Quality Engineering, 25:370–384, 2013

ISSN: 0898-2112 print/1532-4222 online
DOI: 10.1080/08982112.2013.819436

Taylor & Francis
Taylor & Francis Group

Condition-Based Maintenance via Simulation and a Targeted Bayesian Network Metamodel

Aviv Gruber,
Shai Yanovski,
Irad Ben-Gal

Department of Industrial Engineering, Tel Aviv University, Ramat Aviv, Israel

ABSTRACT Condition-based maintenance (CBM) is increasingly applied to operational systems to reduce lifecycle costs. Predicting the performance of various CBM policies is a challenging task addressed in this work. We suggest a CBM framework that is based on system simulations and a targeted Bayesian network model. Simulations explore the robustness of various CBM policies under different scenarios. The Bayesian network, which is learned from the simulation data, is then used as an explanatory compact meta-model for failure prediction. The framework is demonstrated through a study of an operator of a freight rail fleet. This study demonstrates a significant profit improvement compared to other methods.

KEYWORDS Bayesian networks, condition-based maintenance, meta-model, simulation

INTRODUCTION

Man-made systems are prone to deterioration over time and therefore require ongoing maintenance to avoid malfunctioning. Accordingly, it is essential to undertake an effective preventive maintenance (PM) policy that minimizes the life cycle cost (LCC) of the system and maximizes its operational profit.

A relatively simple PM approach is to use a time-based maintenance (TBM) policy, which implies, for example, an a priori scheduling of various PM tasks (based on elapsed time or system cycles). Over the years several extensions of the traditional TBM were suggested, such as the reliability-centered maintenance approach that studies and analyzes the functionalities of different operational tasks and their effects on the system reliability (Moubray 1991; Nowlan and Heap 1978). Nevertheless, despite these developments, the TBM is known to be suboptimal, because it does not account for the operational condition of the systems in real time (Dubi 2000). As an alternative to the prescheduled maintenance approach, the condition-based maintenance (CBM) schedules the maintenance tasks according to system's conditions and (partially) observed system states (Jardine et al. 2006). This article focuses on such a predictive approach; namely, addressing questions such as when a maintenance task should be performed, to which system, and under which observed conditions.

Address correspondence to Irad Ben-Gal, Department of Industrial Engineering, The Iby and Alader Fleischman Faculty of Engineering, Tel Aviv University, Ramat Aviv 69978, Israel. E-mail: bengal@eng.tau.ac.il

In small systems, an optimal PM policy can be obtained by conducting an exhaustive search over all feasible maintenance policies. In larger and more complex systems, simulation-based approaches are often used to obtain some estimation for satisfactory PM policies. Nevertheless, even with powerful simulators, the exhaustive search is typically too time-consuming for evaluating each potential PM setting (Dubi 2000). Moreover, even if a powerful simulator exists, it is not clear how to efficiently use it to produce a good predictive maintenance policy, as we shall propose in this article.

Condition-based maintenance methods attempt to define a PM policy according to the state of the system at various time periods. Previous works showed that the CBM approach can improve the PM plan considerably (Jardine et al. 2006; Peng et al. 2010). Nevertheless, the implementation of CBM methods remains a challenging task, because it requires generating reliable prognosis models that analyze and predict the system operational availability under various system conditions (Sheppard and Kaufman 2005). CBM models are often hard to formulate as closed-form analytical models without relying on physical models that are essentially feasible in simple mechanical systems. As an alternative to the exact modeling, it has been suggested to implement CBM in an automated fashion by using metamodels backed up by expert knowledge (Jardine et al. 2006). To address this challenge, model-free approaches were proposed for CBM optimization. For example, Marseguerra et al. (2002) optimized a CBM plan by means of genetic algorithms and simulation. Other surrogate models that could be used for PM were suggested over the years (see, for example, Forrester and Keane 2009; Keane and Nair 2005; Queipo et al. 2005).

In this article, we follow these approaches and suggest a CBM-based policy that combines both a simulation model of the system and a predictive metamodel. The simulation model of the system is based on expert knowledge and historical data. Such a model can be generated by conventional simulation software, such as Arena or MATLAB and can be further enriched by data that is gathered from an operating system, if such data are available. The simulator can be used to test various system settings, as well as to introduce variability and noise into the modeled system for evaluating the robustness of various CBM plans. The main required inputs into the simulation model are (1) the state-transition distributions of individual system components, such as the components' failure-restoration distributions; (2) an interaction scheme for these components that can be expressed, for example, by a reliability block diagram or another system architecture schema; (3) an initial PM policy by which the system is serviced and maintained that will be improved by the proposed approach; and (4) cost parameters that are related to the PM policy, for example, costs of unscheduled failures, costs of PM operations, and costs of system downtime.

The simulation outputs are represented by system operational measures, such as the system reliability, availability, and maintainability, from which associated costs under various environmental conditions can be calculated. These measures, as well as the system settings, are then used as an input to the targeted Bayesian network (BN) model proposed by Gruber and Ben-Gal (2011). The BN is learned from the simulation scenarios, using the default parameters that are described in Gruber and Ben-Gal (2011). It represents, in a compact and descriptive manner, the effect of various system conditions and settings on the potential failures of components. The BN is then used as a metamodel for predicting system failures according to different system attributes and PM setting. These attributes are associated not only with the state of the physical system but can also be related to environmental, operational, or other factors that can affect the system availability (see, e.g., Prescott and Draper 2009).

Note that system failures can be viewed as a stochastic process that cannot be anticipated precisely.

Nevertheless, often such a process can be characterized by failure distributions of its individual components and by their interactions. Such information, together with a logistic support plan that includes the PM policy, spare parts considerations, and environmental conditions that affect the system availability can be captured by the BN model. Once the BN model is constructed and its parameters are estimated, it is used as a prognosis metamodel for failure prediction of various system components. According to these predictions, different PM scenarios can be evaluated (e.g., triggering maintenance tasks to components with a failure probability higher than a certain threshold) and then an overall system performance can be evaluated to select the best PM plan. Using such an approach, the proposed CBM strategy does not require any underlying closed-form formula. Instead, it uses the targeted BN model, which is designed to predict a system failure by automatically selecting the most influencing features for such a prediction. Using a BN as a metamodel is also appealing because it is a general model that can cover a wide range of systems. Systems that are suitable for the proposed framework are those in which the exact relations between components' failure to the overall system maintenance requirement are undefined and unclear. Some examples for such systems include (but are not limited to) rail, aircraft, wind turbines, vessels, vehicles, nuclear reactors, and aerospace systems.

The rest of this article is organized as follows. The next section overviews the challenges of PM optimization in reliability–availability–maintainability (RAM) models and refers to several methods for addressing these challenges. The section also describes the targeted BN learning and the motivation for using a BN as a metamodel for prediction. The following section provides a schematic framework for the implementation of the proposed CBM approach. This section demonstrates the implementation of the proposed approach on a freight rail fleet based on a real case study of a European operator. The penultimate section analyzes some key features of the proposed approach. Finally, the last section concludes the article with a short discussion of potential future directions.

BACKGROUND

This section is divided into three subsections. The first subsection overviews key principles of maintenance plans and their implications on ageing systems. The second subsection explains what a Bayesian network is and emphasizes the properties of the targeted BN approach, which is implemented in the proposed framework. The third subsection briefly describes the advantages of using Monte Carlo simulation (MCS), particularly for modeling aging systems.

