# A Risk-Based Approach to Prognostics and Health Management Combining Bayesian Networks and Continuous-Time Bayesian Networks

*Jordan Schupbach, Elliott Pryor, Kyle Webster, and John Sheppard*

Performing general prognostics and health management (PHM), especially in electronic systems, continues to present significant challenges. The low availability of failure data makes learning generalized models difficult and constructing generalized models during the design phase often requires a level of understanding of the failure mechanisms that elude the designers. In this paper, we present a generalized approach to PHM based on two types of probabilistic models, Bayesian Networks (BNs) and Continuous-Time Bayesian Networks (CTBNs), and we pose the PHM problem from the perspective of risk mitigation rather than failure prediction. This paper also constitutes an extension of previous work where we proposed this framework initially [1]. In this extended version, we also provide a comparison of exact and approximate sample-based inference for CTBNs to provide practical guidance on conducting inference using the proposed framework.

## Introduction

In previous work, we developed a diagnostic modeling tool using BNs called the Standards-based Analysis Platform for Predictive Health and Integrated Reasoning Environment (SAPPHIRE) [2], which conforms to IEEE Std 1232-2010 (AI-ESTATE) [3]. We also developed a modeling tool designed for prognostics called the Continuous-time Hazard Analysis and Risk Mitigation (CHARM) system [4]. The models used in CHARM are based on CTBNs [5], which represent systems as factored continuous-time conditional Markov processes.

Although these two models are natural to combine, there is little to no application of these models used in combination for conducting diagnostics and prognostics under a single modeling framework. In this work, we discuss an approach to combining their use for PHM. Our intent is not to focus on SAPPHIRE or CHARM specifically, but rather to discuss how BNs and CTBNs can be used together to support PHM. Therefore, we use SAPPHIRE and CHARM for example purposes only. Ultimately, this paper is about describing a new process for risk-based PHM that combines elements of diagnostics and health state information as a starting point from which predictive diagnostics (i.e., prognostics) can then be performed.

## Background

Here, we provide background necessary to follow the method presented in this paper. First, we define what we mean by Prognostics and Health Management (PHM) relative to current views in the industry. We then present the main tools employed in our approach.

### *Prognostics and Health Management*

Simply put, there is little agreement about the scope and relevant practice of PHM. We take a literal approach when considering PHM in that we believe PHM must include both state estimation (health management) and prediction (prognostics). This is contrary to many who believe the focus is on health management as a practice of diagnostics and condition-based maintenance, which largely centers on state estimation.

Vichare and Pecht noted that "The term 'diagnostics' pertains to the detection and isolation of faults or failures. 'Prognostics' is the process of predicting a future state (of reliability) based on current and historic conditions. Prognostics and health management (PHM) is a method that permits the reliability of a system to be evaluated in its actual life-cycle conditions, to determine the advent of failure, and mitigate the system risks [6]." We see both inspiration and limitation in this view. As inspiration, we see that we can use reliability information during the design phase to create initial predictive models and consider risks associated with system failure. However, as limitation, there is no tie between diagnostics and prognostics in this view of PHM.

This paper contains extended research originally presented at IEEE AUTOTESTCON 2022 and recognized as Best Student Paper (© IEEE 2022, used with permission, [1]).



1094-6969/23/$25.00©2023IEEE

Kalgren *et al.* also provide a definition of PHM. They say PHM is "a health management approach utilizing measurements, models, and software to perform incipient fault detection, condition assessment, and failure progression prediction" [7]. Their view includes incipient fault detection and condition assessment, which ties back to the current health state of the system. However, their views related to failure progression prediction largely depend upon physics-of-failure models, which are neither generalizable nor scalable in complex systems.

Li *et al.* pose PHM more literally, as we do. "Prognostic and Health Management (PHM) systems support aircraft maintenance through the provision of diagnostic and prognostic capabilities, leveraging the increased availability of sensor data on modern aircraft. Diagnostics provide the functionalities of failure detection and isolation, whereas prognostics can predict the remaining useful life (RUL) of the system" [8]. In this definition, diagnostics are limited to on-board systems, and prognostics are focused on RUL. We adapt this idea to consider off-board diagnostics and time-to-failure.

We also consider the ideas expressed in the recently approved IEEE Standard 1856, which divides the definition of PHM into two parts [9]. First, the standard defines prognostics to be "the process of predicting an object system's RUL by predicting the progression of a fault given the current degree of degradation, the load history, and the anticipated future operational and environmental conditions to estimate the time at which the object system will no longer perform its intended function within the desired specifications." Once again, the focus is on remaining useful life and on failure progression, which would largely be from a point of failure perspective. Second, the standard defines health management as "The process of decision-making and implementation of actions based on the estimate of the state of health derived from health monitoring and expected future use of the system." This is good in the sense that the dependence is on state of health, but the definition excludes the health assessment itself.

We have previously asserted that all aspects of health assessment, including fault detection, localization, isolation, and even determining there are no faults, are diagnostic processes [10]. We assert that PHM begins with diagnosis and then proceeds to determine when future failures might occur (prognosis). We like to refer to prognostics as predictive diagnostics in that we also want to know what faults are occurring when. This sets up a pipeline process whereby PHM consists of a sequence of five steps: monitoring; health state assessment (diagnosis); prediction (prognosis); assessment; and action. This results in an evidence-based decision-making process that leads to the overall support of the system.

