Carnegie Mellon University

From the SelectedWorks of Ole J Mengshoel

July 2009

Diagnosis and Reconfiguration using Bayesian Networks: An Electrical Power System Case Study

Contact Author

Start Your Own SelectedWorks

Notify Me of New Work

Available at: http://works.bepress.com/ole_mengshoel/25

Diagnosis and Reconfiguration using Bayesian Networks: An Electrical Power System Case Study

W. Bradley Knox
University of Texas at Austin
NASA Ames Research Center
bradknox@cs.utexas.edu

Ole Mengshoel
Carnegie Mellon University
NASA Ames Research Center
ole.j.mengshoel@nasa.gov

Abstract

Automated diagnosis and reconfiguration are important computational techniques that aim to minimize human intervention in autonomous systems. In this paper, we develop novel techniques and models in the context of diagnosis and reconfiguration reasoning using causal Bayesian networks (BNs). We take as starting point a successful diagnostic approach, using a static BN developed for a real-world electrical power system. We discuss in this paper the extension of this diagnostic approach along two dimensions, namely: (i) from a static BN to a dynamic BN; and (ii) from a diagnostic task to a reconfiguration task.

More specifically, we discuss the auto-generation of a dynamic Bayesian network from a static Bayesian network. In addition, we discuss subtle, but important, differences between Bayesian networks when used for diagnosis versus reconfiguration. We discuss a novel reconfiguration agent, which models a system causally, including effects of actions through time, using a dynamic Bayesian network.

Though the techniques we discuss are general, we demonstrate them in the context of electrical power systems (EPSs) for aircraft and spacecraft. EPSs are vital subsystems on-board aircraft and spacecraft, and many incidents and accidents of these vehicles have been attributed to EPS failures. We discuss a case study that provides initial but promising results for our approach in the setting of electrical power systems.

1 Introduction

Automated diagnosis and reconfiguration are important techniques aimed at minimizing human intervention in autonomous systems. In many domains, including both manned and unmanned NASA missions, diagnosis and reconfiguration techniques promise to play an increasing role.

This paper is concerned with the design of automated diagnosis and reconfiguration systems. Following our belief that it is important to consider challenging and significant real-world problems, we demonstrate our approach in an application domain of great interest to NASA, namely, electrical power systems. Electrical power plays an increasing role in aerospace [Bromaghim *et al.*, 2002; Button and Chicatelli, 2005; Poll *et al.*, 2007], and clearly also in terrestrial systems, with a renewed emphasis on upgrading the power grid in the United States and on electrical vehicles.

Unfortunately, electrical power is associated with aerospace incidents and accidents, of which we now mention a few. On September 2, 1998, Swissair flight 111 crashed into the Atlantic Ocean, killing everyone aboard. It was later determined that short-circuited wires most likely led to a catastrophic fire onboard the aircraft. The Electric Propulsion Space Experiment mission, launched and operated in early 1999, ended prematurely when the spacecraft experienced a catastrophic battery failure. On January 14 2005, an Intelsat-operated communications satellite suffered a total loss after a sudden and unexpected electrical power system anomaly. Most likely, the failure was caused by a high-current event in the battery circuitry triggered by an electrostatic discharge. On November 2, 2006 the Mars Global Surveyor last communicated with Earth. It is believed that a software error oriented the spacecraft to an angle that over-exposed it to sunlight, causing the battery to overheat and thereby disabling all communication. On January 7 2008, a Boeing 747 lost main power on its descent into Bangkok because of water entering the generator control unit, and had to use its battery backup. Since this incident occurred while the aircraft was descending into Bangkok, there were no serious consequences, but the loss could have been critical had it taken place over the Pacific ocean.

With the goal of decreasing such incidents and accidents, we develop in this paper novel techniques and models in the context of diagnosis and reconfiguration reasoning using causal Bayesian networks (BNs) [Pearl, 1988; Getoor and Taskar, 2007], and demonstrate them in the EPS setting. This work is part of a larger effort, beyond the scope of this paper, of researching how Bayesian networks can be utilized within an agent framework. In the agent setting, we hypothesize that Bayesian networks will eventually turn out to be useful in integrating diagnosis, prognosis, reconfiguration, and (active) machine learning.

We take as a starting point a successful diagnostic approach, using a static BN developed for a real-world electrical

Yannick Pencolé, Thierry Vidal, and Roberto Micalizio, Editors.
The IJCAI-09 Workshop on Self-* and Autonomous Systems: reasoning and integration challenges (SAS-09), July 13, 2009, Pasadena, California, USA.

power system [Mengshoel *et al.*, 2008; Ricks and Mengshoel, 2009; Mengshoel *et al.*, 2009]. We discuss in this paper the extension of this diagnostic approach along two dimensions, namely: (i) from a static BN to a dynamic BN; and (ii) from a diagnostic task to a reconfiguration task. More specifically, we discuss the auto-generation of a dynamic Bayesian network from a static Bayesian network. In addition, we discuss subtle, but important, differences between Bayesian networks when used for diagnosis versus reconfiguration. We discuss a novel reconfiguration agent, which models a system in a causal manner, including effects of actions through time, using a dynamic Bayesian network. Though the techniques we discuss are general, we demonstrate them in the context of electrical power systems (EPSs) for aircraft and spacecraft. EPSs are, as indicated above, vital subsystems on-board aircraft and spacecraft. We specifically consider the ADAPT EPS, an electrical power system testbed located at the NASA Ames Research Center [Poll *et al.*, 2007]. ADAPT is representative of electrical power systems found in aircraft and spacecraft, and has been developed for the purpose of investigating and benchmarking different aerospace vehicle health management techniques. We additionally discuss a case study on a small subset of the ADAPT EPS that provides initial but promising results for our BN-based approach when applied to electrical power systems in aircraft and spacecraft.

