https://doi.org/10.1007/s10845-015-1143-4

Probabilistic Boolean network modeling of an industrial machine

Pedro J. Rivera Torres[1] · **E. I. Serrano Mercado**[2] · **Luis Anido Rifón**[1]

Received: 18 June 2015 / Accepted: 18 August 2015 / Published online: 30 August 2015

Abstract Theoretical modeling of manufacturing processes assists the design of new systems for predictions of future behavior, identifies improvement areas, and evaluates changes to existing systems. A novel approach is proposed to model industrial machines using probabilistic Boolean networks (PBNs) to study the relationship between machine components, their reliability and function. Once a machine is modeled as a PBN, through identification of regulatory nodes, predictors and selection probabilities, simulation and property verification are used to verify model correctness and behavior. Using real machine data, model parameters are estimated and a PBN is built to describe the machine, and formulate valid predictions about probability of failure through time. Two models were established: one with non-deterministic inputs (proposed), another with components' MTBFs inputs. Simulations were used to generate data required to conduct inferential statistical tests to determine the level of correspondence between predictions and real machine data. An ANOVA test shows no difference between expected and observed values of the two models (p value = 0.208). A two-sample T test demonstrates the proposed model provides values closer to expected values; consequently, it can model observable phenomena (p value = 0.000). Simulations are used to generate data required to conduct inferential statistical tests to determine the level of correspondence between model prediction and real machine data. This research demonstrates that using PBNs to model manufacturing systems provides a new mechanism for the study and prediction of their future behavior at the design phase, assess future performance and identify areas to improve design reliability and system resilience.

Keywords Bio-inspired modeling · Biological manufacturing systems · Probabilistic Boolean networks

Introduction

Probabilistic Boolean networks (PBN) are mathematical constructs that can be used to model Gene Regulatory Networks (GRN). GRNs are collections of DNA segments within a cell that interact indirectly with other segments and substances in a cell in order to govern the expression levels of genes. They can be used to better understand the general rules that govern gene regulation in genomic DNA. PBNs are transition systems that satisfy the Markov Property (Markov 1954), such that the probability that the system will take a transition from a given state to another depends exclusively on the current state, and is not dependent on the past history of the system. PBNs were proposed by Ilya Shmulevich in several publications (Shmulevich et al. 2002a, b; Shmulevich and Dougherty 2010) as an extension of Stuart Kauffman's Boolean Network (BN) concept (Kauffman 1969a, b). This alternative to modeling GRNs combines the rule-based modeling of Kauffman's BNs with uncertainty principles. PBNs consist of a group of constituent BNs that have assigned selection probabilities, where each Boolean Network can be considered a "context". Data for the cells comes from different sources; each source represents a context of the cell (Shmulevich and Dougherty 2010). In a given time t, a system can be governed by one of the constituent BNs, and at

✉ Pedro J. Rivera Torres
privera@ece.uprm.edu

[1] ETSET-Universidade de Vigo, Campus Universitario, s/n, 36310 Vigo, Spain

[2] Polytechnic University of Puerto Rico, 377 Ponce De Leon Ave., San Juan, PR 00918, USA

any other time the system may switch to another constituent BN with a given switching probability. Qualitative frameworks, such as Boolean Networks (Ristevski 2013) permit the description of large biological networks without losing important system properties.

Although, it is frequent to find several applications of BNs and PBNs in Systems Biology to model GRNs, applications or uses outside this realm remain vastly unexplored. One of the few studies venturing on the application of PBNs outside GRNs proposed to model credit defaults (Gu et al. 2013). A PBN-based model was applied to study the link between correlated defaults of different industrial sectors and business cycles, and the impact of these cycles on modeling and predicting defaults. With PBNs, a transition probability matrix that describes the correlated defaults of the business sectors studied was determined and decomposed into several BN matrices that house information about business cycles. Actual default data is used to build the PBN to explain the default structure, and achieve predictions of joint defaults in different business sectors. In this same area of application, Liang et al. (2014) concentrates on the construction of PBNs from credit default data and presents a heuristic construction algorithm. These recent studies provide a baseline to expand further the utility of PBNs. This paper proposes the application of PBNs as a mechanism to model industrial machines.

Industrial manufacturing environments are complex and dynamic due to constant changes in customer expectations and demands, different product types, features and suppliers, and the unexpected disturbances inherent to the manufacture and assembly of products. It is desirable to obtain mathematical models that aid the study of the manufacturing process operation under a set of specific conditions and modes, taking into account the elevated complexity inherent of such a system. In this way, fulfillment of design requirements, alternative design proposals, and the study and control of operating environmental conditions is facilitated. Modeling, paired with simulation, permits the study of behaviors and dynamics, among other factors, in a virtual environment. Modeling is relevant in manufacturing to aid the design of new manufacturing systems, to make predictions of future behaviors of a system, identify areas of improvement, and also, as a mechanism to assess changes on existing systems.

PBNs are proposed here as a simplified representation of a Pick and Place machine's dynamics in order to use this representation to model and predict system behavior through the use of simulation and analysis. The machine chosen is frequently used in manufacturing processes, where different machines are integrated, to load and unload parts in assembly processes. The machine has several components with known reliability; however, it is relevant to assess the reliability and interaction of the integrated components functioning as a system. PBNs are a relevant and appropriate method for modeling industrial machines, because of the similarities in characteristics between them: they are both stochastic, dynamic, and exhibit rule-based, state-transition behavior. The application of PBNs to industrial machines may thus allow improvements in terms of design, maintenance, reliability, availability, and other performance factors.

The main contribution of this paper is the application of a biomimetic modeling methodology that accurately predicts the next state of the modeled machine based on logical rules determined by the physical relationship between its components. This model characterizes machine components into two simple states: operational or failed, and establishes sets of simple logical, predictive functions that define constituent networks, thus allowing for the study of the steady-states of the machine. In turn, a designer or engineer may make educated design decisions about the future behavior of the machine that can result in better reliability-based decisions, including maintenance, time to failure, and risk analysis.