## Risk-based PHM

Motivated by Vichare and Pecht, who draw on reliability information, we employ a "risk-based" approach to PHM (rPHM). We seek to introduce a framework that includes both diagnostics and prognostics and incorporates effects or hazards using the same model semantics. By building hazards into the model, predictions can be made about the risks associated with likely faults and downstream results of those faults. Our approach incorporates user-specified performance functions (i.e., utility functions) that place value on various system states, which allows one to assess potential impact on mission outcomes should hazards be realized or averted. Hence, the framework also allows modeling of risk mitigation strategies to be employed directly into the decision-making process. Our approach combines two types of models, one focused on diagnostics and another on prognostics. We use BNs for diagnostics, allowing us to estimate (with uncertainty) the current health state of the system. Once health state is determined, we use this as "virtual evidence" in a companion CTBN model to reason through time.

## Bayesian Networks

Here, we provide a brief introduction to BNs. A BN is a graph-based representation of a joint probability distribution. Given a set of random variables $X = \{X_1, \ldots, X_n\}$, the BN provides a compact representation of joint distribution $P(X) = P(X_1, \ldots, X_n)$ by applying the product rule of probabilities and properties of conditional independence among the variables. A BN can be regarded as a "factored" representation of the joint distribution corresponding to:

$$P(X_1, \ldots, X_n) = \prod_{X_i \in X} P(X_i \mid Pa(X_i)) \tag{1}$$

Representing conditional probabilities $P(X_i | X_j)$ in a directed acyclic graph, the vertex for $X_j$ is connected by an outward directed edge to the vertex for $X_i$, in which case we say $X_j$ is a parent of $X_i$ (i.e., $X_j \in Pa(X_i)$). The graph structure, combined with a parameterization of the local distributions for each random variable $X_i$, corresponds to the specification of a BN.

## Continuous-Time Bayesian Networks

For predictive modeling, we use CTBNs. At the heart of a CTBN is a Continuous-Time Markov Process (CTMP). A CTMP is a model over continuous-time random process $\mathcal{X}$, consisting of two parts: an initial distribution $P_{\mathcal{X}}(0)$ and a transition intensity matrix $Q_{\mathcal{X}}$ defined over the states of $\mathcal{X}$. The entries $q_{i,j}$ in $Q_{\mathcal{X}}$ govern the rate of transition from state $x_i$ to state $x_j$ as a function of time. The $i$th diagonal entry, denoted $q_i$, is constrained to be the negative sum of the rest of the row (i.e., $q_i = -\sum_{j \neq i} q_{i,j}$). The distribution indicating if the process remains in state $i$ is exponential with rate $q_i$:

$$f_{q_i} = -q_i \exp(q_i t). \tag{2}$$

Conditional on a transition out of state $i$ occurring at time $t$, $X$ transitions from state $x_i$ to state $x_j$ according to a multinomial distribution with probabilities:

$$P(x_j | x_i, t) = -\frac{q_j}{q_i}. \tag{3}$$

CTBNs provide a factored representation of CTMPs. Let $X = \{X_1, \ldots, X_n\}$ be a set of discrete random variables. The model consists of two parts: a graph structure $\mathcal{G}$ and a set of parameters $P$. Graph $\mathcal{G}$ is a directed, possibly cyclic graph with nodes corresponding to variables $X$. Parameterization $P$ correspond to intensity matrices of conditional Markov processes, one for each $X_i \in X$, conditioned on its parents in graph $\mathcal{G}$. These intensity matrices are referred to as "conditional intensity matrices" (CIMs). We use a CTBN to capture the failure and hazard dynamics of the system under test.

## Diagnostic Bayesian Networks

We now present our formulation for the diagnostic BN. Recall, a CTMP (and thereby a CTBN) requires a prior distribution to kick start the process. We use a diagnostic BN as the basis for that prior distribution. Furthermore, we use a D-matrix [11] to provide the structure of that BN.

### *D-Matrices*

A variety of diagnostic models are possible for establishing health state. These include fault trees [10], first principle models [12], expert systems [13], and BNs [14]. Because it integrates with our framework, we use a BN derived from a diagnostic dependency matrix (i.e., D-matrix) [8].

A D-matrix is a binary matrix $D$ mapping faults to tests. Let $F = \{F_1, \ldots, F_d\}$ be a set of faults or diagnostic conclusions to be drawn in a system. Assume each $F_i$ is Boolean. Let $T = \{T_1, \ldots, T_n\}$ be a set of tests designed to detect presence of faults. Assume each test is also Boolean. Finally, let $D$ be the $d \times n$ binary matrix where:

$$D_{i,j} = \begin{cases} 1 & F_i \text{ is detected by } T_j \\ 0 & \text{Otherwise} \end{cases}. \tag{4}$$

A D-matrix can be represented as a BN, similar to the model described by Schwe *et al.* [11] which represents the dependency structure via noisy-Or nodes. Each $F_i$ and $T_j$ are defined as random variables (i.e., vertices) in the network, and conditional dependence relationships are defined where $D_{i,j} = 1$ indicates $F_i$ is a parent of $T_j$. Prior probabilities on each $F_i$ can be based on reliability data, and conditional probabilities $P(T_j | F_i)$ can be defined based on properties of the underlying test system [15].

### *Virtual Evidence*

One difficulty with probabilistic diagnostic systems is accounting for uncertainty in evidence collected. Two different formalisms exist to address evidence uncertainty in BNs: soft evidence and virtual evidence [16]. Soft evidence corresponds to replacing the conditional probability $P(T_j | F_i)$ at the time an observation is made (i.e., the test is performed) to capture the confidence in the test result. Inference is then applied using this distribution. More formally, if $P(T_j)$ reflects probability of a test result, we derive this by computing $P(T_j) = \sum_{T \setminus T_j} P(D)$ (i.e., we marginalize out the rest of the network). With soft evidence, we replace $P(T_j)$ with a revised estimate $P'(T_j)$ and update using Jeffrey's rule:

$$P'(D) = \sum_{T_j'} P(D | T_j') P(T_j') \tag{5}$$

Virtual evidence, on the other hand, inserts additional vertices into the model reflecting confidence of the evidence, $P(obs(T_j) | T_j)$. This is shown graphically in Fig. 1. In this case, we pre-set test confidences through the definition of observation distributions and apply the evidence to those vertices. Corresponding state of fault vertices is then inferred using the usual inference methods.