The rest of this paper is structured as follows. In Section 2, we briefly summarize background concepts before discussing our overall approach and algorithms in Section 3. An electrical power system case study is discussed in Section 4, and Section 5 concludes the paper and outlines future work.

2 Background

A Bayesian network (BN) structures a multi-variate probability distribution by using a directed acyclic graph (DAG). Our main emphasis will be on discrete BN nodes. A (discrete) BN node V is a discrete random variable with a mutually exclusive, exhaustive, and finite state space $\Omega_V = \Omega(V) = \{v_1, ..., v_m\}$. We use the notation Π_V for the parents of a node V, Ψ_V for the children of V, and π_V for an instantiation of all parents Π_V of V. The notion of a Bayesian network can now be introduced [Pearl, 1988].

Definition 1 (Bayesian network) *A Bayesian network is a tuple* $(\boldsymbol{V}, \boldsymbol{W}, \boldsymbol{P})$*, where* $(\boldsymbol{V}, \boldsymbol{W})$ *is a DAG with nodes* $\boldsymbol{V} = \{V_1, ...V_n\}$*, edges* $\boldsymbol{W} = \{W_1, ..., W_m\}$*, and where* $\boldsymbol{P} = \{\Pr(V_1 \mid \Pi_{V_1}), \ldots, \Pr(V_n \mid \Pi_{V_n})\}$ *is a set of conditional probability tables (CPTs). For each node* $V_i \in \boldsymbol{V}$ *there is one CPT, which defines a conditional probability distribution* $\Pr(V_i \mid \Pi_{V_i})$.

The independence assumptions induced by $(\boldsymbol{V}, \boldsymbol{W})$ in Definition 1 imply the following joint distribution:

$$\Pr(\boldsymbol{v}) = \Pr(V_1 = v_1, \ldots, V_n = v_n) = \prod_{i=1}^{n} \Pr(v_i \mid \pi_{V_i}), \tag{1}$$

where $\Pi_{V_i} \subset \{V_{i+1}, \ldots, V_n\}$.

A BN can be provided *evidence* by setting or clamping some variables to known states. These nodes are called *evidence variables*. In our setting, evidence variables are commands, sensor readings, or goals (see Section 3 for details). Taking into account the input on evidence variables, different probabilistic queries can be answered [Pearl, 1988]. These probabilistic queries include marginals, most probable explanation (MPE), and maximum aposteriori probability (MAP). Though probabilistic queries can be used for many purposes, in past work on diagnosis (but not reconfiguration) we query health variables representing the health of components, sensors, or both [Mengshoel *et al.*, 2008]. For this work, which includes both diagnosis and reconfiguration, we query action nodes that represent reconfiguration commands after augmenting the evidence with the desired state. Further details are in Sections 3 and 4.

It has been shown that exact MPE computation is NP-hard [Shimony, 1994], and approximating an MPE to within a constant ratio-bound has also been proven to be NP-hard [Abdelbar and Hedetnieme, 1998]. There are two broad classes of approaches to Bayesian network inference: interpretation and compilation. In interpretation approaches, a Bayesian network is directly used for inference. In compilation approaches, such as the clique tree [Lauritzen and Spiegelhalter, 1988; Shenoy, 1989] and arithmetic circuit [Darwiche, 2003; Chavira and Darwiche, 2007] approaches, a Bayesian network is off-line compiled into a secondary data structure, and this secondary data structure is then used for on-line inference. In clique tree clustering, inference consists of propagation in a clique tree compiled from a Bayesian network. In arithmetic circuit evaluation, inference is performed within an arithmetic circuit that was compiled from a Bayesian network. Computation time depends on a number of structural and numerical factors associated with a BN and is not yet, despite recent progress, sufficiently understood.

For the purpose of this paper, we emphasize causal Bayesian networks, which (informally) are BNs that reflect the causal structure of a domain. For example, in an EPS BN, there is an edge from a node representing the opening of a relay to a node representing electrical current flowing through the relay, and not the other way around (i.e., edge direction is the causal direction). BNs have been successfully applied in a wide range of domains, and the techniques developed in this paper are general. However, for illustrative purposes, we consider electrical power systems.

We now briefly consider related research, particularly research that considers Bayesian networks to diagnose and reconfigure. Bayesian networks have in the past been used for model-based diagnosis and monitoring of EPSs [Chien *et al.*, 2002; Yongli *et al.*, 2006; Mengshoel *et al.*, 2008] as well as for other purposes [Andreassen *et al.*, 1987; Lerner *et al.*, 2000; Romessis and Mathioudakis, 2006], however this previous work generally does not integrate diagnosis and reconfiguration by means of the Bayesian network itself. There has also been work in the area of self-configuring systems [Williams and Nayak, 1996; Barrett, 2005], but this has often not been founded on Bayesian networks. Finally, there are decision-theoretic (including planning) approaches [Basye *et al.*, 1992; Haddawy *et al.*, 1995; Boutilier *et al.*, 1999]. Our research is simpler, and hence potentially easier to deploy, since (i) we retrict ourselves to

Yannick Pencolé, Thierry Vidal, and Roberto Micalizio, Editors.
The IJCAI-09 Workshop on Self-* and Autonomous Systems: reasoning and integration challenges (SAS-09), July 13, 2009, Pasadena, California, USA.

finite-horizon reconfiguration and (ii) we do not rely on the non-trivial specification of utilities (or reward function).