Related work and theoretical background

Bio-inspired techniques in manufacturing systems

As a way of coping with changes endemic to manufacturing, technologies and methodologies have been proposed that mimic biological processes. Genetic (Booker et al. 1989), Biological (Park and Tran 2010), and Holonic (Babiceanu and Chen 2006) Manufacturing Systems mimic the organizational structures and mechanisms seen in biological systems and transfer them into manufacturing environments. Concepts such as Genetic and Evolutionary Algorithms (Kumar and Dhingra 2012 Artificial Neural Networks (ANN) (Assef et al. 1996; Barghash and Santarisi 2004; Leger et al. 1998; Skitt et al. 1993; Sood 2013; Sun et al. 2014), Ant Colony Optimization (Dorigo and Blum 2005), Particle Swarm Optimization (Liu et al. 2011), and Petri nets (Moore and Gupta 1996), among others, are used to implement intelligent functionalities. It is relevant to review previous research performed in biological manufacturing systems (BMS). All these modeling techniques use biomimetics, emulating the behavior of biological systems and using this behavior to solve problems in other fields of knowledge. The concept of biological manufacturing systems (Ueda 1992, 1993, 1994, 1997) has been widely discussed in the literature. It was introduced to research the feasibility of creating dynamic and adaptable manufacturing systems through the use of elements that are related to biological processes and organisms such as, evolution, self organization and self-growth, and adaptation. In this way, part transporters (product carriers) can self-organize towards specific manufacturing facilities/units (product processors), through attractions fields (machine tools, robots) that these generate (using the concept of attraction–repulsion fields from biology). In a living organism, a stimulus from the environment causes a behavioral response. In BMS, the response can be a product or manufactured good. The procedures that generate the response can be described as a network of local procedures that in turn lead to a response at the macro level. For a biological system, the responses are interactions within the cell, but in manufacturing systems, these can be subtasks that are performed to assemble a final product, performed by subunits that have operations similar to biological cells, differentiated for particular tasks. The manufacturing cells "attract" the components needed and "repulse" the products. Cells and subcomponents flow in a manufacturing plant floor, moving each other until a stable condition is reached. All BMS elements (e.g. materials, machine tools, transporters, robots) have an equivalent and are comparable to autonomous biological organisms.

Relevant research includes the application of algorithms that are inspired on the behavior of insects, such as Ant Colony Optimization, and Particle Swarm Optimization. In particular, Ant Colony Optimization (Dorigo 1992) has been used optimize layouts of machines (Corry and Kozan 2004), for process planning optimization (Liu et al. 2013) and wasps for task allocation and factory routing and scheduling (Cicirello and Smith 2001a, b). Particle Swarm Optimization has been applied to the detection of faults in machines (Samanta and Nataraj 2009), inspection of component placement in Printed Circuit Boards (Wu et al. 2009) and the flexible job-shop scheduling problem (Nouiri et al. 2015), among other applications. Swarm Intelligence techniques such as these are, as PBNs, stochastic in nature, but center on agency and definition of behaviors. PBNs offer a predictive, more elegant approach that does not require defining agents or behaviors. ANNs attempt to simulate either the structural or functional aspects of the central nervous systems, and like GRNs in general, have features like flexibility and robustness, which are endemic to biological systems. ANNs are trained by means of techniques that are influenced by evolution. ANNs have a complex structure, and BNs can provide a simpler model of GRNs.

Based on previous studies, there is ample evidence to demonstrate that biomimetic algorithms are appropriate mechanisms for manufacturing systems modeling.

Bayesian Networks have also been discussed extensively in manufacturing scientific literature, especially in reliability related topics (Huang et al. 2008; Mosallam et al. 2014; Tchangani 2004). Dynamic Bayesian Networks (DBN) have been shown to model GRNs in the same way as PBNs can (Li et al. 2007). In said paper, PBNs and DBNs were compared using a biological time series set of data taken from the Drosophila Interaction Database, which served as the basis to evaluate the performance of both approaches. Both were found to have good performance, and while DBN identified a larger number of gene interactions, their accuracy depended on the particular DBN inference algorithm selected, making DBNs more time consuming.

In addition, Multi-Agent Systems (MAS) (Wooldridge 2002) have been applied to manufacturing systems as a framework to develop adaptive, robust and reconfigurable manufacturing systems. These have a distributed nature because they are based on a group of autonomous agents that cooperate, determining the function of the system from their interactions. The agents divide their labor by defining distinctive roles, behaviors, goals and skills, as in insect colonies. MAS also exhibit emergent characteristics. It is an alternative to centralized control that provides flexibility, adaptability and robustness. Recent research has been conducted about MAS applied to manufacturing systems (Ayhan et al. 2013; Hsieh and Lin 2013, 2014; Wang et al. 2014).

MAS can be coupled with self-organization, thus enabling agents to achieve self-configuration, self-optimization, and self-healing (Leitão 2008). In multi-agent systems, a holonic agent is one that can be composed of other agents. The term holon, introduced by Arthur Koestler in his book "The ghost in the machine" (Koestler 1967), points to structures that are neither parts or whole in absolute terms. Holons are classified in holarchies; ontologies or classifications of agents, where a sub component cannot be understood completely without its super-component. Holonic Multi-agent Systems are the cornerstone of Holonic Manufacturing Systems (HMS). HMS are a very active field of study in manufacturing systems.

MAS and HoloMAS are complex-adaptive systems that have been applied extensively in manufacturing science and engineering. PBNs, like MAS, serve to model complex manufacturing systems. PBNs are not agent-based systems but can self-organize as HoloMAS, given that each realization of a PBN is a constituent BN that has transitions that can lead to attractors, the study of these realizations, constituent networks and attractors can yield information about the machine's steady states. Machine states are derived from component states through time. A combination of component states can lead to a particular machine state, which can lead to it being operational or in fault. An analysis can be performed to determine which of these states lead to "healthy" machine states, or which can lead to faults in the machine's operation. Through a separate mechanism called perturbation/intervention, a PBN can be directed to "heal itself". In an open system such as the human genome, there are inputs that are received from outside the system, and these can activate or inhibit the expression of genes. These perturbations can be deliberate (considered then an intervention), and by introducing an intervention vector for a selected set of nodes, the network can be guided to achieve a desired state, or to move from an undesirable one.

The underlying structures in BMS are biomimetic algorithms and techniques, which are used to describe, model, simulate and design manufacturing systems in order to overcome the challenges that these dynamical systems present. Many biomimetic systems are complex, adaptive, intelligent systems, which can self-organize to achieve goals. A system that exhibits self-organization, as described in Gershenson (2007), is "one in which elements interact, achieving dynamically a global function or behavior". Cells (where molecules interact and self-organize to produce life, brains (where neurons interact, producing cognition), insect colonies (where insects interact and self-organize to perform collective tasks), animal swarms, herds, flocks, or schools (where each animal interacts with others to coordinate collective behavior), and other systems of bio-agents, such as ecosystems and societies, are examples of self-organized biomimetic systems.