## Prognostic CTBNs

In previous sections, we spent time setting up tools for probabilistic fault diagnosis. This approach allows us to take observation uncertainty, dependency uncertainty, and failure uncertainty into account in a unified way. It also provides a way to specify prior distribution $P_{\mathcal{X}}(0)$ for the CTBN that we will be using for prognosis. We now discuss how prognostic CTBNs are constructed.

### *Fault Trees*

Within the automatic test systems community, many will have encountered the concept of a fault tree. The question we face is what kind of fault tree? In test program sets (TPS), a fault tree corresponds to the decision process of specifying a test, observing an outcome, and branching to the next step until a diagnosis or call out can be returned. Alternative forms of fault trees arise from Fault Tree Analysis (FTA) [17].

A fault tree arising from FTA corresponds to a directed acyclic graph where edge directions all proceed upward, from leaf to root. Leaves of the tree correspond to faults in the system. Interior vertices of the graph correspond to failures, effects, or hazards resulting from a fault. Interior vertices are also represented using logic gates (e.g., AND, OR, or XOR) indicating whether the corresponding effect is expected to occur because of fault(s) at the leaves of the tree. An example fault tree taken from [1], [18] is shown in Fig. 2.

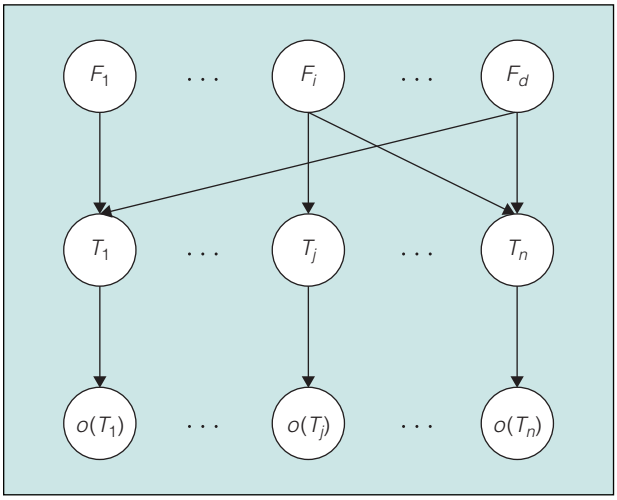


***Fig. 1.*** Diagnostic BN with virtual evidence, from [1], (© IEEE 2022, used with permission).

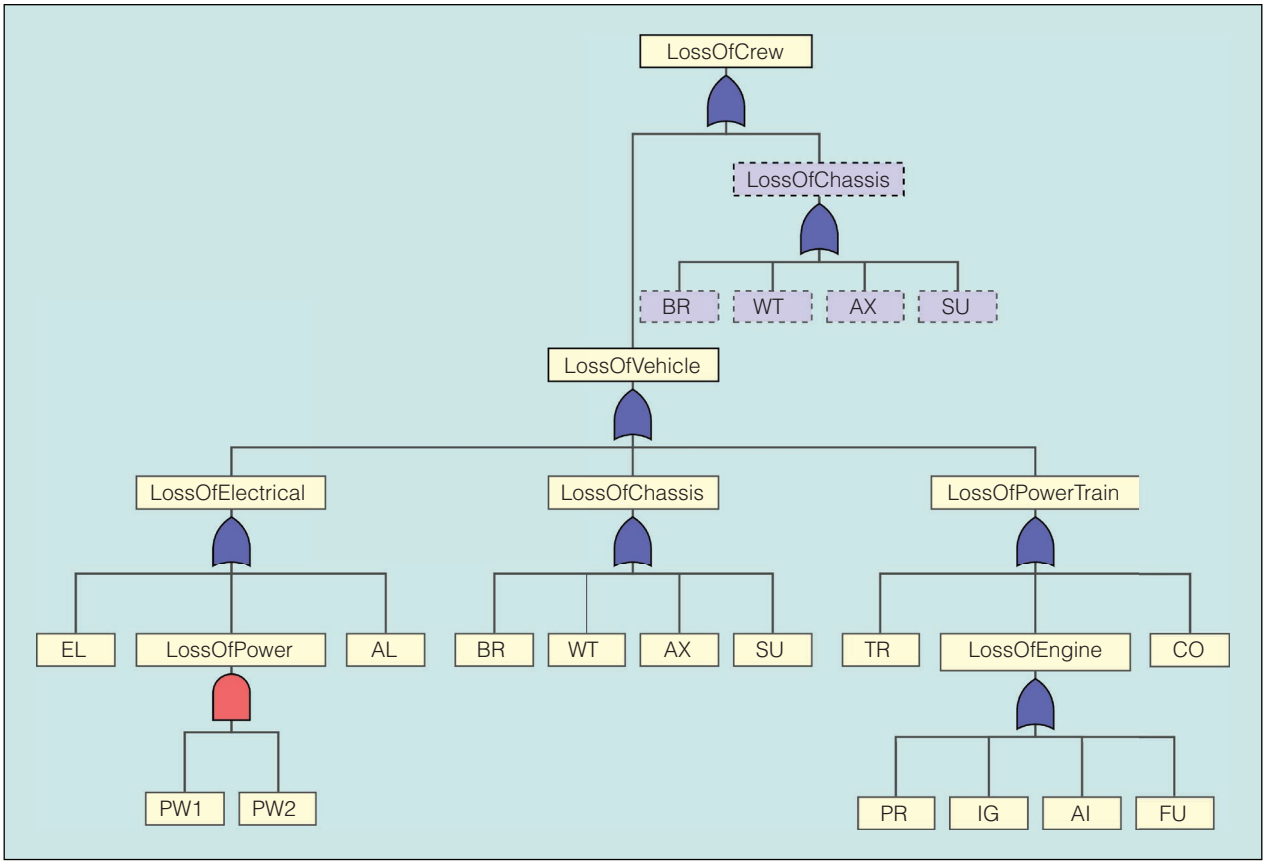


***Fig. 2.*** Sample Fault Tree, from [1], [18] (© IEEE 2022, used with permission).

Perrault *et al*. showed how to encode a fault tree as a CTBN [19]. To parameterize fault nodes, we use an intensity matrix for fault $F$ corresponding to:

$$Q_F = \begin{pmatrix} -\lambda_f & \lambda_f \\ \mu_f & -\mu_f \end{pmatrix} \tag{6}$$

where $\lambda_f$ is the failure rate of the fault and $\mu_f$ is the repair rate. If we assume interior nodes all have two children, each requires two CIMs. For the AND nodes, the intensity matrices correspond to:

$$Q_{X|Pa(X)} = \begin{pmatrix} -\lambda_X & \lambda_X \\ 0 & 0 \end{pmatrix}, \tag{7}$$

when $F_X\left(Pa\left(X\right)\right) = 1$ (all ones) and

$$Q_{X|Pa(X)} = \begin{pmatrix} 0 & 0 \\ \mu_{X|Pa(X)} & -\mu_{X|Pa(X)} \end{pmatrix}, \tag{8}$$