Maintenance Schemes

Maintenance is defined as a set of all activities and resources needed to uphold an element's specific performance and condition in a given time period. The precise definition of the term evolved with time from the simplistic "repair broken items" (Tsang 1995 pp. 3–17) to a more complete definition: "Combination of all technical, administrative, and managerial actions during the life cycle of an item intended to retain it in, or restore it to, a state in which it can perform the required function" (Bengtsson 2004, p. 15).

When examining the phrase "retain it in, or restore" in the definition, it becomes clear that besides the activities that are focused on repairing broken items after breakdowns (restore), there is an additional approach of performing upkeep activities before the next breakdowns happen in order to prevent them (retain).

Bengtsson (2004) defined *corrective maintenance* and *preventive maintenance* as follows:

- Corrective Maintenance (CM): "Maintenance carried out after fault recognition and intended to put an item into a state in which it can perform a required function" (pp. 16–17). CM can be divided into two cases. First, when the breakdown is critical and affecting the functionality of the whole system it must be repaired immediately. In many situations it is characterized by significant costs caused by the breakdowns. Second, as long as the breakdown is not affecting the comprehensive function, the repair can be delayed. In these situations, sometimes it is possible to defer the repair process to a more appropriate time, taking into account the production capacity. The first case is referred to as *immediate CM*, whereas the second case is referred to as *deferred CM* (see Figure 1).
- Preventive Maintenance (PM): "Maintenance carried out at predetermined intervals or according to prescribed criteria and intended to reduce the probability of failure or the degradation of the functioning of an item" (p. 17). PM can be obtained in two ways. The first is known as *time-based maintenance*, which is described as "preventive maintenance carried out in accordance with established intervals of time or number of units of use but without previous condition investigation" (p. 17).
- The second is known as *condition-based maintenance*, which is described as "preventive

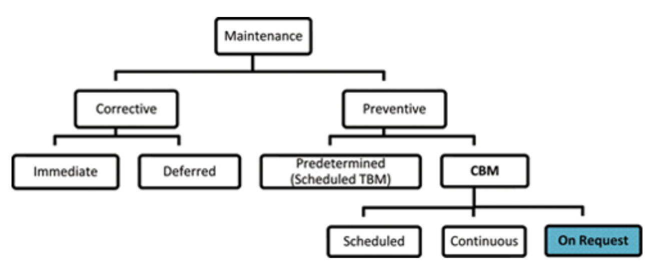

FIGURE 1 Different maintenance strategies as described by Bengtsson (2004). (Color figure available online.)

maintenance based on performance and/or parameter monitoring and the subsequent actions" (pp. 17–19). Note that the difference between performance and parameter in this context is that the performance is often regarded as a target variable, whereas a parameter is considered to be any explanatory monitored variable. The described process of the CBM parameter monitoring can be either continuous, scheduled, or on request. When properly implemented, this approach enables one to combine the benefits of preventive maintenance while suggesting a way to reduce the costs of unnecessary or excessive scheduled maintenance operations and still allowing one to keep the maintained equipment in a healthy operational condition (Jardine et al. 2006). For example, a general formulation of the LCC within a lifetime T is given below:

$$\begin{aligned} \mathrm{LCC} = {} & N_f \times C_f + N_{pm} \times C_{pm} \\ & + C_{T_d} \int_0^T (1 - A(t))\, dt, \end{aligned} \tag{1}$$

where C_f represents the cost of an unscheduled failure, C_{pm} stands for the cost of a preventive maintenance action, C_{Td} is the downtime cost rate, N_f is the number of unscheduled failures up to time T, N_{pm} is the number of PM actions undertaken within time T, and $A(t)$ represents the time-dependent availability, where the term *availability* stands for the probability that the system will be in an operation state.

Figure 2 depicts in a qualitative fashion the general behavior of the downtime of an aging system and of the life cycle cost as functions of a PM policy in a lifetime. The figure illustrates that though a closed-form of these quantities is often unobtainable, a global optimal PM policy lies between two extreme policies. One extreme policy is to perform PMs infrequently (following the "less maintenance" direction), resulting in an increasing failure rate, which could lead to unplanned downtimes and high cost of repair and restoration. The second extreme policy is to perform PMs very frequently (following the "more maintenance" direction), resulting in an unavailable system due to PM and excessive PM costs yet again leading to high costs. Clearly, some other policy exists that minimizes that cost, referred to as the *optimal policy*, in a nonrigorous definition. A general closed-form equation for obtaining the optimal

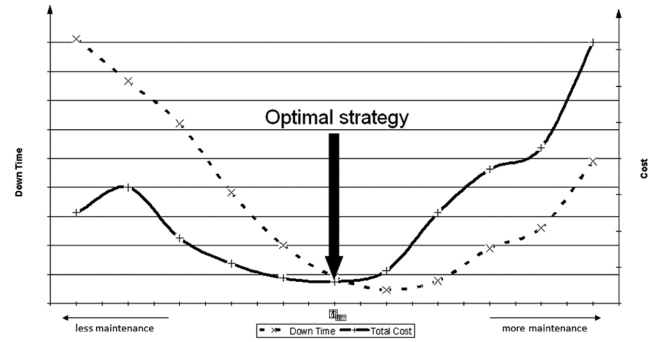

FIGURE 2 Qualitative illustration of the association between downtime and LCC vs. maintenance policy.

policy does not exist. The suggested framework, presented here, attempts to approximate this policy by detecting those cases where PM operation is truly required while, on the other hand, avoiding unnecessary PM operation. A general survey and comparison of various maintenance policies of deteriorating systems was provided by Wang (2002).

A CBM program contains three key phases: (1) data acquisition: collection of the raw data that are considered relevant to system health; (2) data processing: analysis of the collected raw data to allow better interpretation; and (3) maintenance decision making: recommendation of the efficient maintenance activity based on previous phases (see Figure 3).

It is worth noting that CBM practices occasionally distinguish between *diagnostics* and *prognostics*. The first term refers to detection, isolation, and identification of the failure when it occurs. The second refers to the prediction of failures based on some likelihood estimation before they occur.

The literature on the various aspects of the CBM methodology is extensive and treats a variety of systems, components parts, and approaches. A review of different aspects of CBM implementation and methodology was provided by Jardine et al. (2006) and Peng et al. (2010).

This article belongs to a class of studies that can be classified as an artificial intelligence–based approach to CBM (Jardine et al. 2006). Among the artificial intelligence–related methods for prognosis models, (e.g., Vining et al. 1993), there is a body of works using

FIGURE 3 Three phases of CBM (Jardine et al. 2006).

various types of artificial neural networks, as seen, for example, in Paya et al. (1997), Samanta and Al-Balushi (2003), Spoerre (1997), and Y. Li et al. (1999). Another approach is implementing expert systems, where the effort is concentrated on using domain expert knowledge in a computer program allowing one to perform a computerized inference on an engine that is configured in advance for that task, such as demonstrated by Deuszkiewicz and Radkowski (2003) and Hansen et al. (1994). In addition, there are combinations of the above-mentioned approaches, such as Brotherton et al. (2000), Y. Li et al. (1999), and others, as extensively reviewed by Jardine et al. (2006) and Peng et al. (2010).