Compared to previous related work, the approach of this research adds a new perspective to bio-inspired modeling. Instead of simulating the behavior of live organisms to model manufacturing system, this paper is based on the application of models used for GRNs to model manufacturing systems. PBNs are chosen here over other bio-inspired modeling techniques because they predict the behavior of manufacturing systems with good accuracy, based mainly on the logical relationships between system components, without the need of obtaining reliability/failure rate data.

Boolean and probabilistic Boolean networks

Boolean networks

Boolean Networks have been used to model biological systems and its use has been extensively documented in scientific literature (Arnosti and Ay 2012; Bane et al. 2012; Chaouiya et al. 2013; Cheng et al. 2013; Didier and Remy 2012; Ghanbarnejad 2012; Vahedi 2009). probabilistic Boolean networks have also been used extensively to model GRNs (Chen and Sun 2014; Chen et al. 2012; Ching et al. 2009; Gao et al. 2013; Kobayashi and Hiraishi 2010; Trairatphisan et al. 2013).

Boolean Networks, introduced by Kauffman (1969a), are a finite set of Boolean variables for which their state (represented as 0 or 1) can be determined by the state of other variables in the network. Several input genes called regulatory genes/nodes, via a given Boolean function, determine the value of a target gene/node. If the nodes and the corresponding functions are given, then the BN is defined.

Adapted from Ching et al. (2009), formally, a Boolean Network is a graph $\mathrm{G}(\mathrm{V}, \mathrm{F})$ defined by the set $\mathrm{V} = \{\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_\mathrm{n}\}$ that contains the network nodes, and the set $\mathrm{F} = \{\mathrm{f}_1, \mathrm{f}_2, \ldots, \mathrm{f}_\mathrm{n}\}, \mathrm{f}_\mathrm{i} : \{0, 1\}^\mathrm{n} \longrightarrow \{0, 1\}$ of Boolean functions. Each $\mathrm{x}_\mathrm{i} \in \mathrm{V}, \mathrm{i} = 1, \ldots, \mathrm{n}$ $\mathrm{x}_\mathrm{i} \in \mathrm{V}, \mathrm{i} = 1, \ldots, \mathrm{n}$ is a binary variable that represents a gene/node. The rules of interaction between nodes are given by Boolean predictors as such: $\mathrm{x}_\mathrm{i}(\mathrm{t}+1) = \mathrm{f}_\mathrm{i}(\mathrm{x}(\mathrm{t})), \mathrm{i} = 1, \ldots, \mathrm{n}$, where $(\mathrm{x}_1(\mathrm{t}), \mathrm{x}_2(\mathrm{t}), \ldots, \mathrm{x}_\mathrm{n}(\mathrm{t}))$ is a vector that can take any state from the set $\mathrm{S} = \left\{(\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_\mathrm{n})^\mathrm{T} : \mathrm{x}_\mathrm{i} \in \{0, 1\}\right\}$.

Table 1 Boolean network truth table

State	$x_1(t)$	$x_2(t)$	f_1	f_2
A	0	0	0	1
B	0	1	1	0
C	1	0	0	0
D	1	1	1	0

Given an initial state, the BN will transition into a fixed state or set of states within a finite number of steps, known as attractor. Singleton attractors are fixed states, and a set of states is known as a cyclic attractor. Attractors of a BN characterize the networks steady-state or long-run behavior.

As an example, consider a 2 gene BN $G(V, F)$, represented by the set of nodes $\mathrm{V} = \{\mathrm{x}_1, \mathrm{x}_2\}$, and the set of Boolean Functions $\mathrm{F} = \{\mathrm{f}_1, \mathrm{f}_2\}$. Let $\mathrm{x}_\mathrm{i}(\mathrm{t})$ be the state (0 or 1) of the node *i* at time *t*. The rules of interaction among the nodes can be represented by the predictors $\mathrm{x}_\mathrm{i}(\mathrm{t}+1) = \mathrm{f}_\mathrm{i}(\mathrm{x}(\mathrm{t})), \mathrm{i} = 1, 2$ where the vector $\mathrm{x}(\mathrm{t}) = (\mathrm{x}_1(\mathrm{t}), \mathrm{x}_2(\mathrm{t}))$ can take any possible states from the set $\mathrm{S} = \left\{(\mathrm{x}_1, \mathrm{x}_2)^\mathrm{T} : \mathrm{x}_\mathrm{i} \in \{0, 1\}\right\}$, $|\mathrm{S}| = 2^\mathrm{n}$. Table 1 is a truth table listing the possible states, the genes/nodes and the result of the application of the predictor functions that yields the next state.

From the truth table, four states can be determined ((0,0), (0,1), (1,0) and (1,1)), labeled A through D, respectively. If the network is currently in state A, it transitions into state B with a probability of 1. The transition matrix of the BN is given by $\mathrm{M} = \begin{pmatrix} 0\,0\,1\,0 \\ 1\,0\,0\,0 \\ 0\,1\,0\,1 \\ 0\,0\,0\,0 \end{pmatrix}$.

Table 1 gives the one-step transition probability between any given two states. BNs are deterministic models, and each column in M has a single non-zero element. There is only one attractor cycle, with a period of three: $(0, 0) \rightarrow (0, 1) \rightarrow (1, 0) \rightarrow (0, 0)(0, 0) \rightarrow (0, 1) \rightarrow (1, 0) \rightarrow (0, 0)$. The state (1,1) belongs to the basin of attraction of this cycle.