when $F_X\left(Pa\left(X\right)\right) = 0$ (not all ones). On the other hand, for OR nodes, the intensity matrices correspond to

$$Q_{X|Pa(X)} = \begin{pmatrix} 0 & 0 \\ \mu_{X|Pa(X)} & -\mu_{X|Pa(X)} \end{pmatrix}, \tag{9}$$

when $F_X\left(Pa\left(X\right)\right) = 0$ (all zeroes) and

$$Q_{X|Pa(X)} = \begin{pmatrix} -\lambda_{X|Pa(X)} & -\lambda_{X|Pa(X)} \\ 0 & 0 \end{pmatrix}, \tag{10}$$

when $F_X\left(Pa\left(X\right)\right) = 1$ (not all zeroes).

### *Mitigation Strategies*

When employing a risk-based approach to PHM (rPHM), the intention is to be proactive in mitigating risks. This is captured by implementing condition-based maintenance strategies that perform system support *prior* to system failure, mitigating potential effects of a failure occurring. This has the advantage of also providing alternative means for evaluating effectiveness in terms of the relationship between support costs and mission success arising from the application of risk mitigation strategies. Within the context CTBNs, mitigation strategies can be added directly as model components.

To incorporate mitigation strategies, we use a CTBN that incorporates decision nodes. Perreault referred to the resulting model as a Continuous-Time Decision Network (CTDN) [18]. A CTDN has two additional types of vertices—decision vertices (supporting the implementation of a mitigation strategy) and utility vertices (tied to performance functions). A decision node in a CTDN is a node with no parents whose state is known at all times, defining a local trajectory over full trajectory $\sigma\left[X\right]$. The states in $\sigma\left[X\right]$ must conform to a (possibly empty) constraint set which defines the set of possible states that may be assigned over all time intervals $\left[t_s, t_e\right)$. Thus, a decision vertex is a CTBN vertex where the state is predefined over the given time interval, forcing a particular child CIM to be activated.

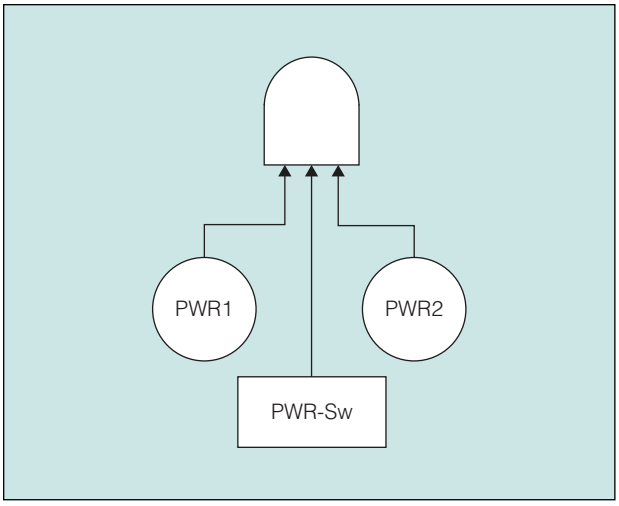


***Fig. 3.*** Simple mitigation of power loss, from [1], (© IEEE 2022, used with permission).

An example mitigation strategy for Fig. 2 is shown in Fig. 3. Here, two different power sources are available to power a vehicle. The "PWR-Sw" decision node switches between PWR1 and PWR2 based on the health of the two power sources by defining a CIM for the AND node conditioned on the state of the decision node and the power nodes.