There are several related studies dealing with applications of Bayesian networks for maintenance and reliability. Sheppard and Kaufman (2005) proposed a Bayesian framework for diagnosis and prognosis, providing an illustration for the approach on a stability augmentation system of a helicopter. Liu and Li (2007) discussed a decision support system for maintenance management using BNs. Arnaiz et al. (2010) proposed a decision support system based on operational risk assessment in order to improve aircraft operability. All of these works, however, do not specifically use the targeted BN approach. The uniqueness of the proposed approach is in the use of a targeted BN as a surrogate model to support CBM prognosis and failure prediction.

The novelty of the proposed framework compared to the studies mentioned above is that the learned targeted BN model exploits merely the relevant information for predicting the failures as gathered from simulation data and uses the predictions by the CBM model within the simulation for assessing the actual effect on the LCC, as will be described in the next sections.

Targeted Bayesian Network Model

A BN is a probabilistic graphical model that encodes the joint probability distribution of some domain in a compact and explanatory fashion. A BN is a directed acyclic graph, $\mathcal{G}$, containing vertices (or nodes) and edges and a set of parameters representing conditional probability tables (CPTs), denoted by Θ, of discrete or discretized random variables (Pearl 1988).

A BN $B(\mathcal{G}, \Theta)$ can often be used to represent the joint probability distribution of a vector of random variables $\mathbf{X} = (X_1, \ldots, X_N)$. The structure $\mathcal{G}$ $(\mathbf{V},\mathbf{E})$ is a directed acyclic graph composed of $\mathbf{V}$, a vector of nodes representing the random vector $\mathbf{X}$, and $\mathbf{E}$, a set of directed edges connecting the nodes. An edge $E_{ji} = V_j \rightarrow V_i$ manifests conditional dependence between the variables X_j and X_i (given prior knowledge about X_i). E_{ji} connects the node V_j to the node V_i (Heckerman 1995) and thus V_j is also called the *parent* of V_i. We denote $\mathbf{Z}_i = \{X_i^1, \ldots, X_i^{L_i}\}$ as the set of "parent" variables of the random variable X_i represented by the set of parent nodes $\mathbf{V}_\mathrm{i} = \{V_i^1, \ldots, V_i^{L_i}\}$ in $\mathcal{G}(\mathbf{V},\mathbf{E})$, where for any literal, the superscript j stands for its index in the corresponding set and where $L_i = |\mathbf{Z}_i|$ is the size (cardinality) of the subset $\mathbf{Z}_i \subset \mathbf{X}$. The set of parameters Θ represents the local conditional probabilities, $p(x_i|\mathbf{z}_i)$, over $\mathbf{X}$ that is estimated from observed data or given a priori by an expert (see also Gruber and Ben-Gal 2011).

One of the advantages of BN models is that they serve as an intuitive tool, due to their qualitative graphical representation, while maintaining a rigorous, well-defined mathematical model that compactly and efficiently represents the domain (Ben-Gal 2007). A BN can be constructed manually, based on knowledge and hypotheses about the relationship among the domain's variables, or constructed automatically from the observed data, including the network-structure learning and the CPT estimation (Ben-Gal 2007). The latter practice has grown remarkably in recent years, especially in the light of information technology, where data availability is growing massively in many industrial domains. Because BN learning is an NP-hard problem (Chickering et al. 1995), most BN learning methods split the learning procedure into structure learning (edges and their directions) and parameter learning (CPTs), given the learned structure of the BN (Claeskens and Hjort 2003). Moreover, most of the methods learn the BN model in a general fashion, namely, encoding the joint probability distribution of the variables' set, irrespectively to the application that will be used; hence, such methods that learn a general BN (GBN) are often referred to as *canonical*.

Though GBN learning methods attempt to best approximate the joint probability distribution, they address the tradeoff between the model complexity versus the prediction error of the learned BN. Unlike the canonical approach of GBN learning methods, target-oriented methods learn the structure of the BN specifically for marginal purposes, such as classification (Gruber and Ben-Gal 2011). These methods aim to be more effective for marginal purposes, rather than approximating well the entire domain. The naïve Bayes model (see Duda and Hart 1973), for instance, is one of the simplest and most well-known target-based Bayesian classifiers, in as much as it does not require structure learning. Instead, the structure is fixed a priori, where the node representing the class variable is predetermined as the common parent of all nodes that represent the attribute variables. The naïve Bayes model is popular due to its simplicity, but it uses a crude conditional independence assumption.

In the suggested framework, we employ the targeted BN learning (TBNL) algorithm (Gruber and Ben-Gal 2011) as the metamodel to reflect the most influential failure causes, as well to produce an interdependence among them. The TBNL follows a targeted approach for accounting for the BN complexity by allowing for the final objective as a target variable while learning. The target variable in the current application is an indicator of system failures in the subsequent time interval (in the next month, for example). Thus, rather than learning the joint probability distribution as a whole, the TBNL algorithm aims at learning only the monitored variables out of all of the simulated ones, as detailed in the sequel, or that best predict possible system failures.

In general, the TBNL first attempts to obtain the most influential set of variables with respect to the target variable and then attempts to construct the connection among these variables. The influence of a variable or a set of variables on another variable is reflected by their dependencies, obtained by using mutual information measures that are well known in information theory (Gruber and Ben-Gal 2011).

One of the advantages of using the TBNL for the current endeavor is that it enables managing the BN complexity and controlling it versus its prediction accuracy. The TBNL refers to a common indicator of model complexity, defined as the number of free parameters that represent the network

$$k = \sum\nolimits_{i=1}^{N} (|X_i| - 1) \prod\nolimits_{j=1}^{L_i} |X_i^j|, \qquad [2]$$

where N denotes the total number of variables, X_i represents the i^{th} variable, and, accordingly, $|X_i|$ denotes the number of entries it can obtain. Denote the parents' set of X_i by $\mathbf{Z}_i$, then $X_i^j \in \mathbf{Z}_i$ represents the j^{th} parent of X_i. Similarly, $|X_i^j|$ represents the number of values that the j^{th} parent of X_i can have (recall that the number of parents is L_i).

Two input parameters of the TBNL that we shall use later on in the example demonstration are the MinPRIG and MaxPRIE. The MinPRIG stands for *minimum percentage relative information gain* (PRIG), and the MaxPRIE stands for *maximum percentage relative information exploitation* (PRIE). These parameters trigger the stopping conditions of the BN construction. The MinPRIG determines the minimal step for adding information, whereas the MaxPRIE determines the total information to be cumulated about each variable. A rather detailed discussion on the properties of the MinPRIG and the MaxPRIE parameters can be found in Gruber and Ben-Gal (2011). These input parameters of the TBNL enable selecting the most important and relevant information about the target variable while controlling the model complexity. This engineering approach property of the TBNL makes the resulting targeted BN an efficient surrogate model in the sense that it can be utilized particularly for predicting a system failure, given a specific condition of the system. The simulation model, on the other hand, provides a holistic and generalized assessment of the system life cycle.

Modeling and Simulation in RAM Problems

Monte Carlo simulation is an effective technique for modeling typical RAM problems, in that it is insensitive to their natural level of complexity (Wang and Pham 1997; Wang 2002). The drawback of using MCS (and simulation in general) while attempting to optimize the PM policy of an aging system is that the search space of possible policies is practically unmanageable, requiring a considerable number of computationally expensive simulations of system availability and failures under various scenarios (Dubi 2000; Gruber and Ben-Gal 2012). Barata et al. (2002) suggested a CBM optimization method based on a modeling and simulation approach and proposed an innovative way for addressing such a problem. However, their approach to the optimization requires the specific domain knowledge.