3 Architecture and Algorithms

From an abstract point of view, we focus on systems that receive service requests. In the case of an EPS, these requests are for power to reach specific loads (i.e. receivers of electrical power). We are interested in systems which are sufficiently complex for a request (or a set of requests) to be served in different ways. For example, EPSs in aerospace vehicles often have redundancies and alternate routes such that power can be supplied even in case of failure of a subset of the components in an EPS. For the purpose of this work, where we assume a fixed system design, we identify two sets of agents:

1. Resource Requesters $\mathfrak{R} = \{\mathfrak{r}_1, \mathfrak{r}_2, \ldots\}$ - agents that request resources from the system.
2. System Configurators $\mathfrak{C} = \{\mathfrak{c}_1, \mathfrak{c}_2, \ldots\}$ - agents that configure (and reconfigure) the system in an attempt to meet resource requests.

A human user of a system may play the role of both a Resource Requester and a System Configurator. However, as system complexity increases, a human user may have difficulty configuring the system accurately and quickly to meet resource demands. An autonomous system configurator–a self-configuring system–becomes desirable, since it may perform better than the human and free the person to focus on other tasks. Our goal is to create an agent that reconfigures systems. In the context of EPSs, that means accounting for component and sensor failures while also minimizing power loss by not providing power to locations where it has not been requested. Assuming for simplicity $|\mathfrak{C}| = 1$, we define the following problem statement:

The System Configuration problem Given periodic sensory data (state observations) and knowledge of past commands (actions), the System Configurator $\mathfrak{c}_1$ should issue commands to route power to Resource Requesters $\mathfrak{R}$ in the face of changing demands and system faults.

The architecture in Figure 1 reflects this problem.

3.1 Framing the problem as a POMDP

Sequential decision making tasks such as the one described above are commonly modeled as Markov decision processes (MDPs). A finite MDP is specified by the tuple (S, A, T, γ, D, R). S and A are, respectively, the sets of possible states and actions. T is a transition function, $T : S \times A \times S \rightarrow \mathbb{R}$, which gives the probability, given a state and an action, of transitioning to another state on the next time step. γ, the discount factor, exponentially decreases the value of a future reward. D is the distribution of start states. R is a reward function, $R : S \times S \rightarrow \mathbb{R}$, where the reward is a function of states s_t and s_{t+1}. A partially observable Markov decision process (POMDP) describes a task for which the agent cannot see the true environmental state, but rather can see an observation (from the set O of possible observations) that reflects the true state and is generated with probability $P(O \mid S)$. We use a more specific definition of observations: an observation o is simply the observable portion of the state s (i.e., O is a subset of the state variables in S). Additionally, for expository purposes, the hidden portion of some state s is called h in this paper.

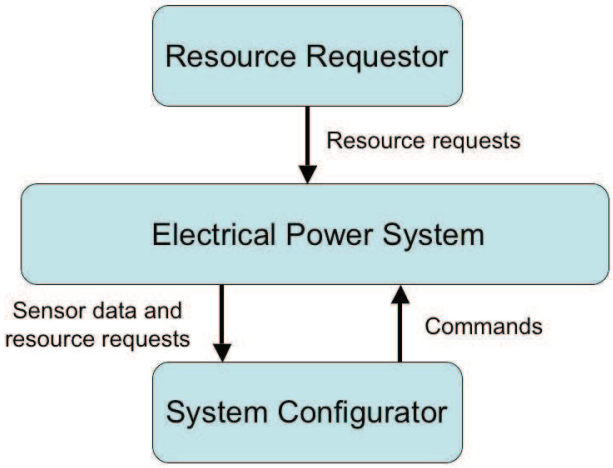

Figure 1: Diagram showing the interaction between the System Configurator, the EPS, and the Resource Requester.

Relating the POMDP formalism to our Bayesian network formalism the set of possible states S is the Cartesian product of all non-command nodes $V_1, ..., V_n$. (i.e., $S = V_1 \times ... \times V_n$). A is the Cartesian product of all command nodes.

Consider the problem statement above. Since the agent has sensors that may not be reliable and often only indirectly measure important states, we cast this problem as a POMDP. However, in Section 3.3, we describe how it departs from the formal POMDP specification.

3.2 Generation of a dynamic BN

In our previous work on designing a diagnostic algorithm for an EPS, a large static Bayesian network was hand-created using knowledge of dependencies within the system's physical structure and documented error rates for individual components [Mengshoel *et al.*, 2008]. Dynamic bayesian networks have previously been used to model POMDPs with success [Boutilier and Poole, 1996; Toussaint *et al.*, 2006]. In a very related study, Kohda and Ciu use dynamic Bayesian networks to model and reconfigure a simulated safety monitoring system [2007]. Following their leads, we created a dynamic Bayesian network from the hand-coded static Bayesian network and used it to model the ADAPT EPS through time.

To create a dynamic Bayesian network from the static Bayesian network which is used for fault diagnosis, we created the following GENERATEDBN algorithm:

1. Replicate the static Bayesian network for the number of time steps that will be expressed by the dynamic Bayesian network. For each node x in the static BN, label its duplicates $x_1, x_2, ..., x_n$ for the n time steps being expressed.
2. For any command node y_n at time n that has an edge to a node x_n, remove that edge and replace it with an edge to x_1. Relabel the command node as y_{-1}.

Yannick Pencolé, Thierry Vidal, and Roberto Micalizio, Editors.
The IJCAI-09 Workshop on Self-* and Autonomous Systems: reasoning and integration challenges (SAS-09), July 13, 2009, Pasadena, California, USA.

3. For all other command nodes at some time t that have an edge to a node x_t, remove that edge and replace it with an edge to x_{t+1}.
4. For a node that describes the health of a component at time t, create an edge from that node to its duplicate node at time $t+1$.

We argue that the dynamic BN created by GenerateDBN is a BN. By assumption, the input BN is a BN. Since all edges in the dynamic BN either come from the input BN or are added by GenerateDBN, and in the latter case go from slice i to slice i+1, acyclicity is preserved. In addition, CPTs are added for all nodes in the dynamic BN that do not have CPTs defined by the static BN.