### *Performance Functions*

In a CTDN, utility nodes are included via performance functions [20]. Utility nodes are used to compare the quality of provided mitigation strategies. Performance functions are represented by another vertex in the network; however, this vertex does not have a CIM associated with it. Rather, the vertex depends upon one or more CTMP vertices and defines a function based on trajectories defined over those CTMPs. Let $\sigma[Y]$ be a trajectory defined over a set of variables $Y \subset X$ and let $\langle t_s, t_e, Y_t \rangle$ be a set of observations over these CTMPs. The performance function for $Y$ can then be defined as:

$$f(\sigma) = \sum_{\langle t_s, t_e, Y_t \rangle} f_Y(t_s, t_e, Y_t). \tag{11}$$

This idea can be extended to include "factored" utility functions [21].

## The rPHM Process

Now that we have described the model, we outline the process for rPHM. For this discussion, we use the diagram in Fig. 4. Note that this process does not employ on-board health monitoring but depends on information collected from a test program set (TPS) on an automatic test system (ATS). The intent is to collect data for fault isolation *and* to establish health state for the unit under test (UUT). Based on health state, risk assessments can be made based on failure progression and mitigation/maintenance strategies assessed while the UUT is under maintenance.

At the start of the rPHM process is the UUT. At this point, the UUT has been pulled from the system and sent to be tested. The UUT is tested on an ATS, such as the US Navy's eCASS system, and faults are isolated. Once fault isolation is complete, the UUT is repaired and re-tested to determine if it can be returned to service. Following return to service testing, test results are captured, perhaps in standard form [22], and provided to a separate diagnostic engine based on a BN derived from a D-matrix. These test results are furnished as virtual evidence to the BN to provide a means to estimate and quantify uncertainty of the UUT health state.

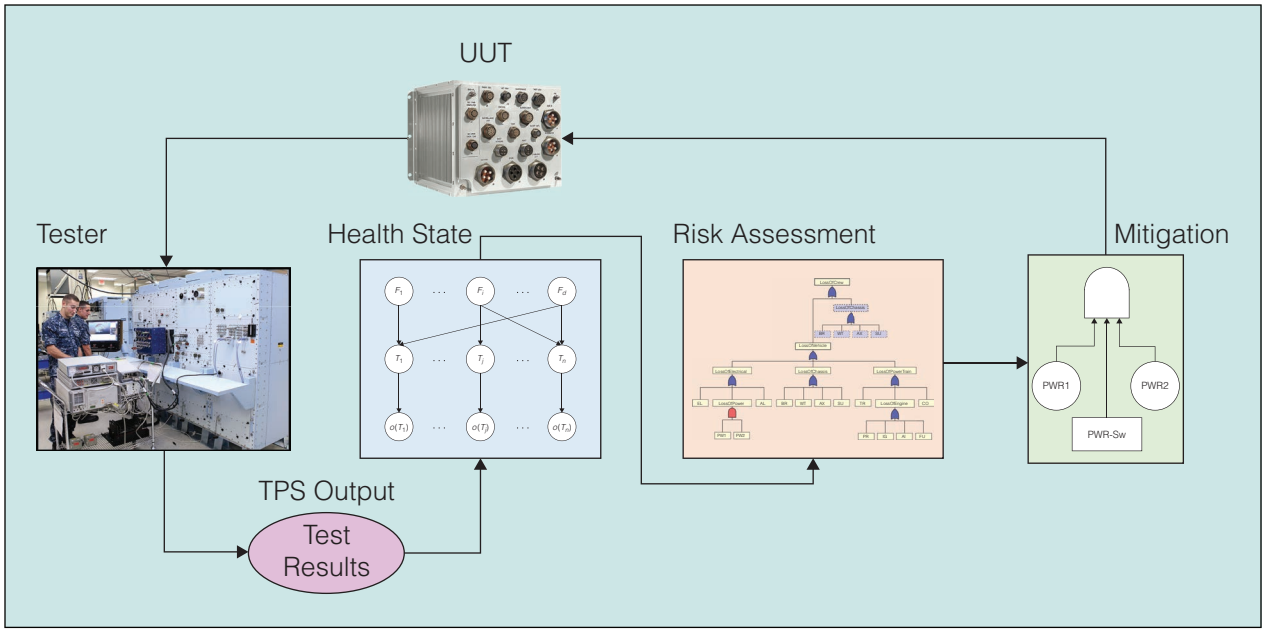


***Fig. 4.*** A probabilistic risk-based PHM process, from [1], (© IEEE 2022, used with permission).

Once health state is determined, resulting information can be provided to the CTDN that assesses potential hazards and mitigation strategies. A default mode where no mitigation is performed can be used for baseline performance using utility nodes in the CTDN. If utility is deemed too low, alternative mitigation strategies are tested to assess changes in utility. If determined that additional maintenance is warranted, information can be provided to technicians to take action, re-test, and re-assess health and failure progression.

## Benchmarking Sampling versus Exact Inference for CTBNs

A computational bottleneck in implementing these models for large systems is conducting inference in the CTBN models. Algorithms for efficient inference with BNs exist, so they are not considered in this part of the discussion. Conducting exact inference in CTBNs involves two steps. The first step "amalgamates" the network into a single CTMP, whose transition intensity matrix has row/column size equal to the total number of states in the system [5]. The second "query step" involves computing a matrix exponential for the amalgamated intensity matrix and a subsequent matrix-vector product [5]. Hence, computation time is largely driven by the total number of states in the system, where the size of the amalgamated matrix is exponential in the number of state variables. In fact, it has been proven that, in general, inference in CTBNs is intractable [23].

Given the general intractability of exact inference, approximate methods are required. In approximate sample-based inference, one generates trajectories through repeated sampling from exponential and multinomial distributions where parameters come from the defined CTBN [5]. Querying the network at a given time is done by computing the proportion of trajectories in a particular state at that time. When querying all states of the network at a given time, computational complexity is again dominated by the number of system states. However, one advantage of sample-based inference is that generation of samples and computing proportions over the samples are perfectly parallelizable. Hence, one can leverage hardware for conducting approximate inference on larger systems.