Proposed Framework

In this study we propose a framework architecture and methodology based on a combination of MCS and targeted BN-based decision engine for RAM optimization. The proposed methodology consists of four main modules, as shown in Figure 4. The framework can be applied to either a known operational system or to a new system in a design stage. The underlying assumption is that the state transition (such as operational-to-failed, failed-to-repair, and repaired-to-storage, which defines a state where the repaired part is delivered back to storage) distributions of the components are based on expert opinion. Typically, the expert opinion is derived both from the equipment manufacturer manual and can be backed up by observed data analyses.

1. System modeling and simulation: Modeling the system operational lifecycle using a simulator, which is based on expert knowledge that can be supported by data, and subsequently conducting the necessary validation tests in order to ensure that the simulation reliably represents the system in various operational scenarios.
2. Learning the targeted BN model from simulated data: Applying the TBNL algorithm to the simulated data to generate an initial targeted BN. The BN is used as a compact CBM prognosis model for failure prediction.
3. Refining the CBM model by tuning the targeted BN model using cross-validation: Performing iterative refinement of the targeted BN model to obtain a satisfactory failure prediction performance (not necessarily optimal) by tuning the parameters of the TBNL algorithm.
4. Searching for an effective maintenance policy based on the developed CBM model: Using the BN model to generate a class of maintenance policies that are triggered by different thresholds on the predicted failure probability of the system. Evaluating each of these PM policies by the simulator and selecting the best one with respect to a desired objective function (e.g., min LCC, max profit).

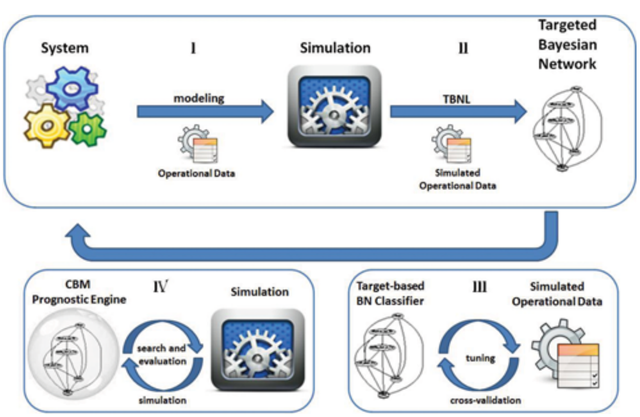

FIGURE 4 **Suggested architecture. (Color figure available online.)**

Applying these four modules enables the selection of a satisfactory CBM policy that can be adapted to the real system.

Implementation Demonstration

The example simulation model is based on a real case study of a European operator of a freight rail fleet. For the sake of the CBM discussion, we demonstrate a reduced model; thus, the demo does not account for all the original components, processes, and logistic considerations. The reason for the reduced model is not due to scalability issues but merely to help the reader gather the concept and main principles of the proposed approach. Essentially, the simulation model can scale up to thousands of components of different types or thousands of systems in the deployed fleet.

The simulation model was validated by benchmarking it against the original model that was modeled by the SPAR modeling platform (www.clockwork-solutions.com). The SPAR is an advanced model development environment that has been designed to evaluate the lifecycle management of a system or fleet of systems. It is a discrete event simulator that is based on a Monte Carlo engine that incorporate a robust solution capability. The Monte Carlo engine samples the time points of all the modeled events, allowing for the modeled dependencies. For example, if a component fails at a certain time point, the subsequent process (e.g., the restoration or removal) will start off by sampling its time point from the current time point. In addition, any modeled dependence rule will apply; for example, a failure of a cooling subsystem can increase the failure rate of another subsystem because the latter is designed to work at some temperature boundaries. The simulation is based on a free-flight kernel, which means that between two subsequent events, all of the quantities, states, and responses do not change; hence, the time step of the simulation is not fixed. The output resolution of the simulated case study is set to one day. SPAR simulates the system lifetime until the time reaches a predetermined service time. Because it is a Monte Carlo application, the entire simulation is repeated for as many iterations as the user has requested. In this case study, 10 iterations were conducted and every numerical measure per each wagon within the fleet was averaged over the iterations throughout the service time. This enables a holistic prediction of asset operation and support, where it supports the evaluation of dynamic operations characteristics and supply scenarios, equipment aging, condition-based maintenance, partial repair, and finite resource capacities. The following is a detailed description of the modeled case study.

The fleet comprises 104 wagons of two frame types in total (78 of type 1 and 26 of type 2). Each type is deployed in a different field and can carry different loads over a period of 20 years. Each component is critical to the system (wagon) functionality; hence, the system function is serial. Upon failure, the failed component is repaired and inserted back, a process known as *restoration*. During the restoration, the wagon is unavailable.

The state-transition distribution is determined by an expert opinion, based on preliminary assumptions and conventions. The common methodology is to use a curve fitting on each of the state-transition mechanisms exclusively. A common practice for a failure process of an aging line replaceable unit (LRU; e.g., mechanical assemblies) is to use a curve fitting of the Weibull distribution to obtain the scale and shape parameters of the distribution. A common practice for a nonaging process (e.g., foreign object damage or failure of electronic equipment) is to use a curve fitting of the exponential distribution to obtain the mean time between failures (MTBF). A common practice for a repair-restoration process of a failed LRU is to use either a curve fitting of the normal distribution or of the log-normal distribution, depending upon the logistic setup, to obtain the corresponding parameters (Balakrishnan and Varadan 1991; Kelton and Law 2000; Saranga and Knezevic 2000). Using such input distributions allows one to introduce another level of uncertainty (e.g., by adding a noise component or increasing the distribution variance) to obtain a robust PM solution that accounts for input settings that were not observed in the data. The use of a simulation enables one not only to introduce variability into the system but also to map it through the TBNL metamodel to a component failures output. All of the components in this example are considered to follow Weibull time-to-failure distributions and normal time-to-restoration distributions. The modeled components are listed in Table 1 along with their properties.

The above estimates are given by the operator as a coherent part of the model description. In addition, are liability block diagram and other properties of the system are provided, together with its operational environment, which are provided as a part of the model design. Corrosion is one of two failure modes attached to the headstock (the second, shown as headstock, is its wear and tear); therefore, corrosion is the only virtual process and is not being "restored," but this failure mode triggers and accelerates the wear and tear of the headstock. Thus, only upon a corrosion "failure" is the headstock failure process activated. After the headstock has completed a corrective maintenance operation, the corrosion is "repaired" ad hoc, the "failure" process associated

TABLE 1 List of the Modeled Components and their Corresponding Weibull Distribution Parameters of Failures

Component/ process	Failure distribution parameters (years)		Restoration distribution parameters (years)		Costs(krona)				
	Scale	Shape	Mean	StD	Removal	Ship	Repair	Inspect	Replace/build
Running gear	5.86	3.8	0.0014	0.0007	0	1,000	2,500	0	600
Wheelsets	11.05	3.8	0.3342	0.0002	0	670	1,500	0	400
Draw gear	9.96	3.8	0.0854	0.0014	0	400	2,000	0	600
Loading frame 1	9.00	3.8	0.0082	0.0001	0	500	5,000	0	1,200
Loading frame 2	9.05	1.9	0.5027	0.0001	0	500	5,000	0	1,200
Unloading	10.86	3.8	0.5027	0.0007	0	400	4,000	0	1,200
Headstock	9.05	1.9	0.5027	0.0001	0	0	0	0	0
Corrosion	10.14	1.9	n/a	n/a	n/a	n/a	n/a	n/a	n/a
PM frame 1	n/a	n/a	0.0274	0.0014	2,000	0	0	2,000	0
PM frame 2	n/a	n/a	0.0548	0.0027	2,000	0	0	2,000	0

with it is reactivated, and the headstock in the failure mode is set to passive. The restoration efficiency is not perfect; thus, a restored component is not as good as a new one. We use the term *probabilistic age* (or simply age hereinafter) as the value of the cumulative distribution function (CDF) of any stochastic process apart from an exponential distribution (Dubi 2000). After restoration, which is obtain by shifting the CDF such that the hazard function represents a younger component, the age of the repaired component is "reduced" by 60% relative to its age prior to the failure. Thus, after a component is repaired, its next failure time is shifted by a time interval that is equivalent to 60% of the failure CDF prior to the repair (Dubi 2000).

