Factoring Dynamic Bayesian Networks using Possible Conflicts

Carlos J. Alonso-Gonzalez [1], **Noemi Moya** [1], **and Gautam Biswas** [2]

[1] *Department of Computer Science University of Valladolid, Valladolid, 47011, Spain*
calonso@infor.uva.es
noemi@infor.uva.es
[2] *Institute for Software Integrated Systems , Nashville, TN 37235, USA*
gautam.biswas@vanderbilt.edu

ABSTRACT

Dynamic Bayesian Networks (DBNs) are temporal probabilistic graphical models that represent in a very compact way dynamic systems. They have been used for model based diagnosis of complex systems because they naturally cope with uncertainties in the diagnosis process, particularly sensor uncertainty in noisy environments. A caveat of DBN is the complexity of the inference procedure which is usually performed with Particle Filtering algorithms. Recently, factoring has been proposed to decompose a DBN into subsystems, distributing the diagnosis process and reducing the computational burden.

This paper proposes decomposing a system with Possible Conflicts (PCs) and, afterwards, building a DBN factor from each resultant PC. The method can be systematically applied to a state space representation of a dynamic system to obtain minimal observable subsystems with analytical redundancy. Assuming single fault hypothesis and known fault modes, the method allows performing consistency based fault detection, isolation and identification with the unifying formalism of DBN. The three tank system benchmark has been used to illustrate the approach. Two fault scenarios are discussed and a comparison of the behaviors of a DBN of the complete system with the DBN factors is also included.

1 INTRODUCTION

The increasing complexity of current engineering systems, together with the increasing demand on their safe and reliable operation even in the presence of system faults, makes fault diagnosis an essential tool. Faults must be detected, and if possible isolated and identified, close to their onset (Narasimhan, 2007) so that quick action can be taken to minimize the effects of the fault and thus prevent damage. Due to the complexity of these systems, formal methods are required for systematic design, analysis, and implementation of system diagnosers. Model-based diagnosis provides a formal framework to achieve these objectives.

Main approaches to model based diagnosis of continuous systems are consistency based, control theory based, and stochastic based (Narasimhan, 2007). Stochastic approaches have promoted the use of probabilistic methods for fault diagnosis. This is motivated by the uncertainty in the diagnosis process. The main sources of uncertainty are the models and the sensors, particularly in noisy environments. Among stochastic approaches, Dynamic Bayesian Networks (DBNs) (Murphy, 2002) play an important role.

DBNs have been applied (Dearden and Clancy, 2001; Koller and Lerner, 2001) to fault diagnosis because they allow estimating state variables of a dynamic system without the usual Gaussian assumption for noise and modeling errors, which no longer apply when faults occur (Arulampalam *et al.*, 2002). Its major drawbacks are computational complexity of learning and inference procedures. In model based diagnosis, network structure and coefficients may be obtained from models, particularly from Temporal Causal Graphs (TCGs) (Lerner *et al.*, 2000; Roychoudhury *et al.*, 2008). Real time inference has been tackled with Particle Filtering (Arulampalam *et al.*, 2002).

A problem with Particle Filtering is 'sample impoverishment': less weighted samples tend to disappear. Importance sampling may reduce this effect that is especially harmful for diagnosis: faulty states have small probabilities.(Roychoudhury *et al.*, 2008) proposes solving this problem using multiple DBNs: a nominal DBN to track the system in normal operation and, under single fault hypothesis, a DBN to model each fault. When a fault is detected, the TRANSCEND method (Mosterman and Biswas, 1999) is used to generate fault hypotheses. The fault hypotheses are tracked in parallel by their associated faulty DBN. Eventually, the DBN which best fits observations provides fault isolation and fault identification. The major drawback of this proposal is the computational complexity of hypotheses tracking, because each DBN models the whole systems plus the hypothesized fault.

To reduce computational complexity of inference, factoring DBNs has been recently proposed. In

(Roychoudhury *et al.*, 2009; Roychoudhury, 2009) a method is proposed to obtain Factored DBNs, based on conditional independence. After initial factors have been obtained, they are augmented to achieve a structural observable system. Structural observability is needed to assure that the state variables of the DBN factor can be estimated from observations and known inputs.

In this paper a different approach is proposed, based on Possible Conflicts (PCs) (Pulido and Alonso-Gonzalez, 2004). For a given a system, PCs induces a decomposition in minimal structurally observable subsystems (Moya *et al.*, 2010). Hence our proposal consists of first finding the set of PCs of a system and afterwards obtaining the DBN factors from the PCs.

An additional advantage of factoring with PCs is that fault isolation may be performed on the standard framework of Consistency Based Diagnosis, because the support of a DBN factor is the same that the support of the PC that generates it. Fault identification requires considering fault modes. Assuming single fault hypothesis, DBN factors can be modify to model the fault and tracking the faulty system.

