe inWater	**NormalFlow**	**LowFlow**	**NoFlow**
NormalFlow	0.9	0.07	0.03
LowFlow	0	0.9	0.1
NoFlow	0	0	1

(c) Medium probability of failure

	Pipe outWater		
Pipe inWater	**NormalFlow**	**LowFlow**	**NoFlow**
NormalFlow	0.8	0.12	0.08
LowFlow	0	0.8	0.2
NoFlow	0	0	1

(d) High probability of failure

	Pipe outWater		
Pipe inWater	**NormalFlow**	**LowFlow**	**NoFlow**
NormalFlow	0.7	0.2	0.1
LowFlow	0	0.7	0.3
NoFlow	0	0	1

4.3 Generation of synthetic data

To generate synthetic data, the choice was made to use the Bayesian network of the waterpipes system. This can be done by sampling the network, which yields a dataset where the individual samples each contain an instantiation of all network variables. The samples are generated reflecting the probabilities in the network, this allows for the generation of data that reflects a specific configuration of pipes with different likelihoods of failure. For this, the *generateSample* function from the pyAgrum library is used.

Along with a specific configuration of pipe failure probabilities, the prior probability for the FlowIn variable is also altered. Within a uniform distribution, $P(\text{FlowIn} = v) = 1/3$ for every $v \in \{\text{NormalFlow, LowFlow, NoFlow}\}$. Because the value NoFlow yields no useful samples (since FlowIn = NoFlow $\implies$ FlowOut = NoFlow with no diagnosis to be made), the prior probability of this value is set to 0. Next to this, the prior probabilities $P(\text{FlowIn} = \text{NormalFlow}) = 2/3$ and $P(\text{FlowIn} = \text{LowFlow}) = 1/3$ were set.

4.4 Forming of a diagnosis

As explained in section 3.2, for a diagnosis to be formed, the known states of the system should be included as evidence after which, through inference, the posterior probabilities of the state vertices are retrieved. In Figure 5, the pyAgrum library is used to visualise the result of the inference for the case where FlowOut < FlowIn. A diagnosis is formed by iterating over the state vertices and selecting

the value with the highest posterior probability, conform equation 2.

4.5 Evaluation methods

To evaluate a specific Bayesian diagnostic model, three evaluation methods are used. In the following section, each method will be discussed; both how the metric is computed as well as the significance/use of the method. Following these, section 4.6 will highlight how each method will be used to work towards answering the research question.

4.5.1 Cartesian product. For every equal instance of evidence, the Bayesian diagnostic model will yield the same diagnosis. So to get an insight into all possible diagnoses that can be given by a specific model, every unique combination of evidence can be used to form a diagnosis. To get every possible combination of evidence, the Cartesian product can be used.

Definition 4.1. (**Cartesian product of evidence**) For a Bayesian diagnostic model with n evidence vertices, let E_n be every vertex that is part of the evidence. Let then V_n be the set of possible values of vertex E_n. Then the Cartesian product of the evidence

$$E_1 \times \cdots \times E_n = \{(v_1, \ldots, v_n) \mid v_i \in V_i \text{ for every } i \in \{1, \ldots, n\}\} \quad (3)$$

Note that the size of the Cartesian product can expand exponentially when the number of evidence vertices or values increases. In the case of the waterpipes system with two evidence vertices, of which each variable has three possible values, the number of tuples in the Cartesian product is $3^2 = 9$. Note that these also include scenarios that are less interesting from the perspective of making diagnoses: where FlowIn = FlowOut so that there is no malfunction in the system and scenarios where the FlowOut > FlowIn. Although the latter is a correct result from the Cartesian product, as described in 4.2.2 the Bayesian network does not allow the flow to be increased. Therefore, these scenarios result in an inconsistency in the network and therefore do not allow for a diagnosis to be formed.

4.5.2 Accuracy. Using a Bayesian diagnostic model, a diagnosis can be made on samples from the synthetic data. The samples can be split into the evidence and diagnosis values (as described in section 4.4). Then the evidence values can be fed to the Bayesian diagnostic model to form a diagnosis, which can then be compared with the diagnosis values from the sample to see whether the formed diagnosis was correct. The accuracy can then be calculated as

$$\text{Accuracy} = \frac{\#\ correct\ diagnoses}{\#\ total\ diagnoses} \quad (4)$$

4.5.3 Log-likelihood. The log-likelihood is a quality measure that describes how likely a dataset is given a particular Bayesian network [3]. With the presence of a dataset, multiple models can be compared to see which one best fits the data.