Apart from the physical components, the list includes some properties of the preventive maintenance operations for type 1 frame and for type 2 frame. The PM restoration designates its duration. The age reduction upon a PM operation is 100%, namely, the component is considered as good as new.

For each component or process that is listed above, a corresponding cost is shown and broken down into subtasks. The operator operates the fleet under a contract. The contract is summarized in Figure 5.

The penalty is not fixed because it grows linearly with the average unavailability, should the availability go below the satisfactory value. This mechanism is considered in order to avoid cases where the availability drops dramatically merely due to maintenance considerations. The contract works as follows: at the end of every month, the availability over the past month is calculated and the operator profits the amount shown next to the corresponding figure if the actual availability met that requirement.

As mentioned above, the example model was derived and reduced from a prototype model, characterizing a European operation. In order to enable integration with the BN models, the model was rebuilt in Visual SPAR modeling platform (courtesy of Clockwork Solutions Ltd.), which uses a .NET

Profit Factor	Availability
30000	0.91 (desired)
20000	0.83 (satisfactory)
$-30000(1-A)$	Otherwise (a penalty)

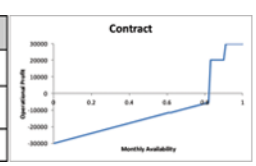

FIGURE 5 **Contract figure of the operator. (Color figure available online.)**

environment. This environment enables communicating with the MATLAB libraries and tools by which the BN was learned and used. The TBNL library includes all the necessary procedures to execute the TBNL algorithm—the BNT library includes all the necessary procedures to run a general BN learning and the SLP library includes specific procedures of learning algorithms and inference that were used in this case study(see also Murphy 2004).

The simulation output was structured and aggregated and then used as an input to the TBNL algorithm. Operational data were aggregated on a monthly resolution, so that at the end of each month a new record of all operating systems was obtained. A data record included features for each LRU, such as the number of failures and the time since the last restoration (the name of the LRU type with "_TSR" extension), as well as the following attributes: wagon ID, TIME, wagon Type, wagon STATUS (i.e., whether the system has currently failed or is available), PM_RECENTLY (i.e., whether the wagon underwent PM in the preceding month), TIME_SINCE_PM (i.e., the time since last PM), PM_TOTAL (i.e., the number of PM operations in the wagon's history), accumulated MILEAGE, and, as the target variable for the learning algorithm, FAILURE, which indicates whether the wagon has failed during the last month (even if it was restored by the end of the month). Note that some attributes do not necessarily cause a failure, but it is up to the BN to determine whether they are statistically dependent. It is also worth noting that the components' ages are not included in the model because in practice it is not realistic to assume that they are known. The ages are monitored within the simulation and are often not obtained by the operator.

Because the focus of the metamodel is on failure prediction given the system conditions' vector (of any wagon out of the fleet), the BN model uses the wagon's condition over a month to predict the wagon's state in the next month (classifying it either as "failed" or "available"). Modeling these sets of input–output vectors determines whether or not a given condition eventually leads to failure.

For compatibility with the BN form, continuous data, such as the TIME_SINCE_PM and MILEAGE, are discretized using a supervised discretization algorithm (Ching et al. 1995). The supervised criterion used by this algorithm aims at maximizing the mutual information in the discretized variable about the target variable, normalized by their joint entropy. During the discretization, the number of symbols of each variable is bounded (in this case up to 20). The scheduled PM policy for the BN model was constructed from a 5-year policy. The main constraint with which the TBNL was executed was a MinPRIG of 2% for the target variable, as well as for the rest of the attributes (MaxPRIE value of 100%). The graph of the learned BN following this learning process is shown in Figure 6. Based on this network, the CPT were estimated for the nodes conditioned on the parent nodes. This resulted in a significantly smaller number of probability estimates in comparison to the joint probability table of all the variables (a detailed view of a BN with CPT was provided by in Ben-Gal 2007).

The resulting BN included five features (in addition to the target variable). The features that were selected by the TBNL were the wagon status (referred to as STATUS); the time since the last PM of the wagon (referred to as TIME_SINCE_PM); the time elapsed since the last restoration of the running gear of type 1 frame (referred to as Running_Gear_Uc_TSR) and of the wheelsets of type 2 frame (referred to as Wheelsets_Upp_TSR); and the wagon mileage (referred to as Mileage). Recall that continuous variables were discretized using the algorithm of Ching et al. (1995), and the number of distinct states may vary accordingly. The PRIE of the failure expectancy was slightly above 32%, which means that the selected attributes provided nearly a third of the potential information about the failure prediction. All of the rest of the features were not selected by the TBNL and thus were not shown in the BN. If, for example, the variables' realizations in particular states were Mileage = 88,020 miles, Wheelsets_Upp_TSR = 15 years, Running_Gear_Uc_TSR = 1 year, TIME_SINCE_PM = 4.89 years (about 4years and 11months), and the STATUS = 1 (namely, the wagon is currently available), then the conditional probability of a failure would be 0.78, which would be classified as failed for a threshold value of 0.5. Thus, in this case a PM task would be performed on the observed wagon.

Note that the classification threshold in the above example can be treated as a PM parameter and modified by the user to trigger different PM policies, as shown in Figure 7.

At this stage one can further fine-tune the BN model to obtain a better failure prediction performance by using, for example, a five-fold cross validation test. Thus, the data were divided into five subsets, where four-fifths were used for training the BN model and the remaining fifth was used as the test set in turn, as commonly done with classification solutions (Maimon and Rokach 2005). However, in this case we tested not only the classification accuracy but generated a receiver operating characteristic (ROC) curve (see Figure 7) that draws the recall versus the false-positive rate (FPR) of the model (see Green and Swets 1966). The recall represents the ratio of correctly predicted failures when such occur, whereas the FPR is the ratio of falsely predicted failures in cases where the system did not fail.

Having tested the TBNL resulting model as a stand-alone element (module III), we integrated it with the simulation. The integration was performed

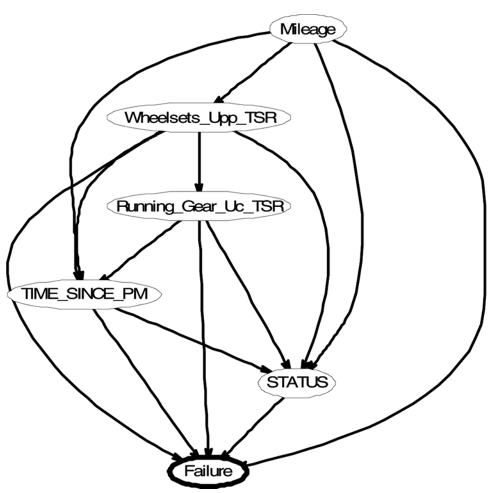

FIGURE 6 Graph of the resulted BN model.