For this work, we create a dynamic Bayesian network with two time slices ($n = 2$). Henceforth, all discussion of the dynamic network assumes that $n = 2$. Given the large number of nodes, converting the static Bayesian Network to an appropriate dynamic Bayesian Network by hand was not an option, so we created a script that automatically made the described changes.

3.3 Reconfiguration algorithm using the dynamic BN

We define an action by the System Configurator to be any subset of the set of all possible commands to the system. Commands give orders to open or close either relays or switches. At time t, we call the current command subset a_t. Additionally, o_t is the agent's observation as provided by sensors within the EPS, and h_t is the hidden state[1].

Whether or not power is being supplied to all loads as requested by the Resource Requester is described by boolean variable g_t. Here, g_t is based on a one-to-one mapping between the states of a command node used by a Resource Requestor and the corresponding Resource Requestor status (or goal) node. In Figure 4, for example, the node *Command_Relay_3* is a Resource Requestor command node, while the *Voltage_Relay_3* is the corresponding Resource Requestor status node. Here, the one-to-one mapping is as follows: *cmdOpen* maps to *voltageLo*, *cmdClose* maps to *voltageHi*. Suppose that there are m such status nodes $\boldsymbol{S} = \{S_1, \ldots, S_m\}$, corresponding to m Resource Requestor command nodes $\boldsymbol{R} = \{R_1, \ldots, R_m\}$. Then $g_t = \{(S_1, s_1), \ldots, (S_m, s_m)\}$ represents the conjunction of requests. Note that g_t is not a part of the formal POMDP definition, and also that there is a small but important distinction between a status node and a sensor node. In particular, our formalization allows for a sensor reading to have low posterior probability for a given g_t, but typically a sensor downstream of a Resource Requestor command node set to *cmdClose* will reflect the command's state with high probability.

Taking our example further, suppose that power (or voltage) is requested for the load in Figure 4. In other words, *Command_Relay_3* is set to *cmdClose* by the Resource Requestor, leading to *Voltage_Relay_3* being set to *voltageHi*, i.e. $g_t = \{(\textit{Voltage_Relay_3}, \textit{voltageHi})\}$, since there is only one load, a light bulb. Given the appropriate command being issued by the System Configurator (upstream of *Command_Relay_3*), and components downstream of *Voltage_Relay_3* being healthy, the load's sensor reading should reflect the power that has been requested, i.e. $\Pr(\textit{Sensor_Light_1} = \textit{readLightHi}) >> \Pr(\textit{Sensor_Light_1} = \textit{readLightLo})$.

The conditions for g_t (power supplied where requested) are dictated by the Resource Requester at time $t-1$. o_t, the sensor information at time t, determines whether those conditions have been satisfied. Since the agent does not perform policy optimization, the environmental reward of the POMDP remains unspecified and is not a part of the Bayesian network. Instead, we use g_t to define the agent's goal. Thus our formulation departs from the POMDP framework. However, in future work, policy optimization will be done by converting g_t into a reward function, so thinking in terms of a POMDP from the start may be useful.

From a probabilistic perspective, the optimal action is $argmax_{a_t} P(g_{t+1} = true \mid a_t, h_t, o_t)$. In both our static and dynamic Bayesian networks, each observation node has a causal link(s) from some subset of the hidden state nodes, which simplifies the optimal action to $argmax_{a_t} P(g_{t+1} = true \mid a_t, h_t)$. For n Boolean commands, there are 2^n possible actions. Unfortunately, for large EPSs with a large number of possible commands, iterating through all possible actions is infeasible. The ADAPT EPS, for example, can support 24 Boolean commands at a time, yielding over ten million possible actions. Calculating the above probability for one of these actions is non-trivial, so ten million such calculations could not be done in a real-time system requiring reconfiguration. As a more tractable alternative, we approximate $argmax_{a_t} P(g_{t+1} = true \mid a_t, h_t)$ by calculating $argmax_{a_t} P(a_t \mid g_{t+1}, h_t)$.

Though it may seem that this approach would have the same problem of needing to iterate through an exponential number of actions, we instead calculate the maximum a posteriori probability (MAP) over all possible actions. To calculate the a posteriori probability $P(g_{t+1} \mid a_t, h_t)$ for a single action, the agent would also have used a MAP calculation, so we are reducing the computational complexity by an exponential factor.

Unfortunately, the MAP inference problem is complete for the complexity class NP^{PP} [Park and Darwiche, 2004], but can often be performed in reasonable time for small Bayesian networks such as the example network described in Section 4. For large but sparsely linked networks like the one created for the ADAPT EPS, we instead calculate the most probable explanation (MPE), or $argmax_{a_t, o_{t+1}, h_{t+1}} P(a_t \mid g_{t+1}, h_t)$ (i.e., maximizing over all unknown nodes in the network). There is sufficient precedent for this, considering that MPE is commonly used as an approximation for the more difficult MAP calculation. Computing MPE for an arbitrary Bayesian network is in the exponential time complexity class, but for sparsely linked, large networks, it can nonetheless be calculated in real time.

Until this point, we have assumed that h_t is known, which is untrue. Instead, the agent calculates an approximation of h_t called $\hat{h}_t$ based on Bayesian inference conditioned on o_t,

[1] o_t and h_t together define the full state s_t.

Yannick Pencolé, Thierry Vidal, and Roberto Micalizio, Editors.
The IJCAI-09 Workshop on Self-* and Autonomous Systems: reasoning and integration challenges (SAS-09), July 13, 2009, Pasadena, California, USA.