Definition 4.2. (**Log-likelihood**) Let B be a Bayesian network with variables $X_1, \ldots, X_n$. Furthermore, let D be a dataset of size m with individual tuples $d \in D$ with d being an instance of the variables $(X_1 = x_1, \ldots, X_n = x_n)$.

Table 3. Possible pipe failure configurations

Pipe 1	Pipe 2	Pipe 3
High	Medium	Low
High	Low	Medium
Medium	High	Low
Medium	Low	High
Low	High	Medium
Low	Medium	High

The likelihood of the dataset D can be calculated as

$$P(D \mid B) = \prod_{d \in D} P_B(d)$$

Where $P_B(d)$ is the joint probability of instance d using the probability distribution of the Bayesian network B:

$$P_B(d) = \prod_{i=1}^{n} P(x_i \mid parents(X_i))$$

Because computers can more efficiently calculate the sum of large series, alternatively, the log-likelihood can be calculated.

$$\log P(D \mid B) = \sum_{d \in D} \log P_B(d)$$

The log-likelihood calculation yields a number in the range $(-\infty, 0]$ where a value of zero indicates a perfect fit [3]. The log-likelihood is not an absolute measure; It is only suitable to compare several models within the context of a single dataset.

4.6 Experiments

To answer the research question, two types of analyses will be made. One to evaluate the performance of a model on test data, and secondly, to evaluate the diagnostic behaviour. The analysis will be made for different Bayesian diagnostic models with each a different configuration of pipe failure probabilities. The different pipe failure configurations are a permutation of the set {High, Normal, Low} which are also displayed in Table 3. The values High, Medium and Low refer to the CPTs as listed in Table 2.

4.6.1 Performance on test data. For each possible pipe configuration as listed in Table 3, a dataset of 10,000 samples will be generated (as described in section 4.3). To evaluate the advantage of a model that reflects the real-world situation, two models will be compared on their fit to the sample data. One model that matches the pipe configuration with which the data was generated and a model with uniform distributions (the base case). Both these models will be used to form diagnoses (as described in section 4.4) on the test data, after which the accuracy for both can be calculated. Next to this, the log-likelihoods of the models given the data are evaluated.

The generated data also contains samples where there is no malfunctioning, i.e., where FlowIn = FlowOut. These are relevant to evaluate since diagnostic models should also be able to form a correct diagnosis when nothing is wrong, i.e. the diagnosis that all pipes behave normally. However, these cases are also way easier to predict. Therefore, the accuracy will also be calculated for the cases where FlowIn != FlowOut. This helps in evaluating the diagnostic