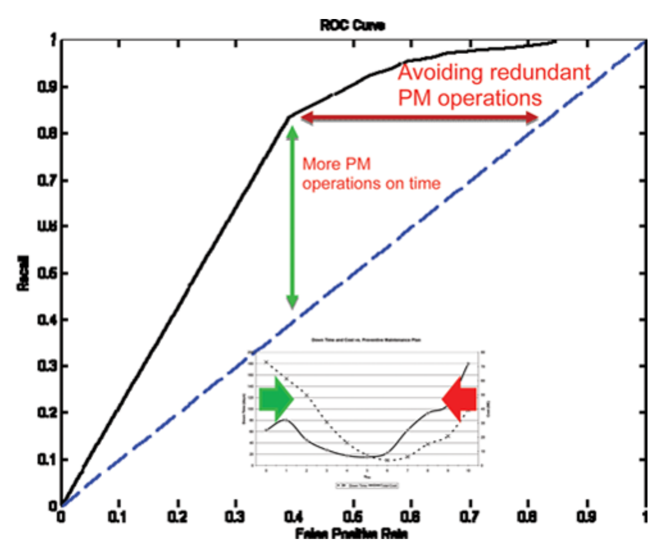

FIGURE 7 ROC curve of the learned BN and its CBM consequences. (Color figure available online.)

by plugging the CBM model into the simulation module. Each month, the condition vector of every single wagon was inputted to the CBM model, which returned the corresponding failure probability of the wagon for the following month. This probability was then used for classifying whether the wagon should be sent to a preventive maintenance operation according to a predetermined PM policy. This approach was found to yield the best performance (in terms of profit) out of a batch of evaluated policies. Each policy was predetermined by a different level of decision threshold, as described in module IV. The best performance was achieved by a decision threshold policy of 40% failure probability. Namely, if the failure probability of a wagon was equal to 40% or more, it was sent to PM (referred to as CBM-tbnl-Thresh 40 policy). In this case study, all of the components in a wagon underwent PM upon PM operation.

A performance bench mark of the resulted CBM policy was evaluated by comparing all of the decision threshold policies with an upper bound result and with a lower bound result of the profit. The upper bound was obtained by simulating a scenario in which failures are perfectly predicted and the corresponding PMs are undertaken just beforehand. This can be artificially obtained by exploiting the technical details of the simulation; that is, accessing the stack of future events.

There are actually two lower bounds of the PM policy or, more precisely, the lower bound is obtained by the higher profit out of two extreme PM policies. Recall that in Figure 2, one extreme policy is to seldom perform PMs (denoted in Figure 8 as "No PM"), where as the second extreme policy is to perform PMs every time the system is inspected (denoted in Figure 8 as "Too often PM"). The system availability for all the above considered policies is presented in Figure 8.

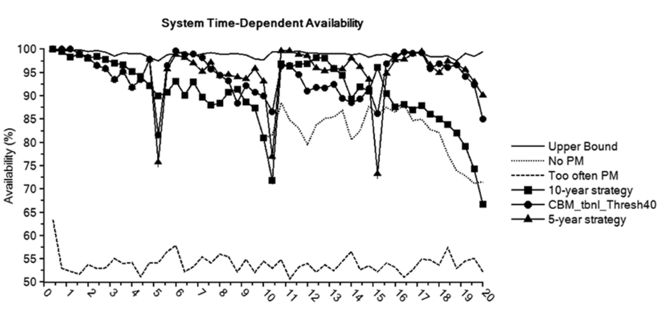

FIGURE 8 **Wagon's availability across various PM management policies.**

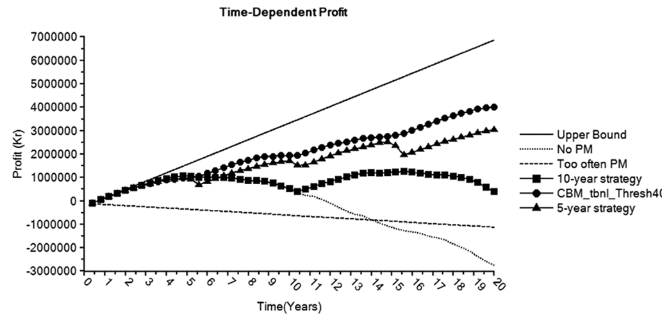

FIGURE 9 **Operator profit across various PM management policies.**

Note the slight difference between minimizing the system downtime vs. minimizing the LCC, as shown in Figure 2. The profit is obtained by the simulation according to the contract detailed in Figure 5, while the rest of the cost contributors of the LCC, as formulated in Eq. [1], are subtracted (the corrective and preventive maintenance elements). This yields an objective function that can be positive or negative, while the goal is to maximize it, as can be seen in Figure 9.

DISCUSSION

In this section we discuss the architectural choices we made in the proposed solution. In order to overcome previously mentioned difficulties in performing RAM optimization of real complex systems, we combine two types of modeling tools in the proposed methodology: MCS and the targeted BN. Each of these tools has its pros and cons, leading to the conclusion to use each tool for a different purpose.

The MCS is used to create the most accurate and reliable model of the real-life system, expressing the existing couplings and complex relations among different system components, as well as key performance indicators. Apart from the ability to collect all of the operational attributes that are required for the prognostics, this type of modeling tool serves as an experimental lab, enabling analysis of how different settings might influence the system. These types of experiments would be very risky and difficult, sometimes impossible, if performed on the actual system.

The targeted BN model, built by the TBNL algorithm based on operational data, is a surrogate model in the suggested framework; hence, it can be less accurate in assessing the behavior of the entire system, but it is a more efficient and descriptive modeling tool and more accurate in predicting the failures using the selected variables (such as MILEAGE, TIME_SINCE_PM, etc.) as descriptors. The TBNL algorithm also takes into account the variables' complex interactions and provides an interpretable graphical representation, making the CBM model accessible for experts' inspection and validation.

The combination of the MCS and the TBNL enables exploiting the advantages of each of these models and to obtain a robust framework for a CBM application. On the one hand, the TBNL is efficient in predicting the failures, given the conditions vector of the wagon; thus, it is used mainly as a surrogate CBM prognosis model in this framework. On the other hand, the MCS is accurate in assessing the complex system behavior and provides the necessary framework for the examination of various key performance indicators under different operational policies.

The tuning is performed in module III, in which the targeted BN model is tuned to provide a good prediction performance, by using cross-validation methods with the provided operational data. This module, which is performed outside of the simulation, is computationally cheaper than using the full simulation scheme, and it is used as a unit test for the prognosis model. Module III is used for selecting the appropriate descriptors, and tuning the parameters of the prognosis model for the best possible performance.

In module IV, the prognosis model is plugged into the MCS to examine its overall influence on the system and to find the most suitable CBM-based operational policy that would provide the desired results.