Probabilistic Boolean networks

Extending the BN concept to a stochastic (non-deterministic) model (Ching et al. 2009), for every node x_i in a PBN, there can be many Boolean Predictor functions $f_i^{(j)}$ $(i = 1, 2, \ldots, l(j))$ instead of only one per node as in a BN, that can be chosen to determine the state of node/gene x_j. The probability of choosing $\mathrm{f}_\mathrm{i}^{(\mathrm{j})}$ as the predictor is given by $\mathrm{c}_\mathrm{i}^{(\mathrm{j})}$, where:

$$0 < c_i^{(j)} \le 1, \sum_{i=1}^{l(j)} c_i^{(j)} = 1, \forall j = 1, 2, \ldots, n.$$

Table 2 Probabilistic Boolean network truth table

State	$x_1(t)$	$x_2(t)$	$f_1^{(1)}$	$f_2^{(1)}$	$f_1^{(2)}$	$f_2^{(2)}$
A	0	0	0	1	0	1
B	0	1	1	0	0	1
C	1	0	0	1	1	0
D	1	1	1	0	0	1

Let f_i be the ith possible realization of the network, with $f_i = \left(f_{i_1}^{(1)}, f_{i_2}^{(2)}, \ldots, f_{i_n}^{(n)}\right)$, for each $1 \le i_j \le l(j)$, *and* $j = 1, 2, \ldots, n$. A realization of the PBN is one of the network's constituent BNs. The maximum number of realizations of BNs is $D = \prod_{j=1}^{n} l(j)$.

The probability $c_i^{(j)}$ of selecting a predictor can be approximated statistically with the use of a coefficient of determination and data sets (Dougherty et al. 2000). Given the fact that there are D possible realizations of a BN and these can be characterized by D vector functions $f_1, f_2, \ldots, f_D$, then $f_1 = \left(f_1^{(1)}, f_1^{(2)}, \ldots, f_1^{(n)}\right)$ is the first vector function for the first Boolean Network, and $f_D = \left(f_{l(1)}^{(1)}, f_{l(2)}^{(2)}, \ldots, f_{l(n)}^{(n)}\right)$ is the last vector function from the Dth BN. Assuming that selecting a predictor f_{i_j} for any node j is an independent process, the probability of choosing a BN with predictors $\left(f_{i_1}^{(1)}, f_{i_2}^{(2)}, \ldots, f_{i_n}^{(n)}\right)$ is:

$$u_{i_1 i_2 \ldots i_n} = \prod_{j=1}^{n} c_{i_j}^{(j)}.$$

Transitioning from state to state of S forms a Markov Chain. If **a** and **b** are any two columnar vectors in S, the transition probability is given by $P\{\boldsymbol{x}(t+1) = \boldsymbol{a} | \boldsymbol{x}(t) = \boldsymbol{b}\} = \sum_{i=1}^{D} P\{\boldsymbol{x}(t+1) = \boldsymbol{a} | \boldsymbol{x}(t) = \boldsymbol{b},\ selecting\ the\ i\text{th}BN\} \cdot u_i$ Let $u_i = u_{i_1 i_2 \ldots i_n}$ and $i = i_1 + \sum_{j=2}^{n} \left((i_j - 1)\left(\prod_{k=1}^{j-1} l(k)\right)\right)$. If **a** and **b** can take any of the states in S, the transition probability matrix for the matrix is $A = \sum_{i=1}^{N} u_i A_i$, where A_i is the transition probability matrix of the ith BN, and u_i is the likelihood of selecting the ith BN matrix A_i, where $\sum_{i=1}^{D} u_i = 1, u_i \ge 0$.

To further illustrate how a PBN is constructed given the concepts discussed above, consider the 2-gene PBN from Ching et al. (2009), with the truth table given in Table 2.

In this PBN, each node has two predictors, with $l(i) = 2, (i = 1, 2)$, where

$$D = \prod_{i=1}^{2} l(i) = 4$$

is the amount of realizations of this PBN, which are:

$$f_1 = \left(f_1^{(1)}, f_1^{(2)}\right), f_2 = \left(f_1^{(1)}, f_2^{(2)}\right),$$

$$f_3 = \left(f_2^{(1)}, f_1^{(2)}\right), f_4 = \left(f_2^{(1)}, f_2^{(2)}\right).$$

For each realization of the PBN, there is a transition matrix for every constituent BN, and a selection probability for each network. If $c_1^{(1)} = 0.6, c_2^{(1)} = 0.4, c_1^{(2)} = 0.5$, *and* $c_2^{(2)} = 0.5$, the selection probabilities of each corresponding BN can be computed, being $u_1 = c_1^{(1)} \cdot c_1^{(2)} = 0.3, u_2 = c_1^{(1)} \cdot c_2^{(2)} = 0.3, u_3 = c_2^{(1)} \cdot c_1^{(2)} = 0.2, u_4 = c_2^{(1)} \cdot c_2^{(2)} = 0.2$.

The transition probability matrices of the corresponding constituent BNs are as follows:

$$M_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}, M_2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix},$$

$$M_3 = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}, M_4 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}.$$

State-Transition diagrams can then be used to further illustrate how each BN transitions from state to state through time.

Probabilistic Boolean network modeling of an industrial machine

According to Beaudin (1990), analysis and design of manufacturing systems are processes that produce functional specifications/requirements that are a consistent representation of the system. The machine modeled in this paper, can be thought of as an agent or organism that behaves as an integrated unit or organization. From the field of Distributed Artificial Intelligence, an organization (Chaib-Draa et al. 1992) may be defined as a set of agents that have mutual commitments, global commitments, mutual beliefs and eventually, joint intentions when these agents act together to achieve a given goal. This definition can be extended to incorporate not only agents, but also machines or components within a manufacturing system, where these objects act together to achieve goals. This organization can be thought of as an organism, since as the dictionary definition reflects, organisms are complex structures of interdependent and subordinate elements, whose relations and properties are determined by their function in the whole. It is appropriate to consider of the machine to be presented here as an organism; a complex, adaptive system, that has several interdependent elements, and these elements serve a particular function for the manufacturing machine as a whole. Instead of utilizing holons and holonic ontologies for describing the elements of the system and their interactions, the proposed method utilizes a state-based approach to system decomposition.

Description of the process

According to Jamhour and García (2012), industrial processes can be modeled using finite state machines. Finite state machines are analogous to BN's, as both are directed graphs with rules that govern their transitions. To illustrate the proposed approach, a Pick and Place that loads and/or unloads parts in a manufacturing assembly process is modeled. The "Pick and Place" is a mechanism that has movement in the x and y axes, and through a grip holds, places and removes the parts to and from an assembly line. Initially, designed features and requirements for the Pick and Place and its components is identified according to their intended function and operation.