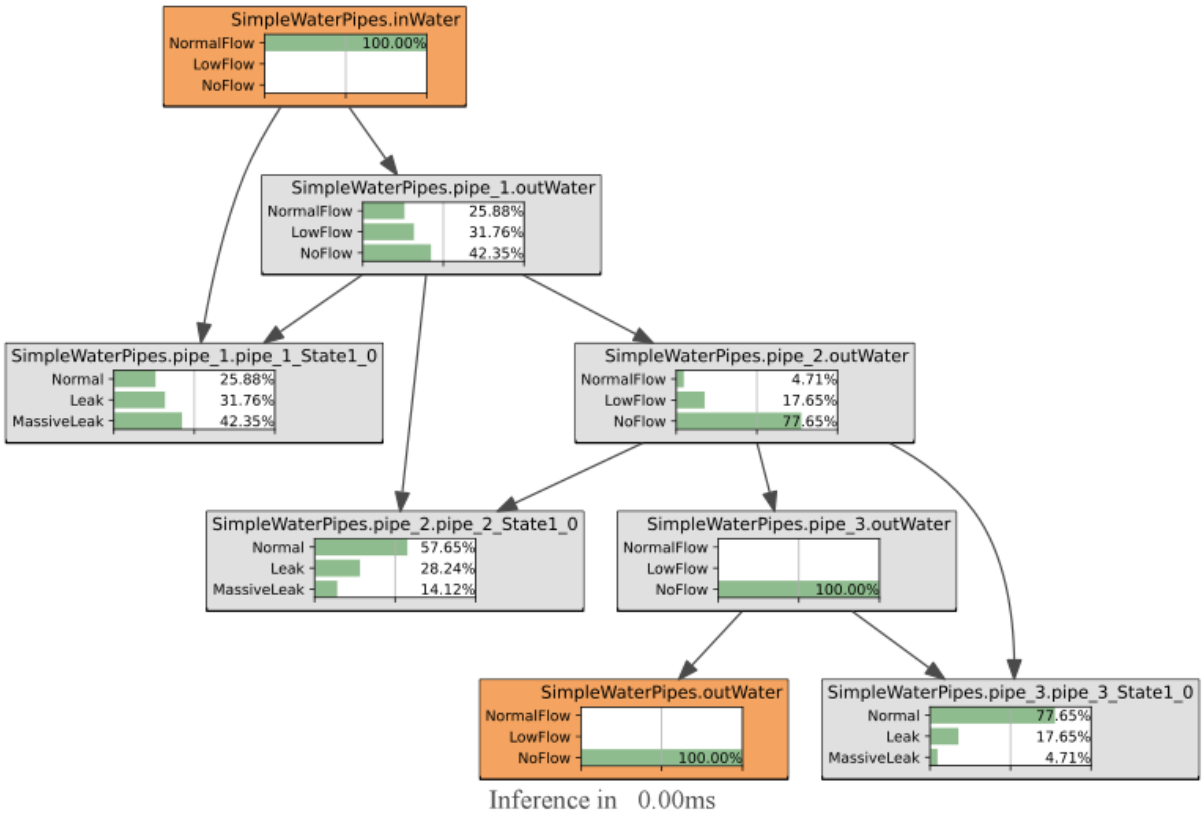


Fig. 5. Visualisation using the pyAgrum library for the inference of the waterpipes system with a uniform distribution, where FlowOut < FlowIn.

capabilities of a system in scenarios where there is a malfunctioning, which is often the most interesting scenario.

4.6.2 Diagnostic behaviour. As explained in section 4.5.1, all possible diagnoses by a specific model can be derived by diagnosing all elements in the Cartesian product of the evidence. For each pipe configuration as in Table 3, the Cartesian product will be generated. Then, for each tuple of evidence, a diagnosis will be made.

For the purpose of evaluating the diagnostic capabilities, only the elements from the Cartesian product where FlowOut < FlowIn will be considered. As the other elements are either scenarios where there is no malfunction or scenarios that are not inferable (as explained in section 4.5.1).

5 RESULTS

In this section, the results of the experiments are listed. This is done per experiment as described in section 4.6.

5.1 Performance on test data

Table 4 lists all the calculated metrics as described in the experiment in section 4.6.1. The log-likelihood shows a better fit for the adjusted model in all cases, with an average advantage of 32%.

The overall accuracy also increased for the adjusted model. There is however an exception in the two cases where pipe 1 has a medium probability of failure. For these, the uniform model performs slightly better.

Worth noting is that for the samples in the data where there is no malfunction (i.e., FlowIn = FlowOut) both models make a correct diagnosis at all times. This results in the fact that the difference between the two models for the general accuracy is solely explained by the difference in the accuracy of the malfunctioning sample cases. From these results it follows that, on average, the adjusted model outperforms the uniform model by 10.4 percentage points.

5.2 Diagnostic behaviour

In Table 5, the possible diagnoses are listed for each permutation of pipe configuration. The state variable values Leak and MassiveLeak are highlighted to give a clearer view of the made diagnoses. It can be seen that the diagnosis always marks the pipe with the highest probability of failure, which is not surprising. What is surprising however, for quite some cases, a **non-diagnosis** is made. Within these cases, the diagnosis states that every component behaves normally when there is a discrepancy between FlowIn and FlowOut, which results in a contradiction.