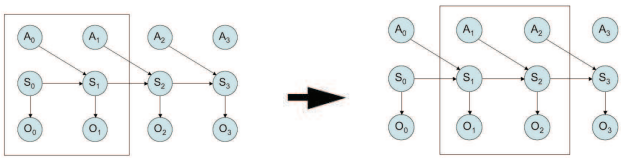

Figure 2: Windowing in a simple dynamic Bayesian network. The moving rectangle encloses the actual dynamic BN.

a_{t-1}, and $\hat{h}_{t-1}$.

At a high level, the agent's algorithm, INFERDBN, is as follows:

1. Receive observation o_t. Fix the corresponding node states.
2. Perform inference on the Bayesian network (or arithmetic circuit) to estimate hidden state $\hat{h}_t$, using o_t, a_{t-1}, and the previous estimate $\hat{h}_{t-1}$. Fix the node states corresponding to $\hat{h}_t$
3. Slide the window of the dynamic Bayesian network from time steps $t-1$ and t to time steps t and $t+1$, forgetting data held in time step $t-1$ (this step, often called "windowing", is illustrated in Figure 2).
4. Fix the subset of the nodes for o_{t+1} which indicate which loads are receiving power (from g_t).
5. Perform MAP inference (or approximation of MAP by MPE) on the arithmetic circuit to determine most likely command set to have caused the desired future state ($argmax_{a_t} P(a_t|o_t, g_{t+1}, \hat{h}_t)$).
6. Issue maximizing command set a_t to the EPS
7. Unfix all previously fixed nodes for o_{t+1}.

4 Case Study

Electrical power systems (EPSs) play a critical role in aerospace. More specifically, EPS loads (i.e. receivers of electrical power) include avionics, propulsion, life support, and thermal management. We now describe a small EPS and how it may be formalized into a Bayesian network. The EPS has some redundancy, as is typical for EPSs used in aerospace, such that it supports reconfiguration. In this particular example, we assume that there are two batteries and one load. At most one battery is allowed to power the load at any one time, and we assume that at the current time, the connected battery is failing to power the load. Then, a request for power arrives (i.e., the relay before a load is turned on). The System Configurator agent must now determine what commands to give that will maximize the chance of power reaching the load where it has been requested.

To illustrate this scenario, we consider the small EPS shown in Figure 3. This figure shows how a load *Load1* can be powered from either *Battery1* or *Battery2* (note that both of them should not power the load at the same time, since this might cause a backward flow of energy from one battery into the other). Suppose that *Relay3* is closed, representing a request for power from $\mathfrak{R}$. At the same time, *Relay1* is closed, *Relay2* is open, but *Light1* is low. This status is suggesting

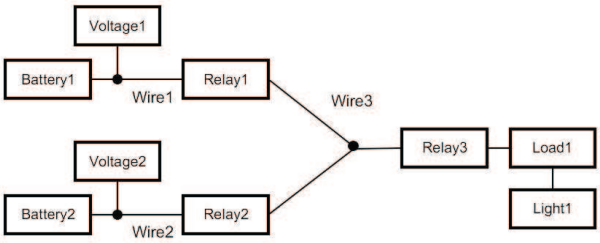

Figure 3: The electrical power system for this case study.

that *Load1* is not powered by *Battery1*, after all, even though it should be. Further, suppose that *Voltage1* is low and *Voltage2* is high, thus there is the potential for the load to be powered from an alternative energy source. Can we demonstrate that our INFERDBN algorithm computes that *Relay1* should be open, and *Relay2* should be closed? Note that this involves both diagnosis and reconfiguration, since the algorithm both needs to figure out what is wrong and work around the problem.

This small EPS can be represented as a static BN as shown in Figure 4. Compared to an ADAPT BN with 503 nodes, this BN has been substantially simplified to illustrate our main points. In this BN, nodes that have a prefix of *Sensor* or *Command* are evidence (i.e., input) nodes, while those with a prefix of *Health* are query (i.e., output) nodes. Nodes that have a prefix of *Closed* or *Voltage* represent, respectively, the state of a part (such as a battery, a wire, a relay, or a load) or the measured voltage. Broadly speaking, this BN represents how power from the batteries, on the left in the figure, "flows" to the load, on the right in the figure, in a left-to-right pattern. More specifically, the *Command* nodes control, through *Closed* nodes, how voltage propagates. In addition, there is a constraint between Relays1 and Relay2, expressing the fact that both should not be closed at the same time.

From this static BN, a DBN with two time slices was generated using the GENERATEDBN algorithm presented above. We now discuss how this DBN, shown fully in Figure 6, can be used in our case study.

Phase 1: We have commands and sensor readings as shown in Figure 5. Note how the posterior over *Health_Battery1_t_0* suggests that *Battery1* is not working, while the posterior *Health_Battery2_t_0* suggests that *Battery2* is working. These posteriors are very reasonable, given the commands given and the sensor readings.

Phase 2: A Resource Requester expresses a need for *Current_Load1_t_1* = currentHi.

Phase 3: Taking into account the state shown in Phase 1 as well as request in Phase 2, the MPE is computed, with the result shown in Figure 6. Note in particular that *Command_Relay1_t_0* = cmdOpen and *Command_Relay2_t_0* = cmdClose, the opposite of what was the case in Phase 1. This reflects the suspected poor health of Battery1 and good health of Battery2.

5 Conclusion and Future Work

The long-term objective of this research is to enable integrated diagnosis and reconfiguration to improve system au-

Yannick Pencolé, Thierry Vidal, and Roberto Micalizio, Editors.
The IJCAI-09 Workshop on Self-* and Autonomous Systems: reasoning and integration challenges (SAS-09), July 13, 2009, Pasadena, California, USA.