Along the paper, the three tank system benchmark is used to illustrate several concepts. Section 2 provides a basic background about DBNs. Section 3 discusses a method to derive DBNs from PCs. Section 4 introduces the diagnosis architecture, explaining how to perform fault detection, isolation and identification with PC factored DBNs. In Section 5 two fault scenarios are examined. Section 6 compares the performance of the DBN of the complete system with the DBNs factors. After Discussion, some Conclusions are stated.

2 DYNAMIC BAYESIAN NETWORKS BACKGROUND

Dynamic Bayesian Networks are a probabilistic temporal model representation of a dynamic system. Basically, a DBN is a two slices Bayes Network (BN). Assuming that the system is time invariant and a First Order Markov process, two static and identical BN connected by inter slice arcs are enough to model the system (Murphy, 2002). Inter slices arcs model system dynamics. Intra slice arcs model instantaneous (algebraic) relations.

The system variables (X, Z, U, Y) represented in a DBN are the inputs (U), the state variables (X), the observed or measured variables (Y) and, in some cases, other hidden variables (Z) . Once we have the nodes, we need to define the parameters of the model, which are the state transition model (graphically represented by the inter slice arcs) and the sensor model (represented by intra slice arcs).

Exact inference in DBNs is not computationally tractable. Hence, Monte Carlo simulation methods are use for approximate inference, particularly Particle Filter algorithm (Koller and Lerner, 2001). The unknown continuous stochastic distribution of the state is approximated by a discrete distribution obtained by weighted samples. After propagation of the state, the weights are updated with current observations. In this work, we assume a Gaussian distribution.

Figure 1 shows the three tank system and Figure 2 shows its DBN model. There are three available measurements: (1) the flow out of tank 1 (F_1), (2) the flow between tank 1 and tank 2 (F_{12}) and (3) the flow out of tank 3 (F_3). They are represented in the network by f_4, f_6 and f_{16} respectively. F_{in} is a constant input, represented by node f_1. Nodes e_2, e_8 and e_{14} are the state variables, the pressures at the bottom of each tank. Hence, $X = \{e_2, e_8, e_{14}\}$, $U = \{f_1\}$, $Y = \{f_4, f_6, f_{16}\}$ and $Z = \{\phi\}$. The DBN can be manually constructed from conditional independence considerations. We have derived it from its TCG, which is automatically generated from a Bond Graph model of the system (Mosterman and Biswas, 1999).

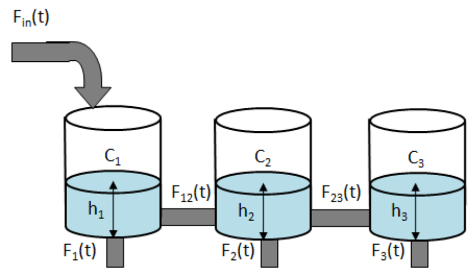

Figure 1: Three tanks system.

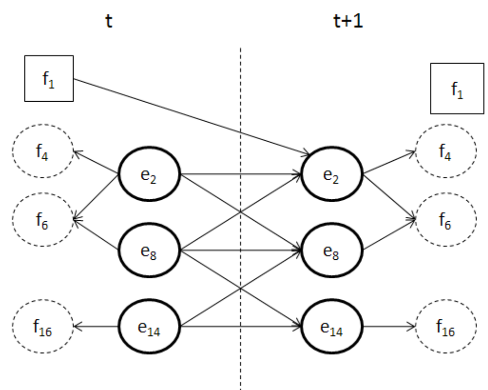

Figure 2: DBN of the the three tank system.

3 OBTAINING DBN FACTORS FROM PCS

3.1 Possible Conflicts background

Possible Conflicts (PCs) is a compilation technique for consistency based diagnosis of dynamic systems (Pulido and Alonso-Gonzalez, 2004). Essentially, PCs are minimal over determined subsystems with analytical redundancy.

Possible conflicts can be automatically derived from a hypergraph model of a system. This hypergraph is just an abstract representation of the system equations in state space form. Hyperarcs of the hypergraph represent an equation (more generally, a constraint) and the nodes included in the hyperarc are the variables of the equation (i.e. constraint).

Each PC has associated a directed hypergraph call *Minimal Evaluable Model (MEM)*. Nodes of the directed hypergraph represent variables of the system and directed hyperarcs represent a constraint with a causal assignment. From a given *MEM*, a computational model of a PC can be directly obtained replacing each hyperarc by its corresponding equation. A distinguished node in a *MEM* is the discrepancy node. The discrepancy node is always a estimated variable that can be estimated by another path or directly observed. It is the node where redundancy manifests.