In Figure 5, it can be seen that the posterior probabilities of a component failure are higher for pipes up in the system (i.e., pipe 1) than for components closer to the output (i.e., pipe 3). This tendency rivals the probability of component failure; As the pipe with the

Table 4. Accuracy and log-likelihood metrics for a uniform model and a model that reflects the probabilities with which the data was generated, for different configurations of pipe failures

Pipe Failure Configuration			Log-likelihood		General Accuracy		Accuracy Malfunctioning cases		**Ratio malfunctioning cases**
Pipe 1	**Pipe 2**	**Pipe 3**	**Uniform**	**Matching**	**Uniform**	**Matching**	**Uniform**	**Matching**	
High	Medium	Low	-30203.834	-21156.643	0.666	0.765	0.331	0.530	0.499
High	Low	Medium	-30729.943	-21049.509	0.682	0.777	0.351	0.545	0.491
Medium	High	Low	-30859.094	-21070.126	0.633	0.611	0.247	0.202	0.487
Medium	Low	High	-31927.939	-21467.365	0.623	0.598	0.244	0.192	0.498
Low	High	Medium	-32066.654	-21417.269	0.561	0.687	0.115	0.370	0.496
Low	Medium	High	-32546.850	-21706.094	0.553	0.588	0.111	0.182	0.503
		Average	**-31389.052**	**-21311.168**	**0.620**	**0.671**	**0.233**	**0.337**	**0.496**

highest probability of failure moves towards FlowOut, more non-diagnoses are made.

6 DISCUSSION AND CONCLUSIONS

On average, the adjusted models perform better on the test data than the uniform model. Especially the log-likelihood shows a consistently better fit for the adjusted model. However, for two configurations of the system, the uniform model outperforms the adjusted one as far as accuracy. This indicates that a Bayesian network containing probabilities from a real-world scenario might not always perform better, at least as used in the context of this research.

Following the analysis of the diagnostic behaviour, the BN structure with state vertices as used in this research shows some undesired behaviour. In several cases, a non-diagnosis is made. This phenomenon does not occur in Bayesian diagnostic models that make use of the more traditional Ab vertex [2]. Furthermore, this structure might also give some challenges when it comes to learning probabilities of failure from data. It would be easiest to learn a value P(fault) from the data. However, this value can not be directly included in the structure of the BN as used in this research, which instead asks for a value of P(FlowOut | FlowIn). As another illustration of this problem, if for a certain pipe FlowIn = LowFlow and FlowOut = NoFlow, the state of this component will be Leak. However, in a real-life scenario, a MassiveLeak may result in the same flow levels.

A possible way of mitigating these issues could be by having the state vertex as the root vertex for the output vertex of a component, similarly to the Ab vertex. As opposed to the Ab vertex, the state would still contain different fault modes (where the Ab is limited to Normal or Abnormal). The component CPT could then include the behaviour for different fault modes. Alternatively, a different method of forming diagnoses could be used that (partly) bypasses the posterior probabilities of the state vertices.

It should be noted that these limitations of the BN structures only apply to applications where a diagnosis is made for a similar purpose as this research. It does not impede other analytical procedures, such as assessing the diagnosability of a system as illustrated in the works of Barbini et al. [1].

All in all, this research contributes to ideas on how MBD can be applied to cyber-physical systems. Especially for MBD in later stages of products where performance data is available. While in the first stages design knowledge is used, the performance data can improve the diagnostical process at later stages. The best methods for achieving this are up for further research, in which the findings from this research can play a guiding role.

ACKNOWLEDGMENTS

I want to thank Peter Lucas for the supportive manner in which he supervised me. Next to this, I would like to thank Thomas Nägele from ESI(TNO) for the provision of, and assistance with the system model that is used in this research.