A model of the Pick and Place has been developed to capture the dynamics and interactions of each of its components using PBN's in a high-level language. PBNs can model the selected machine because of its similarities with GRNs that are modeled using BNs and PBNs, which are:

- GRNs and PBNs are state-based stochastic transition systems, with transitions based on probabilities of occurrence of certain factors.
- Their components/nodes can assume binary states, relationships between nodes can be expressed using Boolean logic,
- Relevant nodes can be considered regulatory nodes.

Model semantics

This research proposes the application of PBNs as a mechanism to model the behavior of a Pick and Place machine by studying the relationship between the state of the machine and its main components. In the context of a manufacturing machine, each of its components is considered equivalent to a node in a PBN. Each node of the system is treated analogous to the gene abstraction in a GRN using binary quantization, where an expressed gene is assigned a value of 1 and an unexpressed gene a value of 0. In this model, when a particular machine component is operating properly, it is analogous to an expressed gene while an unexpressed gene is analogous to a component that is experiencing a failure. Understanding this relationship between the states of the components and states of the components, a transition probability matrix is calculated. The transition matrix describes correlated component states constructed through the application of predictor functions that are stochastically selected. Each realization of the network is a BN that has a set of transitions that represent the possible states of the machine that can be achieved by applying the selected predictors. The transitions probability matrices that compose each realization of the machine lead to attractor or cyclic states. These states are reached though the combined effect of component failures and operation. These are then interpreted as states of the machine. Some of the states of the machine that are described in the transition matrix of each realization are states that equate to a machine failure, and some are healthy states that translate to normal operation of the machine. Figure 1 illustrates key concept of a Probabilistic Boolean Network, a transition from state to state.

Each box shows a state of the Pick and Place at different times. At $t = 0$, the state of the Pick and Place is in normal operation (represented by a '1') as all individual components or nodes are functioning (111111), these nodes are denoted as $x_1 \dots x_6$. The double arrows between boxes denote a transition of the Pick and Place from an operating state at $t = 0$ to failure state (101111) at $t + 1$. This particu-

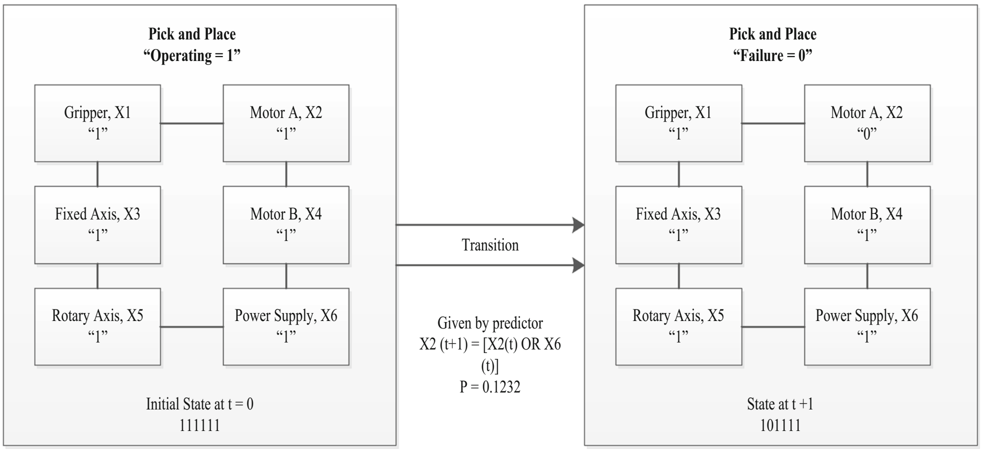


Fig. 1 Semantic model of PBN: example of a transition from $t = 0$ to $t + 1$

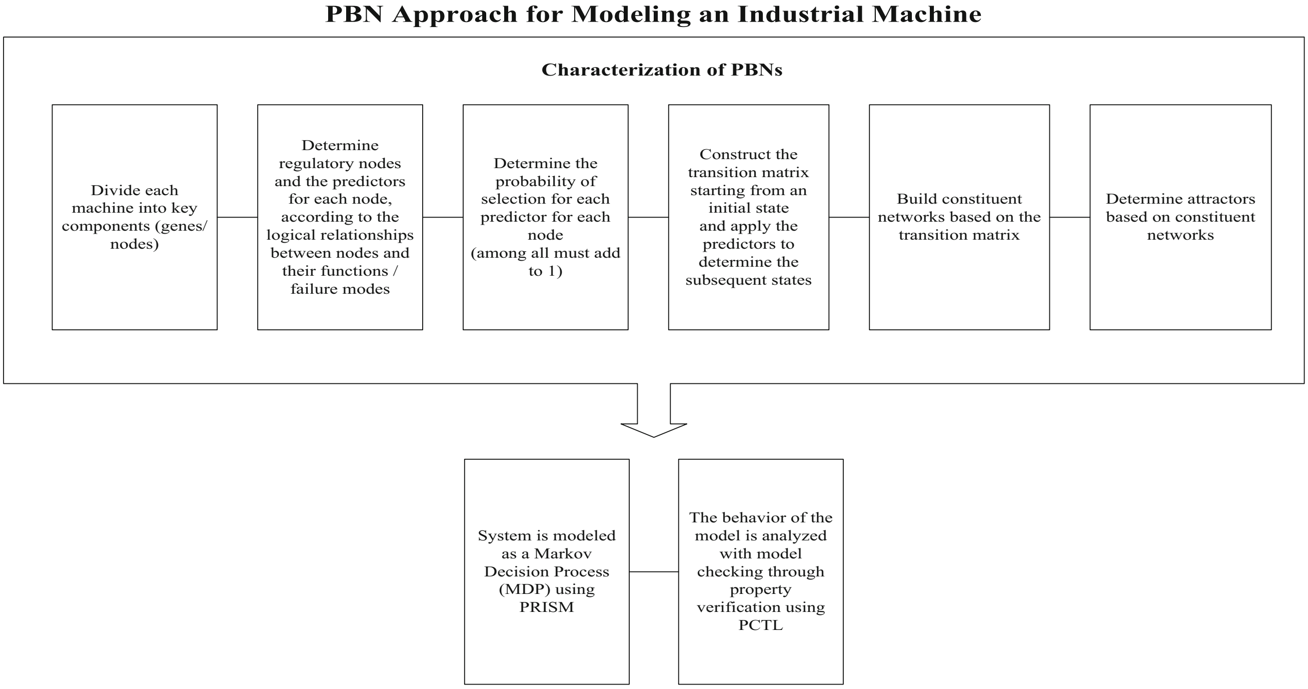

PBN Approach for Modeling an Industrial Machine
Characterization of PBNs
Divide each machine into key components (genes/ nodes)
Determine regulatory nodes and the predictors for each node, according to the logical relationships between nodes and their functions / failure modes
Determine the probability of selection for each predictor for each node (among all must add to 1)
Construct the transition matrix starting from an initial state and apply the predictors to determine the subsequent states
Build constituent networks based on the transition matrix
Determine attractors based on constituent networks
System is modeled as a Markov Decision Process (MDP) using PRISM
The behavior of the model is analyzed with model checking through property verification using PCTL