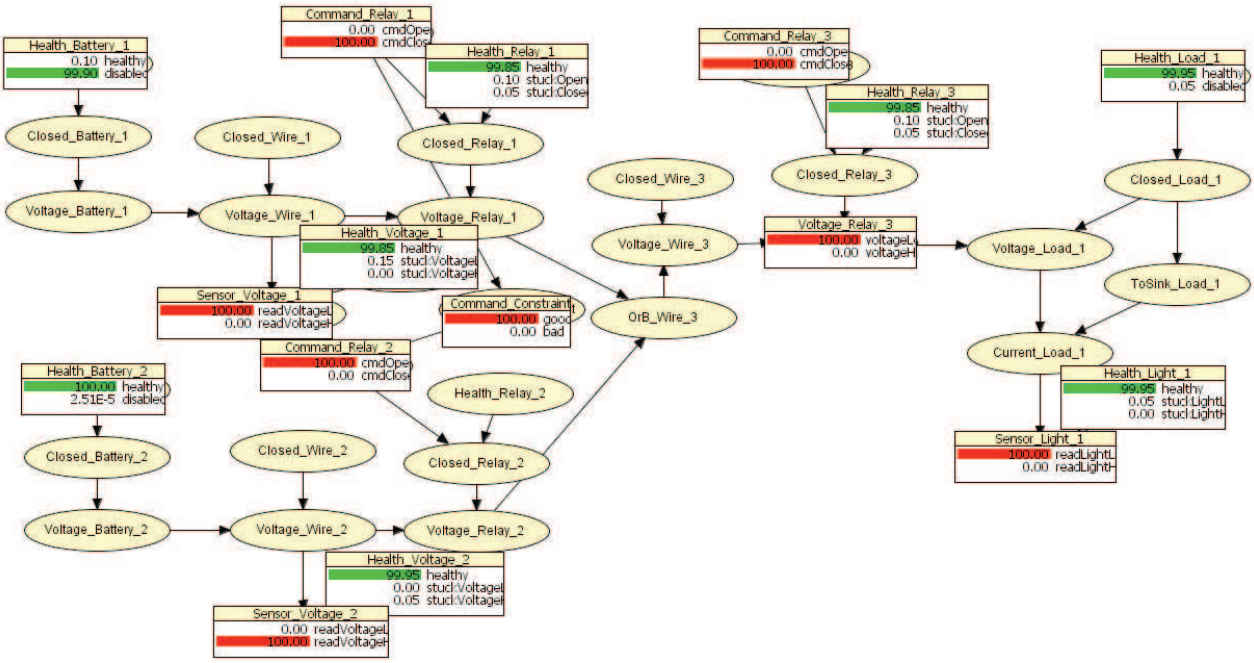

Figure 4: The static BN used in this case study before converting to a dynamic BN. The boxes that are covering their corresponding nodes show the probability distribution for that node. Boxes with green highlighting derive their distributions from Bayesian inference, and boxes with red highlighting are clamped to states known through evidence or desired states.

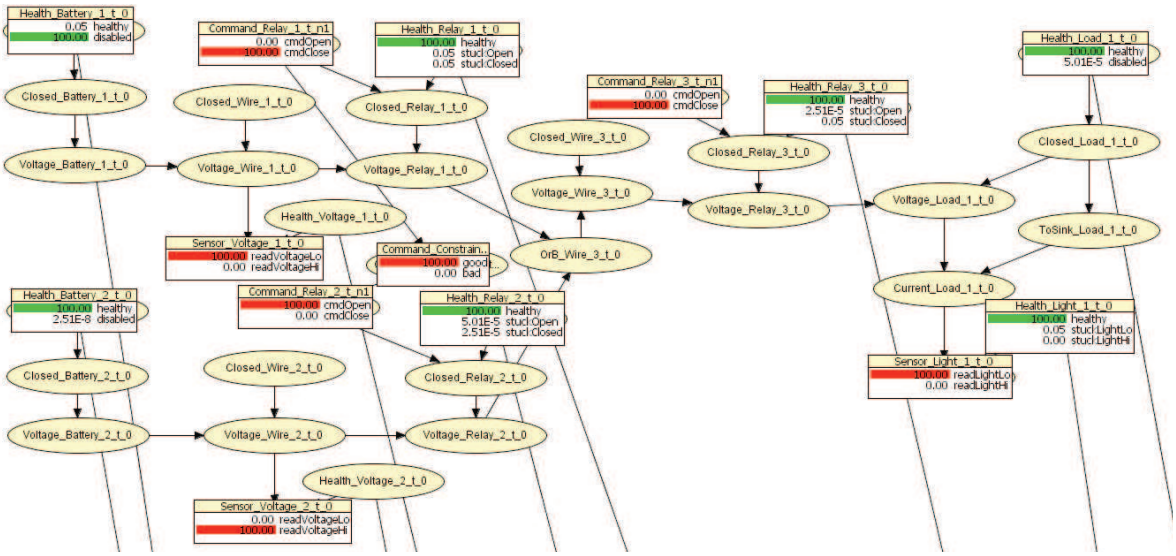

Figure 5: The half of our dynamic Bayesian network that corresponds to the current time step ($t = 0$). Note that the posterior over *Health_Battery1_t_0* suggests that Battery1 is not working, while the posterior *Health_Battery2_t_0* suggests that Battery2 is working.

Yannick Pencolé, Thierry Vidal, and Roberto Micalizio, Editors.
The IJCAI-09 Workshop on Self-* and Autonomous Systems: reasoning and integration challenges (SAS-09), July 13, 2009, Pasadena, California, USA.