The three tank system of Figure 1 has three PCs. Figure 3 shows the *MEM* of PC1 for this system. $ec1\_1$ models the mass balance at tank 1, $ec4\_1$ models the output and $ec12\_1$, $ec13\_1$ and $ec14\_1$ model the sensors. The dash arc is a differential constraint in integral causality. The discrepancy node in PC1 is f_4.

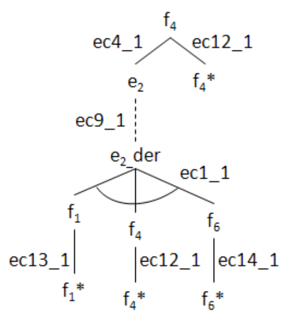

Figure 3: Possible conflict 1 of the system three tanks system.

An important aspect of PCs is that they induce a decomposition of a system into a set of subsystems (Alonso-Gonzlez *et al.*, 2008). A first feature of this decomposition is that it is systematic and unique. A second important feature is that the subsystems are independent, in the sense that they can be analyzed isolated, because they are structurally observable (Moya *et al.*, 2010). Moreover, they are minimally redundant. These properties make them an interesting tool to decompose DBNs.

3.2 Generating a DBN from a PC

PCs can also be obtained using the Bond Graph formalism (Bregon *et al.*, 2009). Hence it is to be expected that factored DBNs may be obtained from a PC. In this work we propose a method to obtain a DBN directly from the associated *MEM* of a PC.

Proposition 1. *Those Possible Conflicts with a* MEM *containing:*

- ***Condition 1****, a state variable and a differential arc,*
- ***Condition 2****, a path made only of instantaneous arcs from an estimated state variable to a discrepancy node that is observed,*

provide the minimal structural description of a DBN for the subsystem defined by the possible conflict.

Condition 1 is required to have a dynamic system. Condition 2 is necessary to avoid an empty sensor model in the DBN.

The demonstration of **Proposition 1** is constructive and generally requires two steps:

- **Structure :** Generate DBN structure from nodes and hyper arcs of the related *MEM* according to the criteria of Tables 1 and 2.
- **Simplification :** For any state variable which is conditionally dependent only on input nodes, replace that state variable and inputs by a new input node, according to algebraic *MEM* computation.

The criterion 1 of the *Structure* step defines the initial set of nodes of the DBN. Arcs in the network are added according to criterion 2. Second step of the construction process just simplifies the network, eliminating state variables that are algebraically estimated from known inputs and observed variables in the original *MEM*.

Table 1: Equivalence between nodes in the hypergraph of a MEM and nodes in the DBN.

PCs	**DBNs**
Inputs (U)	Inputs
Observation of the discrepancy node	Observation (sensor model)
Any other observation	Input
States	States

Table 2: Equivalence between relations in the hypergraph of a MEM and arcs in the DBN.

PCs	**DBNs**
Differential constraint	Inter slice arc for related state variable
Path from a state variable to a state variable, including only one differential constraint	Inter slice arcs from state variable to state variable
Path from an observation or input to a state variable, including only one differential constraint and no additional state variables	Inter slice arcs from nodes to state variable
Paths without differential constraints, starting or ending at a state variable	Intra slice arcs

Figure 4 shows the DBN obtained from the PC1 of the three tank systems, applying just the *Structure* step. None of the PCs of this system needs to perform the *Simplification* step to generate the DBN. Figure 5 shows a PC whose transformation on a DBN requires both steps. Figure 6 shows the intermediate network created applying step 1 and Figure 7 the final DBN after step 2.

4 DIAGNOSIS ARCHITECTURE WITH DBNS AND PCS

Factored DBNs from Possible Conflicts allows tackling all the stages of model based diagnosis, that is, fault detection, fault isolation and fault identification,

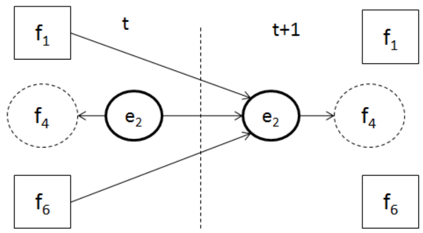

Figure 4: DBN of the possible conflict 1 of the three tank system.

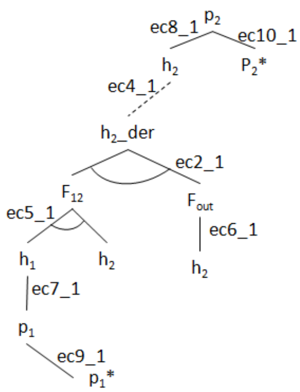

Figure 5: Possible conflict of a two tank system with the pressure of each tank measured.

in the Consistency Based Diagnosis framework with fault models in a predictive approach. Figure 8 shows the architecture of the system.