Fig. 2 PBN approach for modeling an industrial machine

lar transition occurred based on a predictor function given by $x_2(t+1) = [x_2(t) \text{ OR } x_6(t)]$. In physical terms, the predictor function tells that at $t+1$ the Pick and place will be in failure because either Motor A or the power supply failed, which can occur with a probability equal to 0.1232. The Probabilistic Boolean Network, then:

- Is a collection Boolean Networks that consist of a grouping of nodes/genes, such as each components of the Pick and Place (gripper, motor A, and so forth).
- Transitions from state to state in time based on a set that contains the group of Boolean functions or predictors that govern the network in that specific time, such as the transition shown on Fig. 1 on which Pick and Place transitioned from (111111) to (101111).
- Consists of 16 different constituent BNs for the Pick and place, and each has a specific group of predictors. At every time step, a stochastic decision is made on whether or not to continue with the same BN or switch to a new realization.
- Each realization has a different Transition Probability Matrix.

Description of the method

The Pick and Place is modeled as a set of components (the genes of the PBN), and a set of Boolean functions that describe the logical relationships between them. The proposed approach of applying PBN modeling to manufacturing systems is depicted in Fig. 2.

Construction of the PBN: the pick and place

The Pick and Place is modeled as a PBN in the form of $PP = G(V, F)$, where $V = \{x_1, x_2, x_3, x_4, x_5, x_6\}$, and for every x_i in the network a set $F_i = \left\{f_1^{(i)}, \ldots, f_6^{(i)}\right\}$ of Boolean functions is assigned that represents the predictors of target node x_i. The set $F = \{F_1, \ldots, F_6\}$ contains the network's predictors, each F_i in F as described before. At each time step of the network, a function $f_j^{(i)}$ is chosen with a probability $c_i^{(j)}$to predict x_i. The nodes are analogous to the robot's components, where x_1 represents the gripper, x_2 is motor A, x_3 is motor B, x_4 the fixed axis, x_5 the rotary axis, and x_6 represents the power supply.

Predictors in F are Boolean functions, estimated through component relationships and connectivity. In GRNs, the Coefficient of Determination (CoD), as in Dougherty et al. (2000), is used to discover such associations, by measuring the strength of a predictor in using an observed gene set to infer a target gene set, in the absence of observations. In the case of the Pick and Place, component connectivity and influence determine the logical functions that dictate the relationships between nodes. In order to utilize PBNs as the modeling basis, the genes/nodes of the PBN are equated to the basic components of the machine. These components were treated as non-repairable items, which can have one of two

Table 3 Pick and place predictors and regulatory nodes

Component	Predictor	Regulatory nodes
x_1, gripper	$f_1^{(1)} : x_1\,(t+1) = x_1\,(t).$	Gripper
x_2, motor A	$f_1^{(2)} : x_2\,(t+1) = (x_2\,(t)\;AND\,x_6\,(t)).$	Motor A, power supply
	$f_2^{(2)} : x_2\,(t+1) = (x_2\,(t)\;OR\,x_6\,(t)).$	
x_3, motor B	$f_1^{(3)} : x_3\,(t+1) = (x_3\,(t)\;AND\,x_6\,(t)).$	Motor B, power supply
	$f_2^{(3)} : x_3\,(t+1) = (x_3\,(t)\;OR\,x_6\,(t)).$	
x_4, fixed axis	$f_1^{(4)} : x_4\,(t+1) = (x_2\,(t)\;AND\;(x_4\,(t)\;AND\,x_6\,(t))).$	Fixed axis, motor A, power supply
	$f_2^{(4)} : x_4\,(t+1) = (x_2\,(t)\;OR\;(x_4\,(t)\;OR\,x_6\,(t))).$	
x_5, rotary axis	$f_1^{(5)} : x_5\,(t+1) = (x_5\,(t)\;AND\;(x_3\,(t)\;AND\,x_6\,(t))).$	Rotary axis, motor B, power supply
	$f_2^{(5)} : x_5\,(t+1) = (x_5\,(t)\;OR\;(x_3\,(t)\;OR\,x_6\,(t))).$	
x_6, power supply	$f_1^{(6)} : x_6\,(t+1) = x_6\,(t).$	Power supply

values: ‘0’, representing total failure, and ‘1’, representing normal operation.

Evaluating the Pick and Place as a system, only two states are distinguished in the basic components, a functioning state and a failed state. The state of each component $i = 1, 2, \ldots, n$ can be described by a binary variable X_i, where:

$$X_i = \begin{cases} 1 - \text{if the component } i \text{ is functioning} \\ 0 - \text{if the component } i \text{ is in a failed state} \end{cases}$$

The state variables $\mathrm{X}_1(\mathrm{t}), \mathrm{X}_2(\mathrm{t}) \ldots \mathrm{X}_\mathrm{n}(\mathrm{t})$, $n = 6$, are treated as random variables, with a state vector $\mathbf{X}(\mathrm{t}) = (\mathrm{X}_1(\mathrm{t}), \mathrm{X}_2(\mathrm{t}) \ldots \mathrm{X}_\mathrm{n}(\mathrm{t}))$, and a structure function $\phi(\mathbf{X}(\mathrm{t}))$, where $\phi(0) = 0$, and $\phi(1) = 1$, meaning that if all components in the machine are failed, the machine is failed, and if all are functioning, the machine is functioning. Knowing the states of all n components, the state of the machine is also known. Similarly the state of the machine can be described by a logic function $\phi(\mathbf{X}(\mathrm{t})) = \phi(\mathrm{X}_1(\mathrm{t}), \mathrm{X}_2(\mathrm{t}) \ldots \mathrm{X}_\mathrm{n}(\mathrm{t}))$, where:

$$\phi\,(\mathbf{X}\,(\mathrm{t})) = \begin{cases} 1 - \text{if the machine is operational} \\ 0 - \text{if the machine is in a failed state} \end{cases}.$$