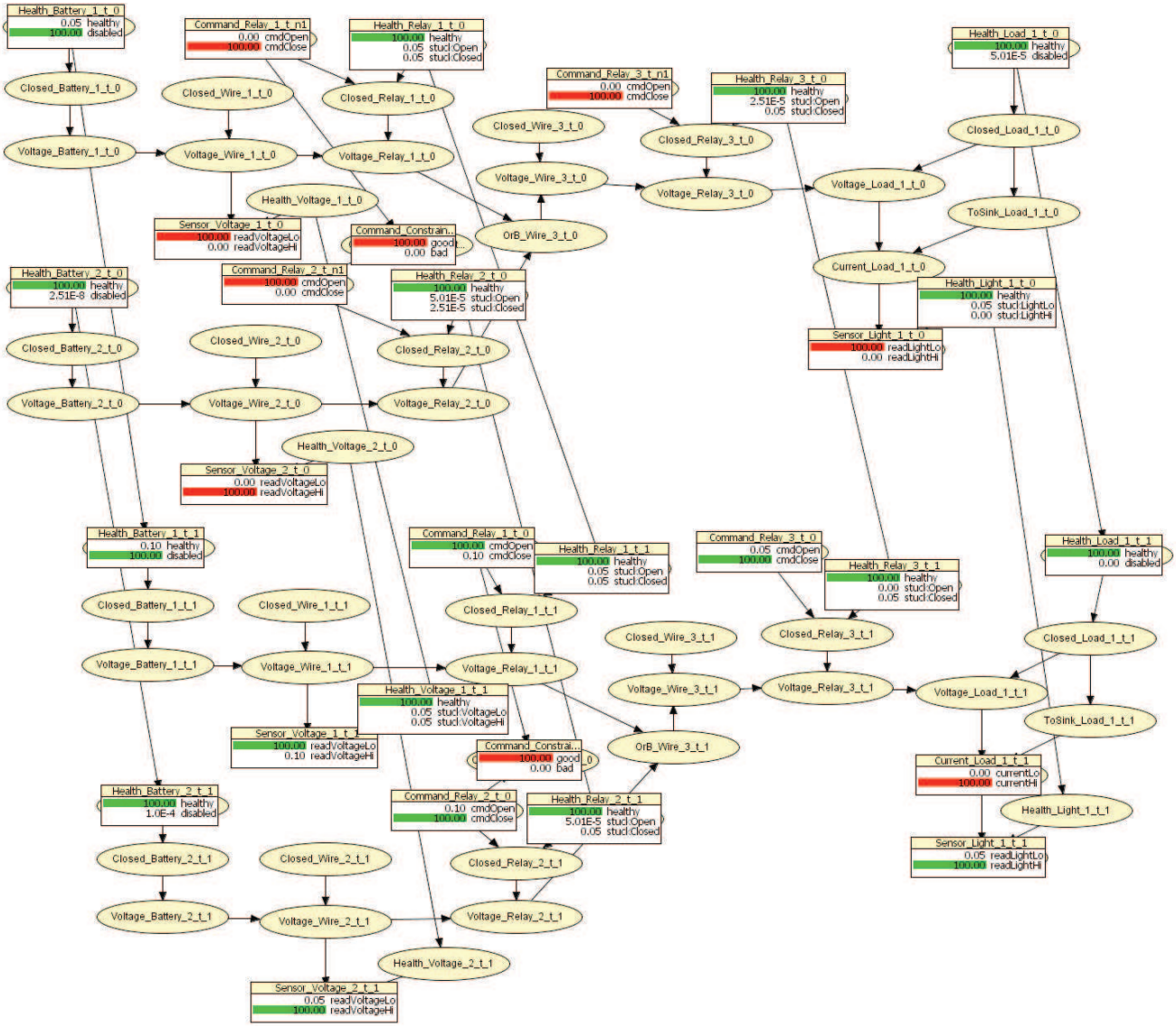

Figure 6: The Bayesian network after it infers the commands a_t such that $argmax_{a_t} P(a_t|o_t, g_{t+1}, \hat{h}_t)$. Note that *Command_Relay1_t_0* = cmdOpen and *Command_Relay2_t_0* = cmdClose, the opposite of what was the case in Phase 1.

Yannick Pencolé, Thierry Vidal, and Roberto Micalizio, Editors.
The IJCAI-09 Workshop on Self-* and Autonomous Systems: reasoning and integration challenges (SAS-09), July 13, 2009, Pasadena, California, USA.

tonomy. As a step towards that objective, in this paper we reported progress in the following two areas. First, we developed an approach to automatically generate a dynamic BN from a static BN for a certain class of static BNs. Second, utilizing the generated dynamic BN, we discussed an approach for a System Configurator agent to handle requests from a Resource Requester.

Our results are preliminary. In future work we plan to first perform extensive reconfiguration experiments on electrical power system testbeds such as the ADAPT testbed, and then further develop the reconfiguration system with an eye towards deployment on an aerospace vehicle. Another interesting direction of further research is to extend the approach with learning capabilities, having the agent use past experience to adjust the values within the dynamic Bayesian network's conditional probability tables.

References

[Abdelbar and Hedetnieme, 1998] A. M. Abdelbar and S. M. Hedetnieme. Approximating MAPs for belief networks is NP-hard and other theorems. *Artificial Intelligence*, 102:21–38, 1998.

[Andreassen *et al.*, 1987] S. Andreassen, M. Woldbye, B. Falck, and S.K. Andersen. MUNIN – A causal probabilistic network for interpretation of electromyographic findings. In *Proceedings of IJCAI-87*, pages 366–372, August 1987.

[Barrett, 2005] A. Barrett. Model compilation for real-time planning and diagnosis with feedback. In *Proceedings of IJCAI-05*, 2005.

[Basye *et al.*, 1992] Kenneth Basye, Thomas Dean, Jak Kirman, and Moises Lejter. A decision-theoretic approach to planning, perception, and control. *IEEE Expert*, 7(4):58–65, August 1992.