However, knowing how many samples one should take is a difficult task. To understand this problem better, we empirically investigate how approximate inferences are affected by sample sizes and properties of the network. We use average KL-divergence integrated across time to measure goodness of our approximation (denoted as IAKL). For discrete probability distributions $P$ and $Q$, KL-divergence of $Q$ from $P$ is defined as:

$$D_{KL}(P|Q) = \sum_i P(i)\log\left(\frac{P(i)}{Q(i)}\right). \tag{12}$$

We compute average KL-divergence over the network at time $t$ as:

$$\frac{1}{|X|}\sum_{x\in X} D_{KL}\left(P(x(t))|P'(x(t))\right) \tag{13}$$

where $P(x(t))$ is the true probability of state $x$ at time $t$ and $P'(x(t))$ is our approximation. Since this is defined for each $t$, we obtain average KL-divergence curves over our domain, and then reduce these curves via integration (i.e., IAKL).

In our experiments, we consider network structure as a parameter. The structures we consider are chain, ring, star, random binary-tree, random directed, and random directed-acyclic networks with given network size. See Fig. 5 for an illustration of these networks. They are networks (or subnetworks) that can each occur in the PHM process. For example, the chain network can represent a Go/NoGo chain, the ring network can represent a test/retest OK scenario, the star network represents a "many-causes" model, and the binary tree model can represent our fault tree example provided previously.

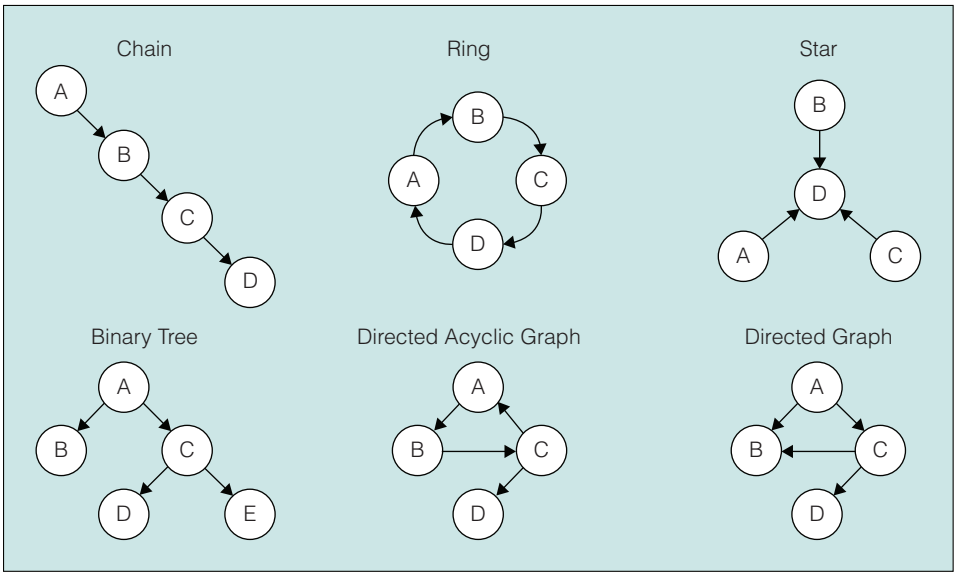


***Fig. 5.*** Network types considered for experiments.

### *Network Size, Network Type & Network Parameters*

In our experiments, all nodes have two states and priors over those states are specified as $P(0) = \{0.99, 0.01\}$. In our first experiment, CIMs are parameterized with $\lambda_f = 1.0$ and $\mu_f = 10.0$ (see (6)) and we vary the size of the network from 2 to 8, network type (Fig. 5), and sample sizes $n = \{1000, 2000, \ldots, 30000\}$. We conducted 10 independent replicates of each experiment. In our second experiment, we vary parameters of our CIMs with

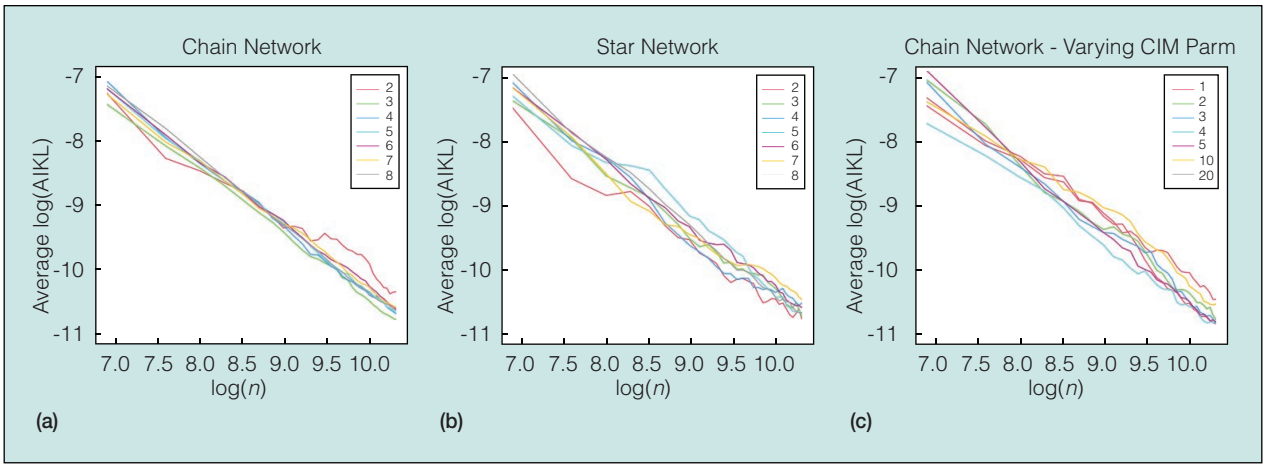


***Fig. 6.*** Average log(IAKL) plotted against log(*n*) for (a) Chain and (b) Star networks in first experiment and for (c) Chain with varying CIM parameters in the second experiment.