4.1 Fault Detection

Nominal DBNs factors are obtained off line from the system model through PCs decomposition. The three resultant DBNs factors for the three tank system are shown in Figure 4, Figure 9 and Figure 10. These DBN factors are run in parallel to perform fault detection. A ztest (Biswas *et al.*, 2003; Gelso *et al.*, 2008) on the residual of tracked variables is used to decide on detection of each DBN factor.

4.2 Fault isolation

In a predictive approach, fault isolation requires introducing fault modes. We have opted for a simple abrupt fault model (Roychoudhury *et al.*, 2008).

Abrupt fault An abrupt fault is characterized by a fast change in a parameter value. The temporal

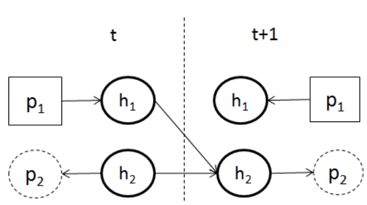

Figure 6: Intermediate DBN of the possible conflict in Figure 5 after the *Structure* step.

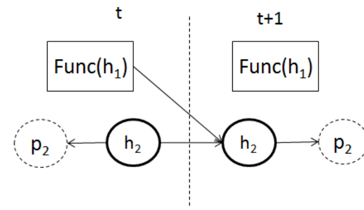

Figure 7: DBN of the possible conflict in Figure 5 after the *Simplification* step.

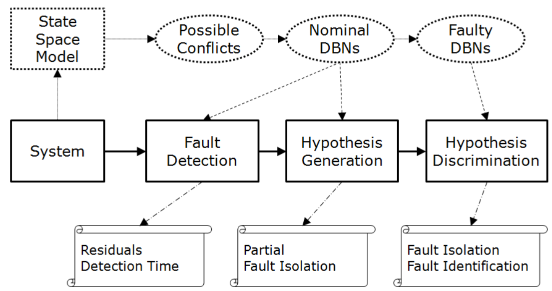

Figure 8: The diagnosis architecture integrating DBNs and PCs

profile of a parameter with an abrupt fault, $p^a(t)$ is given by:

$$p^a(t) = \begin{cases} p(t) & t < t_f \\ p(t) + b(t) = p(t) + \sigma_p^a & t \geq t_f \end{cases}$$

where σ_p^a models the absolute change of the parameter value. For all the considered faults, σ_p^a is set to 10% of the nominal value of the parameter.

We have considered 8 abrupt faults: in the capacitances of each tank (C1, C2 and C3), in the resistance of the output of each tank (Rv1, Rv2 and Rv3) and in the resistance of the flow between tanks (Rv12 and Rv23). The fault signature matrix of the DBNs factors is the same as the fault signature matrix of the PCs of the system, shown in Table 3. We use this fault signature matrix to generate Reiter candidates, which are updated if new observations generate new detections,

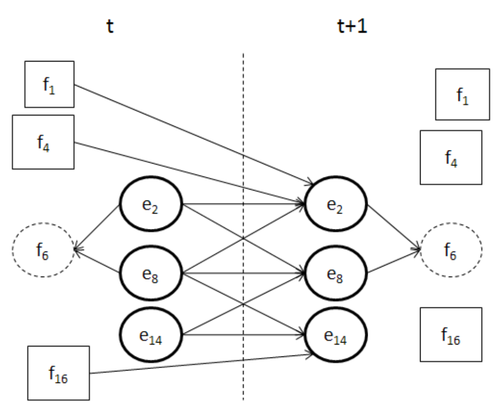

Figure 9: DBN of the PC2 of the three tank system.

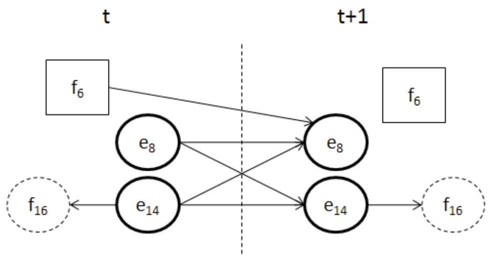

Figure 10: DBN of the PC3 of the three tank system

assuming non intermittent faults.

Table 3: Signature matrix of the system in Figure 1

	PC1	PC2	PC3
C1	1	1	
C2		1	1
C3		1	1
Rv1	1		
Rv12		1	
Rv2		1	1
Rv23		1	1
Rv3			1

For complexity reasons, we limit fault identification to single faults. DNB factors for each fault mode are obtained from DBN factors of the nominal system according to (Roychoudhury *et al.*, 2008; Roychoudhury, 2009) proposal. Nominal DBN factors are extended with an additional node for the faulty parameter. If some network node is conditionally dependent on the new node, and edge is added from the new node to the *'not conditionally independent'* node. Figure 11 shows the faulty network factor obtained from PC1 for an abrupt fault in the capacitance of tank 1. For each DBN factor it is necessary to build as many faulty DBNs as indicated in the fault signature matrix.