REFERENCES

[1] Leonardo Barbini, Carmen Bratosin, and Thomas Nägele. 2021. Embedding Diagnosability of Complex Industrial Systems Into the Design Process Using a Model-Based Methodology. In *PHM Society European Conference*, Vol. 6. 9–9.
[2] Leonardo Barbini, Carmen Bratosin, and Emile van Gerwen. 2020. Model based diagnosis in complex industrial systems: a methodology. In *PHM Society European Conference*, Vol. 5. 8–8.
[3] Remco R. Bouckaert. 1995. *Bayesian belief networks: from construction to inference.* Ph. D. Dissertation.
[4] Andreas Bunte, Benno Stein, and Oliver Niggemann. 2019. Model-Based Diagnosis for Cyber-Physical Production Systems Based on Machine Learning and Residual-Based Diagnosis Models. Association for the Advancement of Artificial Intelligence (AAAI), 2727–2735. https://doi.org/10.1609/aaai.v33i01.33012727
[5] Johan de Kleer, Alan K. Mackworth, and Raymond Reiter. 1992. Characterizing diagnoses and systems. *Artificial Intelligence* 56, 2-3 (1992), 197–222. https://doi.org/10.1016/0004-3702(92)90027-u
[6] Barry Dowdeswell, Roopak Sinha, and Stephen G. MacDonell. 2020. Finding faults: A scoping study of fault diagnostics for Industrial Cyber–Physical Systems. *Journal of Systems and Software* 168 (2020), 110638. https://doi.org/10.1016/j.jss.2020.110638
[7] Gaspard Ducamp, Christophe Gonzales, and Pierre-Henri Wuillemin. 2020. aGrUM/pyAgrum : a Toolbox to Build Models and Algorithms for Probabilistic Graphical Models in Python. In *10th International Conference on Probabilistic Graphical Models (Proceedings of Machine Learning Research, Vol. 138)*. Skørping, Denmark, 609–612. https://hal.archives-ouvertes.fr/hal-03135721
[8] Igor Mozetič. 1992. Model-based diagnosis: an overview. *Advanced Topics in Artificial Intelligence* (1992), 419–430.
[9] Judea Pearl. 1988. *Probabilistic reasoning in intelligent systems: networks of plausible inference.* Morgan kaufmann.
[10] Raymond Reiter. 1987. A theory of diagnosis from first principles. *Artificial intelligence* 32, 1 (1987), 57–95. https://doi.org/10.1016/0004-3702(87)90062-2
[11] Stuart Russell and Peter Norvig. 2016. Probabilistic Reasoning. In *Artificial Intelligence: A Modern Approach* (3 ed.). Pearson, 510–565.
[12] Sampath Srinivas. 1994. A probabilistic approach to hierarchical model-based diagnosis. In *Uncertainty in Artifical Intelligence*. Elsevier, 538–545.

Table 5. Possible diagnoses per pipe configuration

(a) "High-medium-low" failure configuration

		Pipe1	**Pipe2**	**Pipe3**
Likelihood of failure		*High*	*Medium*	*Low*
Flow In	**Flow Out**			
NormalFlow	LowFlow	Leak	Normal	Normal
NormalFlow	NoFlow	MassiveLeak	Normal	Normal
LowFlow	NoFlow	Leak	Normal	Normal

(b) "High-low-medium" failure configuration

		Pipe1	**Pipe2**	**Pipe3**
Likelihood of failure		*High*	*Low*	*Medium*
Flow In	**Flow Out**			
NormalFlow	LowFlow	Leak	Normal	Normal
NormalFlow	NoFlow	MassiveLeak	Normal	Normal
LowFlow	NoFlow	Leak	Normal	Normal

(c) "Medium-high-low" failure configuration

		Pipe1	**Pipe2**	**Pipe3**
Likelihood of failure		*Medium*	*High*	*Low*
Flow In	**Flow Out**			
NormalFlow	LowFlow	Normal	Leak	Normal
NormalFlow	NoFlow	Normal	Normal	Normal
LowFlow	NoFlow	Normal	Normal	Normal

(d) "Medium-low-high" failure configuration

		Pipe1	**Pipe2**	**Pipe3**
Likelihood of failure		*Medium*	*Low*	*High*
Flow In	**Flow Out**			
NormalFlow	LowFlow	Normal	Normal	Leak
NormalFlow	NoFlow	Normal	Normal	Normal
LowFlow	NoFlow	Normal	Normal	Normal

(e) "Low-high-medium" failure configuration

		Pipe1	**Pipe2**	**Pipe3**
Likelihood of failure		*Low*	*High*	*Medium*
Flow In	**Flow Out**			
NormalFlow	LowFlow	Normal	Leak	Normal
NormalFlow	NoFlow	Normal	Normal	Normal
LowFlow	NoFlow	Normal	Leak	Normal

(f) "Low-medium-high" failure configuration

		Pipe1	**Pipe2**	**Pipe3**
Likelihood of failure		*Low*	*Medium*	*High*
Flow In	**Flow Out**			
NormalFlow	LowFlow	Normal	Normal	Leak
NormalFlow	NoFlow	Normal	Normal	Normal
LowFlow	NoFlow	Normal	Normal	Normal