$\mu_f \in \{1, 2, 3, 4, 5, 10, 20\}$, holding all other parameters at previously specified values for the chain network.

With this experimental setup, we plot the log(IAKL) averaged across the 10 replicates against log(*n*) for each network size and two representative network types (Fig. 6a and Fig. 6b) and for varying CIM parameters (Fig. 6c). Notice average log(IAKL) and log(*n*) follows a linear relationship with a slope of approximately –1. Hence, a factor increase in sample size leads to a factor decrease in average approximation error of predictions. Surprisingly, we see no effect from network size and little effect from network type and varying CIM parameters (only differences in variability). This would be good news and a further argument to use sampling-based methods over exact methods for larger networks. However, we caution against extrapolating these results to larger networks or parameters outside the range investigated in these experiments since relationships that appear linear at one scale may appear quadratic (or worse) at another.

We also fit repeated measures regression models for each of these two experiments for log(IAKL) as a function of log(*n*) with varying intercepts and provide an ANOVA table in Table 1, estimated coefficients in Table 2 and estimated random effects in Table 3. Each of the variables (log of the number of samples, network type, and node size) have "significant" p-values (at an $\alpha = 0.05$ significance threshold). However, investigating the coefficient table, we see that this is largely being driven by large sample sizes, and the effects from network type and node size are not large enough to be of practical significance. We also see the relationship with the log of the number of samples is almost exactly –1. In our second experiment, we saw similar results to this. That is, we observed no practical significance associated with varying CIM parameters and a slope of –1 associated with the log of the sample size. We omit these tables for the sake of brevity.

Overall, these experiments suggest that for small networks (up to 1024 states) and steady-state probabilities not too close to the boundary (between 0.1 and 0.99), average approximation error decreases by a factor for each factor increase in sample size. Thus, average accuracy on the order of $10^{-7}$ would require a sample size somewhere on the order of $10^7$ to $10^8$. Note, it may still be appropriate to use exact inference when network size is small and desired accuracy is high.

**Table 1 – ANOVA results from first experiment**

| | DF | SSE | MSE | F-value |
|---|---|---|---|---|
| **log(*n*)** | 1 | 8585.1 | 8585.1 | 65693.60 |
| **networktype** | 5 | 72.6 | 14.5 | 111.18 |
| **nodesize** | 1 | 60.5 | 60.5 | 463.10 |

**Table 2 – Estimated coefficients from first experiment**

| | Coefficients | Std. Error |
|---|---|---|
| **(Intercept)** | –0.621869 | 0.039405 |
| **log(*n*)** | –0.987208 | 0.003852 |
| **networktypeRBT** | –0.071483 | 0.011156 |
| **networktypeRDAG** | 0.107251 | 0.011156 |
| **networktypeRDG** | –0.0788877 | 0.011156 |
| **networktypeRing** | –0.033228 | 0.011156 |
| **networktypeStar** | –0.134366 | 0.011156 |
| **nodesize** | 0.034652 | 0.001610 |

**Table 3 – Estimated random effects from first experiment**

| Groups | Name | Variance | Std. Dev. |
|---|---|---|---|
| **rep** | **(Intercept)** | 0.0735 | 0.2711 |
| **Residual** | | 0.0594 | 0.2437 |
| Number of observations: 12600, groups: rep, 420 | | | |

## Summary

We have described an approach to risk-based PHM which combines BNs for diagnostics and CTBNs for prognostics.

This approach establishes a new way of looking at the larger PHM problem where the focus is on managing risks associated with emerging faults in a system. By employing this perspective, the risk-based PHM approach also offers an alternative method for assessing PHM performance by focusing on probability of successful operation rather than life-cycle cost from associated repair efforts. As a means of implementing this approach, the output of a diagnostic model serves as the prior distribution for a CTBN that models hazard progression in a system. In taking this approach, we have utilized two previously developed tools, namely SAPPHIRE (for diagnostics BNs), developed with support of the US Navy, and CHARM (for prognostics CTBNs), developed with support from NASA. We have illustrated computational benefits of using approximate inference with the prognostic CTBNs as a way of mitigating the computational complexity that results from such an approach. As part of an ongoing effort with support from the US Navy, the two systems are being combined into a single system for prognostics and diagnostics using the modeling approach described in this work.

## Acknowledgments

This work is supported under a Small Business Technology Transfer contract in collaboration with GSS LLC. We would like to thank several people affiliated with the US Navy and F-35 JPO including Michael Malesich, Jennifer Fernandi, and Anthony Conway. We would also like to thank the members of NISL at Montana State University for their support, encouragement, and feedback.

## References

[1] J. Schupbach, E. Pryor, K. Webster, and J. Sheppard, "Combining dynamic Bayesian networks and continuous time Bayesian networks for diagnostic and prognostic modeling," in *Proc. IEEE AUTOTESTCON*, 2022.

[2] L. Sturlaugson, N. Fortier, P. Donnelly, and J. W. Sheppard, "Implementing AI-ESTATE with prognostic extensions in java," in *Proc. IEEE AUTOTESTCON*, 2013.