[Boutilier and Poole, 1996] C. Boutilier and D. Poole. Computing optimal policies for partially observable decision processes using compact representations. In *Proceedings of the National Conference on Artificial Intelligence*, pages 1168–1175, 1996.

[Boutilier *et al.*, 1999] C. Boutilier, T. Dean, and S. Hanks. Decision-theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research*, 11:1–94, 1999.

[Bromaghim *et al.*, 2002] D. R. Bromaghim, J. R. Leduc, R. M. Salasovich, G. G. Spanjers, J. M. Fife, M. J. Dulligan, J. H. Schilling, D. C. White, and L. K. Johnson. Review of the electric propulsion space experiment (ESEX) program. *Journal of Propulsion and Power*, 18(4):723–730, 2002.

[Button and Chicatelli, 2005] R. M. Button and A. Chicatelli. Electrical power system health management. In *Proceedings of the 1st International Forum on Integrated System Health Engineering and Management in Aerospace*, Napa, CA, 2005.

[Chavira and Darwiche, 2007] M. Chavira and A. Darwiche. Compiling Bayesian networks using variable elimination. In *Proceedings of IJCAI-07*, pages 2443–2449, Hyderabad, India, 2007.

[Chien *et al.*, 2002] C.-F. Chien, S.-L. Chen, and Y.-S. Lin. Using Bayesian network for fault location on distribution feeder. *IEEE Transactions on Power Delivery*, 17:785–793, 2002.

[Darwiche, 2003] A. Darwiche. A differential approach to inference in Bayesian networks. *Journal of the ACM*, 50(3):280–305, 2003.

[Getoor and Taskar, 2007] L. Getoor and B. Taskar, editors. *Introduction to Statistical Relational Learning*. The MIT Press, August 2007.

[Haddawy *et al.*, 1995] Peter Haddawy, AnHai Doan, and Richard Goodwin. Efficient decision-theoretic planning: Techniques and empirical analysis. In *Proceedings of the Eleventh Annual Conference on Uncertainty in Artificial Intelligence (UAI-95)*, pages 229–236, Montreal, Canada, 1995.

[Kohda and Cui, 2007] T. Kohda and W. Cui. Risk-based reconfiguration of safety monitoring system using dynamic Bayesian network. *Reliability Engineering and System Safety*, 92(12):1716–1723, 2007.

[Lauritzen and Spiegelhalter, 1988] S. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems (with discussion). *Journal of the Royal Statistical Society series B*, 50(2):157–224, 1988.

[Lerner *et al.*, 2000] U. Lerner, R. Parr, D. Koller, and G. Biswas. Bayesian fault detection and diagnosis in dynamic systems. In *Proceedings of AAAI-00*, pages 531–537, 2000.

[Mengshoel *et al.*, 2008] O. J. Mengshoel, A. Darwiche, K. Cascio, M. Chavira, S. Poll, and S. Uckun. Diagnosing faults in electrical power systems of spacecraft and aircraft. In *Proceedings of IAAI-08*, pages 1699–1705, Chicago, IL, 2008.

[Mengshoel *et al.*, 2009] O. J. Mengshoel, M. Chavira, K. Cascio, S. Poll, A. Darwiche, and S. Uckun. Probabilistic model-based diagnosis: An electrical power system case study. *IEEE Trans. on Systems, Man, and Cybernetics*, 2009. Accepted for publication.

[Park and Darwiche, 2004] J. D. Park and A. Darwiche. Complexity results and approximation strategies for MAP explanations. *Journal of Artificial Intelligence Research (JAIR)*, 21:101–133, 2004.

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

[Poll *et al.*, 2007] S. Poll, A. Patterson-Hine, J. Camisa, D. Garcia, D. Hall, C. Lee, O. J. Mengshoel, C. Neukom, D. Nishikawa, J. Ossenfort, A. Sweet, S. Yentus, I. Roychoudhury, M. Daigle, G. Biswas, and X. Koutsoukos. Advanced diagnostics and prognostics testbed. In *Proceedings of the 18th International Workshop on Principles of Diagnosis (DX-07)*, pages 178–185, Nashville, TN, 2007.

[Ricks and Mengshoel, 2009] B. W. Ricks and O. J. Mengshoel. The diagnostic challenge competition: Probabilistic techniques for fault diagnosis in electrical power systems. In *Proceedings of the 20th International Workshop on Principles of Diagnosis (DX-09)*, Stockholm, Sweden, 2009.

[Romessis and Mathioudakis, 2006] C. Romessis and K. Mathioudakis. Bayesian network approach for gas path fault diagnosis. *Journal of engineering for gas turbines and power*, 128(1):64–72, 2006.

[Shenoy, 1989] P. P. Shenoy. A valuation-based language for expert systems. *International Journal of Approximate Reasoning*, 5(3):383–411, 1989.

[Shimony, 1994] E. Shimony. Finding MAPs for belief networks is NP-hard. *Artificial Intelligence*, 68:399–410, 1994.

[Toussaint *et al.*, 2006] M. Toussaint, S. Harmeling, and A. Storkey. Probabilistic inference for solving (PO)MDPs. *Institute for Adaptive and Neural Computation*, 2006.

[Williams and Nayak, 1996] B. C. Williams and U. Nayak. A model-based approach to reactive self-configuring systems. In *Proceedings of AAAI-96*, pages 971–978, 1996.

[Yongli *et al.*, 2006] Z. Yongli, H. Limin, and L. Jinling. Bayesian network-based approach for power system fault diagnosis. *IEEE Transactions on Power Delivery*, 21:634–639, 2006.

Yannick Pencolé, Thierry Vidal, and Roberto Micalizio, Editors.
The IJCAI-09 Workshop on Self-* and Autonomous Systems: reasoning and integration challenges (SAS-09), July 13, 2009, Pasadena, California, USA.