Components can follow series, parallel, or k-out-of-n structures (Rausand and Høyland 2004). A system with all of its n components functioning is said to be in a series structure, with a structure function:

$$\phi\,(\mathbf{X}\,(t)) = X_1\,(t) \cdot X_2\,(t) \ldots X_n\,(t) = \prod_{i=1}^{n} X_i\,(t)$$

When establishing the structure of a system, the components that do not play a direct role for the functioning ability of the system are left out. Those that are considered are called *relevant*, and the ones that are left out are called *irrelevant*. A component is said to be irrelevant with respect to a particular system function; however, it may be relevant to another function. In terms of a system of components, it is said to be relevant if all of its components are relevant and its structure is non-decreasing (Rausand and Høyland 2004). The six components that are being modeled for the Pick and Place constitute the relevant components of the machine.

In the case of the Pick and Place, in order for the fixed axis to operate, both motor A and the power supply must also operate. Because of the relationship between the motors, axes and power supplies, the logical function that predicts the state of node X_4 and X_5 can be determined. Therefore, the logical function that correctly expresses this relationship is AND.

Predictors for each node are determined based on node physical connectivity. Instead of using the CoD as a means of determining which nodes/genes are regulating the node that is being analyzed, the physical and logical relations between components determine which nodes are regulatory nodes. Based on this assumption, regulatory nodes and predictors for each node (component) were determined, as shown on Table 3 below. Therefore, relying in an estimation of error as the CoD for the predictors is not required, because the effect that the state of their regulatory nodes have on their state on the next time step it is known based on the physical and logical relationships of the components.

Predictors: selection probabilities estimation

In GRN Analysis using PBNs, the probability $c_i^{(j)}$ can be estimated by using a statistical method, or Coefficient of Determination (Dougherty et al. 2000) with real gene expression data sets. For the proposed approach, in order to determine the selection probability for each predictor, $c_i^{(j)}$, a reliability analysis of the machine is performed in order to assess the frequency of occurrence of the failures of each component and of the machine. Similar to the estimation of error in the CoD, this analysis facilitates the estimation of predictor selection probabilities.

Table 4 Annualized failure rate of all pick and place components

Machine	Component	AFR
Pick & place robot	Gripper	0.442337
	Fixed axis	0.103718
	Motor A	0.021662
	Motor B	0.021662
	Rotary axis	0.103718
	Power supply	0.056727

The reliability of components was calculated in the constant rate of failure period, and it assumed that the probability of failure of each component is independent. Let $R = P\{successful\, component\, operation\} = reliability$, and $Q = P\{unsuccessful\, component\, operation\} = 1 - R$.

Reliability of each component is dependent of time, intended use, and the environmental conditions of usage. Data on Mean Time Between Failure (MTBF) for each of the key components of the modeled machine was obtained from technical data sheets from manufacturers. Based on the MTBF, their failure occurrence was calculated in terms of their Annualized Failure Rate (AFR) using the formula:

$AFR = 1 - \exp(-\frac{8760}{MTBF})$, which is a form of the equation $F(t) = 1 - exp(-\lambda t)$ from Ebeling (1997), where λ is the failure rate of each component, and t is time. Table 4 details the AFR of the Pick and Place components.

Since the state variables (components) are binary, $E[X_i(t)] = 0 \cdot P(X_i(t) = 0) + 1 \cdot P(X_i(t) = 1) = p_i(t), \forall i = 1, 2, \ldots,$ where $p_s(t) = E(\phi(\boldsymbol{X}(t)))$. Here $p_i(t)$ is the probability that component i is functioning at time t, and $p_s(t)$ the probability that the system is functioning at time t. When components are independent, $p_s(t)$ will be a function of $p_i(t)$ only. Therefore, $p_s(t) = h(p_1(t), p_2(t), \ldots, p_n(t)) = h(\boldsymbol{p}(t))$, where h is used to refer to system reliability of independent components exclusively. The structure function of series systems is $\phi(\boldsymbol{X}) = X_1(t) \cdot X_2(t) \ldots X_n(t) = \prod_{i=1}^{n} X_i(t)$, which can be expressed as $R_S(t) = \prod_{i=1}^{n} R_i(t)$. But it is known from Rausand and Høyland (2004) that $R_i(t) = e^{-\int_0^t z_i(u)du}$, where $z_i(t)$ is the failure rate of component i at time t. Inserting $R_i(t)$, $R_i(t) = \prod_{i=1}^{n} e^{-\int_0^t z_i(u)du} = e^{-\int_0^t \sum_{i=1}^{n} z_i(u)du}$, therefore the failure rate $z_s(t)$ of a series structure of independent components is the sum of the failure rate of each of the components, $z_s(t) = \sum_{i=1}^{n} z_i(t)$ and the MTBF of the structure is then MTBF $= \int_0^\infty R_s(t)\, dt = \int_0^\infty e^{-\int_0^t \sum_{i=1}^{n} z_i(u)du} dt$. Since $X_1(t), \ldots, X_n(t)$ are independent, the system reliability is

$$h(\boldsymbol{p}(t)) = E(\phi(X(t))) = E\left(\prod\nolimits_{i=1}^{n} X_i(t)\right)$$
$$= \prod\nolimits_{i=1}^{n} E(X_i(t)) = \prod_{i=1}^{n} p_i(t),$$

where $h(\boldsymbol{p}(t)) \leq \min_i(p_i(t))$, since a series structure is as reliable as its least reliable component.

The Pick and Place machine has six main components for which corresponding AFRs are detailed on Table 4. These calculations were performed based on the MTBF of each of the components. A series system structure is one that is functioning if all of its components are functioning. The Pick and Place machine is a coherent series structure of non-repairable components. The structure function of the Pick and Place, which has six components is: $\phi(\boldsymbol{X}) = \prod_{i=1}^{6} X_i(t) = X_1(t) \cdot X_2(t) \cdot X_3(t) \cdot X_4(t) \cdot X_5(t) \cdot X_6(t)$.