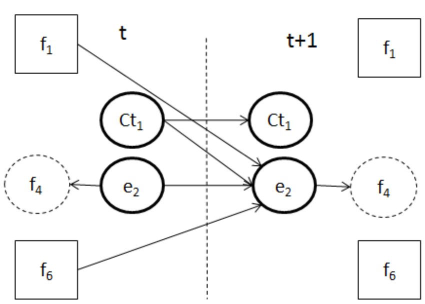

Figure 11: DBN of possible conflict 1 of the three tanks system with a fault in the capacitance of tank 1.

Fault isolation requires tracking the system with faulty DBNs. For each single fault candidate a faulty DBN factor has to track the system. If a new detection allows reducing the number of single faults candidates, the corresponding fault hypotheses are rejected and the associated DBN factors no longer track the system. Eventually, one of the faulty DBNs will converge identifying the new value of the parameter.

5 FAULT SCENARIOS

We have developed two fault scenarios for the three tank system: (1) an abrupt fault in the capacitance of tank 1 and (2) an abrupt fault in the resistance out of tank 3. In both cases the fault magnitude is 10% of the nominal value of the parameter. We have simulated for 10.000 time steps, starting with the three tanks empty and injecting the faults at time stamp 2000. Simulink has been used to generate data of the faulty system. A 5% and a 0.5% Gaussian noise has been added to sensors and input, respectively. The number of particles used in the Particle Filter algorithm has been 500. Ztest has been applied to decide on network detection and also on network convergence for fault identification.

5.1 Abrupt fault in C1

Fault detection is performed with the nominal DBN factors (see Figure 12). The DBN from PC2 detect the fault at time 2001 (Figure 12 b)) while the factor from PC1 detect the fault at time 2002 (Figure 12 a)). According to the fault signature matrix of the system, Table 3, the factor from PC3 does not detect the fault (Figure 12 c)).

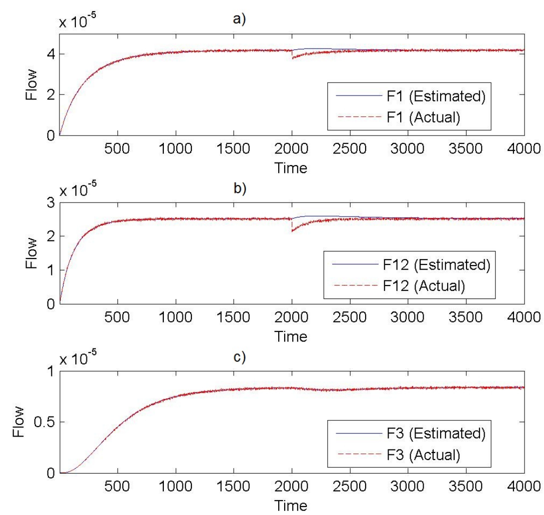

Figure 12: Behaviour (observed variable) of the nominal DBN of a) PC1, b) PC2 and c) PC3 tracking an abrupt fault (10%) in the capacitance of tank 1.

Hence, from time 2002 there is only one single fault candidate: C1. We have run the faulty DBN of PC1 for a fault in C1 (Figure 11) starting 50 time steps before the fault is injected, to launch simulation from a known system state with nominal behavior. The behavior of the network is shown in Figure 13. Convergence time is 389 time steps, (339 after fault injection). Figure 13 shows that the faulty parameter converges to an acceptable value many instants before, but ztest is performed on the residual of the observed flow (not shown in the figure) and introduces a 170 time stamps delay.

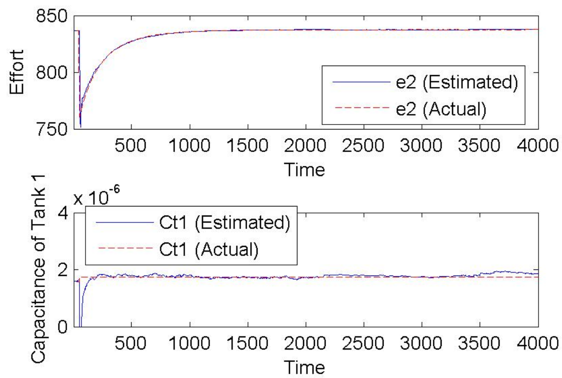

Figure 13: State variables tracked with the DBN of PC1 for a fault in capacitance of tank 1. The last chart at the bottom is the estimation of the parameter C1.

5.2 Abrupt Fault in Rv3

Like in the previous scenario, fault detection is performed with the three nominal DBN factors. Now, only the factor form PC3 detects the fault, at time 2007, (see Figure 14).