The resulting BN clearly reflects the efficiency of the TBNL for such purposes. Given a constraint of a minimum 2% of information gain, with respect to each variable, the network becomes very simple and compact. This constraint of 2% minPRIG was set after some parameter exploration, where it was mainly set in order to avoid noise. Although the resulted BN is very compact and exploits slightly over 32% of information regarding the target variable, it suffices to best predict failures under the specified constraints. The advantage of the network's compactness is reflected by the cheap computational complexity while using it for inference. The complexity of the problem, as defined in Eq. [2], is 7 M bits, whereas the complexity of the resulted BN is 18 K bits (99.7% less). It can be also indicated that the selected features were a mixture of physical components and some other expected attributes. It would be extremely surprising if an expert could point out this exact mixture.

The resulting BN was benchmarked against two popular BN models: the Tree Augmented Network (TAN) and naïve Bayes algorithms. The accuracies of each type of BN models were 96, 91, and 90%, respectively. However, the ROC curve shown in Figure 7 further emphasizes the contribution of using the TBNL as a decision support model for PM management. Each point on the ROC curve indicates a tradeoff between the recall (a.k.a. sensitivity), which is the rate of detected failures out of those failures that did occur versus the FPR, which are the cases in which failures were falsely anticipated. In Figure 7 it is illustrated that by increasing the recall (i.e., the true-positive rate) while maintaining limited FPR pushes the CBM model to the optimal PM policy. This stems from the fact that there are two extreme policies that provide a lower bound on the system's performance. One extreme PM policy is to maintain very unsatisfactory or not at all. This could be regarded as if the CBM model underestimated failures, resulting in an increasing failure rate. The second extreme PM policy is to over maintain. This could be regarded as if the CBM model overestimated failures, resulting in a highly reliable, yet unavailable and expensive system. The more accurate the targeted BN model is, the better the tradeoff achieved between recall and FPR, pursuing the optimal PM policy.

The optimal PM policy is reached when the profit is maximized. We pursued this upper bound with the policy that attempts to maximize the availability. More precisely, the upper bound was estimated by aborting anticipated failures as closer as to their failure time. Hypothetically, there might be situations in which the profit could be higher, because the components that put the profit together, namely, the availability, failures, and PM actions, are all interdependent and are hard to break down. However, the upper bound was obtained via the simulation model, simply by aborting each failure ad hoc and maintaining the system instead. In between the upper bound and the two lower bounds we ran two scheduled PMs, one every 10 years (referred to as the 10-year policy) and one every 5 years (referred to as the 5-year policy). The TBNL follows the 5-year policy but exploits more information and manages better future analysis. Therefore, although its overall availability is not as good as that of the original 5-year policy, the profit

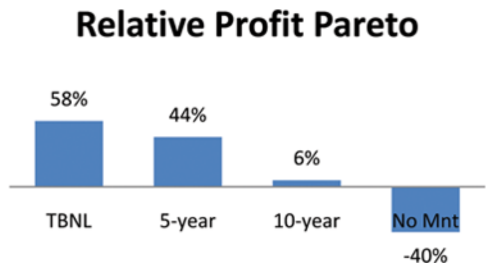

FIGURE 10 Profit Pareto of each considered PM policy relative to the upperbound. (Color figure available online.)

related to it is boosted. Figure 10 shows the relative profit cumulated throughout 20 years of service time.

The TBNL fulfills the contract in a more effective manner with respect to the profit, mainly because it spreads the PM actions slightly more than the 5-year policy does. The 5-year policy suffers from profit drops, because the penalty of maintaining a large portion of the fleet altogether is high. As a result, the TBNL improves the profit by 18% compared to the 5-year policy, where the 100% line represents the profit that could have been gained by the upper bound policy. Note that the 10-year policy is considerably minor in profit and the lower bound of no PM represents a loss of 40%.

CONCLUSIONS AND FUTURE WORK

This article proposes a CBM framework based on an MCS model of the system and on a predictive analytics engine. The latter is a targeted BN that learns from data generated by the simulation, thus exploiting possible contingencies that are essential for determining the PM policy. On the other hand, the targeted BN model is used solely for the specific purpose of failure prediction and hence it is used as a meta model in the framework rather the model itself. As a result, the learned BN model is more effective when considering the failure causes.

Although the proposed architecture does not necessarily provide an optimal PM policy, this work provides a proof-of-concept of a tool that enables an effective design of a CBM policy. The added value of the proposed architecture is twofold: (1) because the CBM model is learned from simulation data of a modeled system, it enables exploring PM policies and scenarios that might not have been considered by analyzing real operational data; (2) the TBNL learns a predictive model from the simulation data that efficiently focuses on failure prediction conditioned on the system state, rather than learning a model of the entire operational and environmental system, and is used specifically as a CBM meta model.

We anticipate two possible directions for future work. One direction is using the approach's concept in a dynamic fashion, so that the BN model is learned or updated in a timely manner. Another direction is an exploration of more informative features that would potentially yield more information and thus improve the prediction efficiency.

ABOUT THE AUTHORS

Dr. Aviv Gruber received the B.Sc. (2000 and the M.Sc. (2003) in Nuclear Engineering from Ben-Gurion University of the Negev and the Ph.D. (2013) in Industrial Engineering from Tel Aviv University. His Ph.D. thesis combines applications from Machine Learning and Information Theory to industrial domains. Currently, Dr. Gruber is in a position of a visiting scientist in the Faculty of Industrial Engineering at the Technion institute, and his research is focused in multi-fields and multi-echelons supply chain. Dr. Gruber is a Modelling & Simulation expert of industrial systems. Experience: a consultant and a senior tutor, and an algorithms developer. Senior tutor in the Computer Integrated Manufacturing (CIM) Laboratory, Developer in the Advanced Analytics group at Intel Inc. Awarded first prize at the IE & M 2010 and third prize at the IE & M 2011 for an outstanding research work.

Shai Yanovski discovered the field of data mining a decade ago, and has been passionate about it ever since. He received his B.Sc. in Industrial Engineering from Tel Aviv University, and continued his research and development activities in the field as a member of Tel-Aviv University's Applied Machine Learning Group led by Prof. Irad Ben-Gal. and via collaborated with a range of companies in the private sector where he was involved in a series of projects applying machine learning techniques to problems in a variety of domains such as automatic information retrial from scientific papers, nano-particles toxicology, military helicopter fault prediction, terrorist activity detection in communication networks and more. Currently (2012) he is holding a lead data scientist position in Lacoon Security LTD where he is applying predictive analytic techniques on large volumes of data for solving challenges in the domain of cyber security.

Professor Irad Ben-Gal is the Head of the Department of Industrial Engineering & Management at Tel Aviv University. His research includes statistical methods for control and analysis of complex processes, machine learning applications to industrial and service systems, and big data analytics. He has written and edited five books, has published more than 80 scientific papers and patents, has received numerous best papers awards, and has supervised dozens of graduate students. He is a department editor in *IIE Transactions on Quality and Reliability* and serves on the editorial boards of several other international journals.

Professor Ben-Gal is the Chair-Elect of the Quality Statists and Reliability society at the Institute for Operations Research and Management Sciences (INFORMS), The European Network for Business and Industrial Statistics (ENBIS), and the International Statistical Institute (ISI). He received several research grants and awards, among them from General Motors, IEEE, Israel Ministry of Science, Israel Scientific Foundation (ISF), Israeli Prime Minister Ministry, and the European Community. He has led many R&D projects and worked with companies such as IBM, Pratt & Whitney, Siemens, Proctor & Gamble, Kimberly-Clark, Applied Materials, Erikson, SAS, SAP, and more. Irad is a cofounder of Context-Based 4 casting (C-B4), a software company that develops novel predictive analytics tools for retail and service organizations.