[3] *IEEE Standard for Artificial Intelligence Exchange and Service Tie to All Test Environments (AI-ESTATE)*, IEEE Std 1232-2010.

[4] T. Forrester, M. Harris, J. Senecal, and J. Sheppard, "Continuous time bayesian networks in prognosis and health management of centrifugal pumps," in *Proc. Ann. Conf. Prognostics and Health Management Soc.*, 2019.

[5] U. Nodelman, "Continuous time Bayesian networks," Ph.D. dissertation, Department of Computer Science, Stanford University, Palo Alto, California, USA, 2007.

[6] N. M. Vichare and M. G. Pecht, "Prognostics and health management of electronics," *IEEE Trans. Components and Packaging Technol.*, vol. 29, no. 3, pp. 222–229, Mar. 2006.

[7] P. W. Kalgren, C. S. Byington, and M. J. Roemer, "Defining PHM, a lexical evolution of maintenance and logistics," in *Proc. IEEE AUTOTESTCON*, pp. 353–358, 2006.

[8] R. Li, W. J. Verhagen, and R. Curran, "A systematic methodology for prognostic and health management system architecture definition," *Reliability Engineering System Safety*, vol. 193, p. 106598, 2020.

[9] *IEEE Standard Framework for Prognostics and Health Management of Electronic Systems*, IEEE Std 1856-2017.

[10] W. R. Simpson and J. W. Sheppard, *System Test and Diagnosis.* Boston, Massachusetts, USA: Springer, 1994.

[11] J. W. Sheppard and S. G. W. Butcher, "A formal analysis of fault diagnosis with d-matrices," *J. Electronic Testing: Theory and Applications*, vol. 23, no. 4, pp. 309–322, 2007.

[12] R. Reiter, "A theory of diagnosis from first principles," *Artificial Intelligence*, vol. 32, pp. 57–95, 1987.

[13] E. Shortliffe, "Mycin: A knowledge-based computer program applied to infectious diseases," in *Proc. Ann. Symp. on Computer Applications in Medical Care*, pp. 66–69, Oct. 1977.

[14] M. A. Shwe, B. Middleton, D. E. Heckerman *et al.*, "Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base: I. the probabilistic model and inference algorithms," *Methods of Information in Med.*, vol. 30, pp. 241–255, 1991.

[15] J. W. Sheppard and M. A. Kaufman, "A Bayesian approach to diagnosis and prognosis using built-in test," *IEEE Trans. Instrum. Meas.*, vol. 54, no. 3, pp. 1003–1018, 2005.

[16] J. Bilmes, "On virtual evidence and soft evidence in Bayesian networks," University of Washington Electrical Engineering Technical Report UWEETR-2004-0016, May 2004.

[17] L. Xing and S. V. Amari, "Fault tree analysis," in *Handbook of Performability Engineering*, K. B. Misra, Ed. London, UK: Springer London, pp. 595–620, 2008.

[18] L. Perreault, "On the usability of continuous time Bayesian networks: improving scalability and expressiveness," Ph.D. dissertation, Gianforte School of Computing, Montana State University, Bozeman, Montana, USA, 2017.

[19] L. Perreault, M. Thornton, and J. W. Sheppard, "Deriving prognostic continuous time Bayesian networks from fault trees," in *Proc. Ann. Conf. Prognostics and Health Management Soc.*, 2016.

[20] L. Sturlaugson, "Extensions to modeling and inference in continuous time bayesian networks," Ph.D. dissertation, Department of Computer Science, Montana State University, Bozeman, Montana, USA, 2014.

[21] L. Sturlaugson and J. Sheppard, "Factored performance functions with structural representation in continuous time Bayesian networks," in *Proc. Int. Florida Artificial Intelligent Research Soc. Conf.*, pp. 512–517, 2014.

[22] *IEEE Standard for Software Interface for Maintenance Information Collection and Analysis (SIMICA): Exchanging Test Results and Session Information via the eXtensible Markup Language (XML)*, IEEE Std1636.1-2018.

[23] L. Sturlaugson and J. W, Sheppard, "Inference complexity in continuous time Bayesian networks," in *Proc. Uncertainty in Artificial Intelligence*, 2014.

***Jordan Schupbach*** (jordan.schupbach@montana.edu) is a Ph.D. student in statistics at Montana State University, Bozeman, Montana, USA, where he earned B.S. degrees in mathematics and economics and an M.S. degree in statistics.

His current areas of research include topological data analysis, functional data analysis, and point processes.

***Elliott Pryor*** (elliott.pryor@virginia.edu) is a Ph.D. student in systems engineering at the University of Virginia, Charlottesville, Virginia, USA. He holds B.S. degrees in computer science and mathematics from Montana State University, Bozeman, Montana, USA. His current areas of research include machine learning, reinforcement learning, and optimization-based control, specifically for type 1 diabetes.

***Kyle Webster*** (kwebster@hopliteindustries.com) is a Software Engineer at Hoplite Industries, Bozeman, Montana, USA. He holds B.S. and M.S. degrees in computer science from Montana State University, Bozeman, Montana, USA. His research interests include machine learning, time series analysis, and data compression.

***John Sheppard*** (john.sheppard@montana.edu) is a Norm Asbjornson College of Engineering Distinguished Professor of Computer Science at Montana State University, Bozeman, Montana, USA. He holds a B.S. degree in computer science from Southern Methodist University, Dallas, Texas and M.S. and Ph.D. degrees in computer science from The Johns Hopkins University, Baltimore, Maryland. His research interests include system-level prognostics and health management, machine learning, population-based algorithms, and ethical and explainable artificial intelligence. In 2007, he was named a Fellow of the IEEE "for contributions to system-level diagnostics and prognostics."