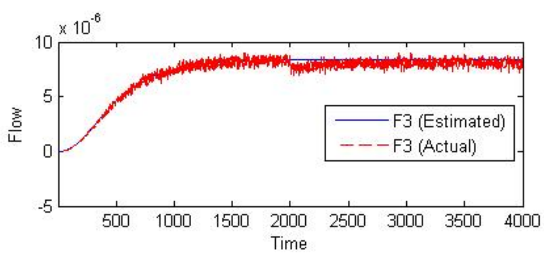

Figure 14: Behaviour (observed variable) of the DBN of PC3 for a fault in the resistance out of tank 3.

In this case, we have 5 single fault candidates (C2, C3, Rv2, Rv23 and Rv3) and we have to run the faulty DBNs from PC3 for all these faults to check which one converges.

In Figure 15 we can see how the DBN from PC3 with the extra node for the fault in Rv3 is able to track the state variables and also gives us a pretty good estimation of the parameter after the fault. Convergence time is 378 (328 after fault injection).

On the other hand, checking the behavior of the other DBNs launched to identify this fault we can see charts like the ones in Figure 16 where it is shown how the DBN of PC3 with a fault in C2 is not able to track the system behavior, as it was expected. It does not converge because it does not track the actual fault.

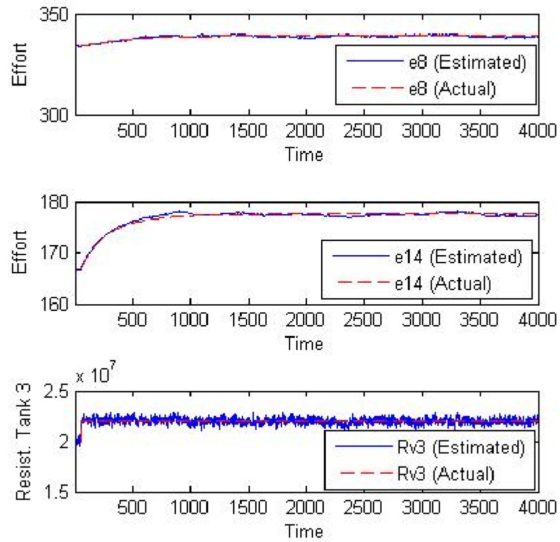

Figure 15: State variables tracked with the DBN of PC3 for a fault in resistance out of tank 3. The last chart at the bottom is the estimation of the parameter Rv3.

6 COMPARING COMPLETE DBN AND DBN FACTORS PERFORMANCE

This section presents a quantitative comparison of the performance of the DBN of the complete system and the performance of the DBNs factors. For the two considered abrupt faults, C1 and Rv3, mean detection, execution and convergence times are computed. Parameter value convergence is also considered. All experiments have been repeated ten times. Table 4, row *Execution Time*, shows execution time for 10.000 time steps. As it was to be expected, DBN factors require less computation time that the original DBN. Faults are injected at time 2000. Rows *C1A* and *Rv3A* shows that there are no false positive detections and that detection time is similar for every network considered.

Table 4: Mean execution and detection time for nominal DBNs (standard deviation in brackets).

	Complete	**PC1**	**PC2**	**PC3**
Execution Time	84.2495	71.1413	77.7708	78.402
C1A +0.1	2001 (0)	2002 (0)	2001 (0)	
Rv3A +0.1	2008 (0)			2007 (0)

For fault identification, simulation starts 50 time steps before the fault is injected and simulation time extends to 8.050 seconds. Table 5, first two rows, show execution time for faulty networks, that are also smaller for DBN factors. Second two rows show convergence time. Compared with the complete DBN, convergence time is smaller for the fault in capacitance of tank 1, and it is equal for the fault in resistance Rv3.

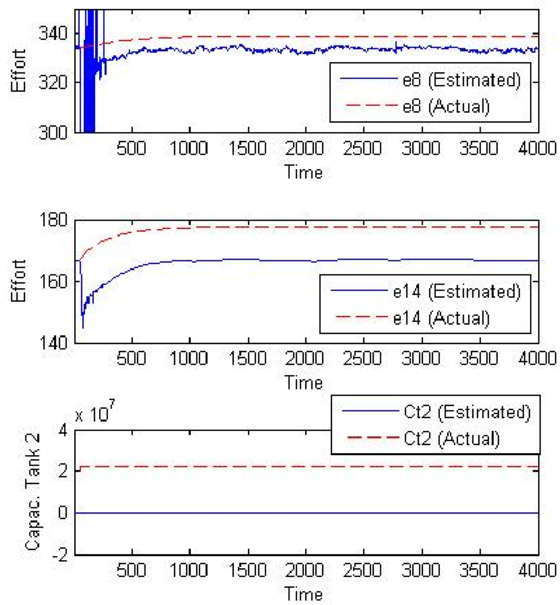

Figure 16: State variables tracked with the DBN of PC3 for a fault in the capacitance of tank 2. The last chart at the bottom is the estimation of the parameter Ct2. The data used to simulate the DBN are from a fault in Rv3.