REFERENCES

Arnaiz, A., Ferreiro, S., Buderath, M. (2010). New decision support system based on operational risk assessment to improve aircraft operability. *Proceedings of the Institution of Mechanical Engineers - Part O: Journal of Risk and Reliability*, 224(3):137–147.

Balakrishnan, N., Varadan, J. (1991). Approximate MLEs for the location and scale parameters of the extreme value distribution with censoring. *IEEE Transactions on Reliability*, 40(2):146–151.

Barata, J., Soares, C., Marseguerra, M., Zio, E. (2002). Simulation modelling of repairable multi-component deteriorating systems for ''on condition'' maintenance optimisation. *Reliability Engineering & System Safety*, 76(3):255–264.

Ben-Gal, I. (2007). Bayesian networks. In: Ruggeri, F., Faltin, F., Kenett, R., Eds. *Encyclopedia of Statistics in Quality and Reliability*, London: John Wiley & Sons.

Bengtsson, M. (2004). Condition based maintenance system technology—Where is development heading? Paper read at the 17th European Maintenance Congress, May 11–13, 2004, Barcelona, Spain.

Brotherton, T., Jahns, J., Jacobs, J., Wroblewski, D. (2000). Prognosis of faults in gas turbine engines. Paper read at the IEEE Aerospace Conference, March 18–25, 2000, Big Sky, MT.

Chickering, D. M., Geiger, D., Heckerman, D. (1995). Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20:197–243.

Ching, J. Y., Wong, A. K.C., Chan, K. C.C. (1995). Class-dependent discretization for inductive learning from continuous and mixed mode data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(7):641–651.

Claeskens, G., Hjort, N. (2003). The focused information criterion. *Journal of the American Statistical Association*, 98:900–945.

Deuszkiewicz, P., Radkowski, S. (2003). On-line condition monitoring of a power transmission unit of a rail vehicle. *Mechanical Systems and Signal Processing*, 17:1321–1334.

Dubi, A. (2000). *Monte Carlo Applications in Systems Engineering*. New York: John Wiley & Sons. pp. 125–138, 145–155.

Duda, R., Hart, P. (1973). *Pattern Classification and Scene Analysis*. NewYork: Wiley.

Forrester, A. I. J., Keane, A. J. (2009). Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45(1–3):50–79.

Green, D. M., Swets, J. A. (1966). *Signal Detection Theory and Psychophysics* New York: John Wiley & Sons.

Gruber, A., Ben-Gal, I.(2011). Managing the trade-off between complexity and accuracy of Bayesian classifiers. Paper read at the 5th Israeli Industrial Engineering Research Meeting (IIERM), March 16–17, 2011, Dead-Sea Resort, Israel.

Gruber, A., Ben-Gal, I. (2012). Efficient Bayesian network learning for optimization in systems engineering. *Quality Technology and Quantitative Management*, 9(1):97–114.

Hansen, C., Autar, R., Pickles, J. (1994). Expert systems for machine fault diagnosis. *Acoustics*, 22:85–90.

Heckerman, D. (1995). *A Tutorial on Learning with Bayesian Networks*. MS TR-95–06. Redmond, WA: Microsoft Research.

Jardine, A. K.S., Lin, D., Banjevic, D. (2006). A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, 20: 1483–1510.

Keane, A., Nair, P. (2005). *Computational Approaches for Aerospace Design: The Pursuit of Excellence* Chichester, UK: John Wiley & Sons.

Kelton, W. D., Law, A. M. (2000). *Simulation Modeling and Analysis* Boston: McGraw Hill.

Li, Y., Billington, S., Zhang, C., Kurfess, T., Danyluk, S., Liang, S. (1999). Adaptive prognostics for rolling element bearing condition. *Mechanical Systems and Signal Processing*, 13:103–113.

Liu, Y., Li, S. Q. (2007). Decision support for maintenance management using Bayesian networks. In: *Proceedings of the International Conference on Wireless Communications, Networking and Mobile Computing*. 5713–5716. September 21–25, 2007, Shanghai.

Maimon, O., Rokach, L. (2005). *The Data Mining and Knowledge Discovery Handbook* Heidelberg, Germany: Springer.

Marseguerra, M., Zio, E., Podofillini, L. (2002). Condition-based maintenance optimization by means of genetic algorithms and Monte Carlo simulation. *Reliability Engineering & System Safety*, 77(2):151–165.

Moubray, J. (1991). *Reliability-Centered Maintenance* Oxford, United Kingdom: Butterworth-Heinemann.

Murphy, K. P. (2004). *Bayes Net Toolbox v5 for MATLAB*. Cambridge, MA: MIT Computer Science and Artificial Intelligence Laboratory. Available at: http://www.cs.ubc.ca/~murphyk/Software/BNT/bnt.html. (accessed June 14, 2009)

Nowlan, F. S., Heap, H. F. (1978). *Reliability-Centered Maintenance*. Report Number AD-A066579. Washington, DC: U.S. Department of Defense.

Paya, B., Esat, I., Badi, M. (1997). Artificial neural network based fault diagnostics of rotating machinery using wavelet transforms as a preprocessor. *Mechanical Systems and Signal Processing*, 11:751–765.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* San Francisco: Morgan Kaufmann.

Peng, Y., Dong, M., Zuo, M. J. (2010). Current status of machine prognostics in condition-based maintenance: A review. *International Journal of Advanced Manufacturing Technology*, 50(1–4):297–313.

Prescott, P., Draper, N. R. (2009). Balanced, nearly optimal mixture experiments for models with interactions with process variables. *Quality Technology and Quantitative Management*, 6(2):67–86.

Queipo, N., Haftka, R., Shyy, W., Goel, T., Vaidyanathan, R., Tucker, P. K. (2005). Surrogate-based analysis and optimization. *Progress in Aerospace Sciences*, 41(1):1–28.

Samanta, B., Al-Balushi, K. (2003). Artificial neural network based fault diagnostics of rolling element bearings using time-domain features. *Mechanical Systems and Signal Processing*, 17:317–328.

Saranga, H., Knezevich, J. (2000). Reliability analysis using multiple relevant-condition parameters. *Journal of Quality in Maintenance Engineering*, 6(3):165–176.

Sheppard, J., Kaufman, A. (2005). Bayesian diagnosis and prognosis using instrument uncertainty. Paper read at AUTO-TESTCON, 2005, September 26–29, 2005, Orlando, FL.

Spoerre, J. (1997). Application of the cascade correlation algorithm (CCA) to bearing fault classification problems. *Computers in Industry*, 32:295–304.

Tsang, A. H. C. (1995). Condition-based maintenance: Tools and decision making. *Journal of Quality in Maintenance Engineering*, 1(3):3–17.

Vining, G. G., Cornell, J. A., Myers, R. H. (1993). A graphical approach for evaluating mixture designs. *Applied Statistics*, 42(1):127–138.

Wang, H. (2002). Survey of maintenance policies of deteriorating systems. *European Journal of Operational Research*, 139(3):469–489.

Wang, H., Pham, H. (1997). Survey of reliability and availability evaluation of complex networks using Monte Carlo techniques. *Microelectronics Reliability*, 37(2):187–209.