Finally, Table 6 shows the Mean Square Error (MSE) of the estimated parameter. Error is smaller for PC1 and PC2 for faults in capacitance of tank 1, but it is slightly bigger for PC3 for fault of resistance Rv3.

Table 5: Mean execution and convergence time for faulty DBNs (standard deviation in brackets).

	Complete	**PC1**	**PC2**	**PC3**
Exec. Time C1	46.6603	43.4288	45.4047	
Exec. Time Rv3	67.1525			63.3121
C1A +0.1	$1.15 \cdot 10^3$ $(2.1 \cdot 10^3)$	$3.89 \cdot 10^2$ $(2.8 \cdot 10^1)$	$4.95 \cdot 10^2$ $(1.16 \cdot 10^2)$	
Rv3A +0.1	$3.78 \cdot 10^2$ (1.39)			$3.78 \cdot 10^2$ $(8.82 \cdot 10^{-1})$

7 DISCUSSION

The two faulty scenarios considered in this paper show that PCs decomposition form a state space representation of a system allows using a unique formalism, DBNs, to perform consistency based fault detection, isolation and identification with a simple architecture.

The goal of the paper was not to obtain the most efficient diagnoser, but to present the approach in simple terms. There are several possibilities for improvement, especially to reduce the number of fault hypothesis that have to be tracked. A TRANSCEND like approach (Mosterman and Biswas, 1999), including qualitative

Table 6: Mean Square Error of the estimation for the faulty parameter by each DBN (standard deviation in brackets).

	Complete	**PC1**	**PC2**	**PC3**
C1A +0.1	$2.62 \cdot 10^{-8}$ $(2.4 \cdot 10^{-7})$	$2.06 \cdot 10^{-8}$ $(1.6 \cdot 10^{-8})$	$1.67 \cdot 10^{-8}$ $(7.2 \cdot 10^{-9})$	
Rv3A +0.1	$2.77 \cdot 10^4$ $(3.58 \cdot 10^8)$			$2.81 \cdot 10^4$ $(9.01 \cdot 10^3)$

fault propagation information, can be use to that end. Specially using the Reduce Qualitative Fault Signature Matrix (Bregon *et al.*, 2009), that provides qualitative information to each DBN factor.

Interpretation of quantitative efficiency measures should be done carefully, because final quantitative figures depend on several factors, including fine parameter tuning of the diagnoser. All experiments have been performed with Matlab, with the same Particle Filtering software and on the same machine. Ztest parameters have been selected conservatively, favoring the convergence to the real parameter value against fast fault identification. These initial experimental results indicate that the approach is computationally advantageous even for a small system like the three tank systems. Further research is needed to obtain confident conclusions about parameter value estimation accuracy. The DBN factor from PC2, which keeps all the state variables of the system and a single element in the observational model, provides the best estimation of the new value of the capacity in tank 1. However the DBN factor from PC3, with a simpler structure, estimates the value of Rv3 slightly worse than the DBN of the complete system.

A closely related approach to fault detection, isolation an identification of continuous systems with factored DBNs is presented in (Roychoudhury *et al.*, 2009). Their proposal to obtain DBN factors is based on conditional independence. They define DBN factors as a subset of random variables of the complete DBN, conditionally independent of the variables in all other DBN factors, for a given set of observations. To generate new factors, the main idea is to substitute some of the state variables for new ones worked out as algebraic functions of measurements. The proposed algorithm to generate factors from DBNs can be divided into three main steps: (1) split the system DBN into the maximal number of conditionally independent factors, (2) map each factor with the bond graph factor which models its structure to determine whether it is observable or not and (3) merge every unobservable DBN factor with other DBN factors to get an observable DBN factor. Finally, repeat steps (2) and (3) while needed.

Both approaches have some similarities, like eliminating state variables that can be computed by algebraic relations and assuring that the resulting factors are observable. However, the factoring methodology is different. In (Roychoudhury *et al.*, 2009) network splitting does not consider observability, which has to be recovered later merging unobservable factors with other factors. In contrast, PCs decomposition warranties the observability of the factors, which also assures their conditionally independence. Factoring is more systematic with the PCs decomposition, essentially because structure searching work is made at PCs computation time. Deriving factors from the PCs has the advantage that all minimal factors with analytical redundancy are found. Minimal factor are desirable because they have the potential to maximally reduce computing time on a simulation based approach, particularly for fault identification. Nevertheless, further research is needed to characterize both approaches on that dimensions and to compare their performance on complex, real systems.

8 CONCLUSIONS

This work has presented a method to factor Dynamic Bayesian Networks. A method to generate DBNs factors from Possible Conflicts has been introduced. These factors are minimal redundant structurally observable subsystems. Factoring is desirable because it reduces system complexity, simplifying its analysis and enabling the design of more efficient diagnosers. Structural observability of the factors is needed not only to compute the state variables of the factors, but also to assure their conditional independence of the other minimal factors.

Based on DBNs factors, a unified solution has been proposed to Consistency Based fault detection, isolation and identification. Two scenarios have been developed on the three tank system benchmark to illustrate the proposal. A quantitative comparison of the performance of the DBN of the complete system and the DBNs factors has also been done, in terms of execution, detection, and convergence time plus parameter estimation error. Comparison has confirmed that DBN computation is more efficient for factors than for complete DBNs. Conclusions on parameter value estimation accuracy require additional research.

ACKNOWLEDGMENTS

This work has been partially supported by the Spanish Office of Science and Innovation (MCIN) through grant TIN2009-11326.

REFERENCES

(Alonso-Gonzlez *et al.*, 2008) C. Alonso-Gonzlez, J. J. Rodrguez, O. Prieto, and B. Pulido. Machine learning and model based diagnosis using possible conflicts and system decomposition. In *Proceeding of the 19th International Workshop on Principles of Diagnosis, DX08*, pages 215–222, Blue Mountains, Australia, September 2008.

(Arulampalam *et al.*, 2002) M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174188, 2002.

(Biswas *et al.*, 2003) Gautam Biswas, Gyula Simon, Nagabhushan Mahadevan, Sriram Narasimhan, John Ramirez L, and Gabor Karsai I. A robust method for hybrid diagnosis of complex systems. In *5th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes (SAFEPROCESS*, pages 1125–1131, 2003.

(Bregon *et al.*, 2009) A. Bregon, B. Pulido, G. Biswas, and X. Koutsoukos. Generating possible conflicts from bond graphs using temporal causal graphs. In *Proceeding of the 23rd European Conference on Modelling and Simulation, ECMS09*, Madrid, Spain, 2009.

(Dearden and Clancy, 2001) R. Dearden and D. Clancy. Particle filters for real-time fault detection in planetary rovers. In *the 12th International Workshop on Principles of Diagnosis*, pages 1–6, 2001.

(Gelso *et al.*, 2008) E. R. Gelso, G. Biswas, S. M. Castillo, and J. Armengol. A comparison of two methods for fault detection: a statistical decision, and an interval-based approach. In *Proceeding of the 19th International Workshop on Principles of Diagnosis, DX08*, Blue Mountains, Australia, 2008.

(Koller and Lerner, 2001) D. Koller and U. Lerner. *Sequential Monte Carlos Methods in Practice*, chapter Sampling in factored dynamic systems. Springer, 2001.

(Lerner *et al.*, 2000) Uri Lerner, Ronald Parr, Daphne Koller, and Gautam Biswas. Bayesian fault detection and diagnosis in dynamic systems. In *Prooccedings of the AAAI/IAAI*, page 531537, 2000.

(Mosterman and Biswas, 1999) P. Mosterman and G. Biswas. Diagnosis of continuous valued systems in transient operating regions. *IEEE Transactions on Systems, Man, and Cybernetics*, 29(6):554–565, 1999.

(Moya *et al.*, 2010) N. Moya, G. Biswas, C.J. Alonso-Gonzalez, and X. Koutsoukos. Structural observability. application to decompose a system with possible conflicts. In *Submitted to the 21th International Workshop on Principles of Diagnosis, DX10*, June 2010.

(Murphy, 2002) Kevin Patrick Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, University of California, Berkeley, 2002.

(Narasimhan, 2007) S. Narasimhan. Automated diagnosis of physical systems. In *Proceedings of ICALEPCS07*, pages 701–705, 2007.

(Pulido and Alonso-Gonzalez, 2004) B. Pulido and C. Alonso-Gonzalez. Possible conflicts: a compilation technique for consistency-based diagnosis. *Part B: Cybernetics, IEEE Transactions on Systems, Man, and Cybernetics*, 34(5):2192–2206, Oct. 2004.

(Roychoudhury *et al.*, 2008) I. Roychoudhury, G. Biswas, and X. Koutsoukos. Comprehensive diagnosis of continuous systems using dynamic bayes nets. In *Proceeding of the 19th International Workshop on Principles of Diagnosis, DX08*, Blue Mountains, Australia, September 2008.

(Roychoudhury *et al.*, 2009) I. Roychoudhury, G. Biswas, and X. Koutsoukos. Factoring dynamic bayesian networks based on structural observability. In *In 48th IEEE Conference on Decision and Control (CDC 2009)*, 2009.

(Roychoudhury, 2009) I. Roychoudhury. *Distributed Diagnosis of Continuous Systems: Global diagnosis through local analysis*. PhD thesis, Graduate School of the Vanderbilt University, August 2009.