The reliability of the machine is:

$$\begin{aligned} h(\boldsymbol{p}(t)) &= \prod_{i=1}^{6} E(X_i(t)) = \prod_{i=1}^{6} p_i(t) \\ &= p_1(t) \cdot p_2(t) \cdot p_3(t) \cdot p_4(t) \cdot p_5(t) \cdot p_6(t) \\ &= 0.557663 \cdot 0.978338 \cdot 0.978338 \cdot 0.896282 \\ &\quad \cdot 0.896282 \cdot 0.943273 \\ &= 0.404461. \end{aligned}$$

The AFR is $1 - 0.404461 = 0.595539$.

The Pick and Place model can assume two different modes: failure or normal operation. It can be observed that for each mode, there is a unique set of predictors. Their selection probability, as with GRNs, is based on the observations made (analogous to the CoD) of the machine component's interactions and their reliability assessment. When in either mode, the power supply and gripper's next state will only be dependent on their current state. Since there is only a single predictor for both of these nodes, the selection probability is always 1. The rest of the nodes have two predictor functions. The selection probability is based on the reliability calculation. Predictors corresponding to normal operation have a probability of 0.1232 in each node, and those for failure, a 0.8768 probability of occurrence. Table 5 details the predictors of the Pick and Place, along with the selection probabilities for each predictor.

Determining constituent networks and attractors

Based on the predictor functions, a determination can be made of how many realizations, and transitions are part of the Pick and Place's PBN. The Pick and Place is a six gene PBN, where $l(i)$ is the number of possible predictors per node. This means that $l(i), i = 1, \ldots, 6$ is $l(1) = 1, l(2) = 2, l(3) = 2, l(4) = 2, l(5) = 2, l(6) = 1$. The total number of realizations D is: $D = \prod_{i=1}^{6} l(i) = 1 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 1 = 16$. There are 16 possible BNs, characterized by 16 vector functions, listed in Table 6 below.

Table 5 Pick and place predictors and selection probability

Component	Predictor	Selection probability, $c_i^{(j)}$
x_1, gripper	$x_1\,(t+1) = x_1\,(t)$	1.0
x_2, motor A	$x_2\,(t+1) = (x_2\,(t)\,\&\,x_6(t))$	0.8768
	$x_2\,(t+1) = (x_2\,(t)\,\vert x_6(t))$	0.1232
x_3, motor B	$x_3\,(t+1) = (x_3\,(t)\,\&\,x_6(t))$	0.8768
	$x_3\,(t+1) = (x_3\,(t)\,\vert x_6(t))$	0.1232
x_4, fixed axis	$x_4\,(t+1) = (x_2\,(t)\,\&\,(x_4\,(t)\,\&\,x_6(t)))$	0.8768
	$x_4\,(t+1) = (x_2\,(t)\,\vert\,(x_4\,(t)\,\vert x_6(t)))$	0.1232
x_5, rotary axis	$x_5\,(t+1) = (x_5\,(t)\,\&\,(x_3\,(t)\,\&\,x_6(t)))$	0.8768
	$x_5\,(t+1) = (x_5\,(t)\,\vert\,(x_3\,(t)\,\vert x_6(t)))$	0.1232
x_6, power supply	$x_6\,(t+1) = x_6\,(t)$	1.0

Table 6 Pick and place's BN vector functions

Realization	Vector function
1	$f_1 = \left(f_1^{(1)}, f_1^{(2)}, f_1^{(3)}, f_1^{(4)}, f_1^{(5)}, f_1^{(6)}\right)$
2	$f_2 = \left(f_1^{(1)}, f_1^{(2)}, f_1^{(3)}, f_1^{(4)}, f_2^{(5)}, f_1^{(6)}\right)$
3	$f_3 = \left(f_1^{(1)}, f_1^{(2)}, f_1^{(3)}, f_2^{(4)}, f_1^{(5)}, f_1^{(6)}\right)$
4	$f_4 = \left(f_1^{(1)}, f_1^{(2)}, f_1^{(3)}, f_2^{(4)}, f_2^{(5)}, f_1^{(6)}\right)$
5	$f_5 = \left(f_1^{(1)}, f_1^{(2)}, f_2^{(3)}, f_1^{(4)}, f_1^{(5)}, f_1^{(6)}\right)$
6	$f_6 = \left(f_1^{(1)}, f_1^{(2)}, f_2^{(3)}, f_1^{(4)}, f_2^{(5)}, f_1^{(6)}\right)$
7	$f_7 = \left(f_1^{(1)}, f_1^{(2)}, f_2^{(3)}, f_2^{(4)}, f_1^{(5)}, f_1^{(6)}\right)$
8	$f_8 = \left(f_1^{(1)}, f_1^{(2)}, f_2^{(3)}, f_2^{(4)}, f_2^{(5)}, f_1^{(6)}\right)$
9	$f_9 = \left(f_1^{(1)}, f_2^{(2)}, f_1^{(3)}, f_1^{(4)}, f_1^{(5)}, f_1^{(6)}\right)$
10	$f_{10} = \left(f_1^{(1)}, f_2^{(2)}, f_1^{(3)}, f_1^{(4)}, f_2^{(5)}, f_1^{(6)}\right)$
11	$f_{11} = \left(f_1^{(1)}, f_2^{(2)}, f_1^{(3)}, f_2^{(4)}, f_1^{(5)}, f_1^{(6)}\right)$
12	$f_{12} = \left(f_1^{(1)}, f_2^{(2)}, f_1^{(3)}, f_2^{(4)}, f_2^{(5)}, f_1^{(6)}\right)$
13	$f_{13} = \left(f_1^{(1)}, f_2^{(2)}, f_2^{(3)}, f_1^{(4)}, f_1^{(5)}, f_1^{(6)}\right)$
14	$f_{14} = \left(f_1^{(1)}, f_2^{(2)}, f_2^{(3)}, f_1^{(4)}, f_2^{(5)}, f_1^{(6)}\right)$
15	$f_{15} = \left(f_1^{(1)}, f_2^{(2)}, f_2^{(3)}, f_2^{(4)}, f_1^{(5)}, f_1^{(6)}\right)$
16	$f_{16} = \left(f_1^{(1)}, f_2^{(2)}, f_2^{(3)}, f_2^{(4)}, f_2^{(5)}, f_1^{(6)}\right)$

Table 7 Pick and Place: constituent Boolean network selection probabilities

Network	Probability
1	$u_1 = c_1^{(1)} \cdot c_1^{(2)} \cdot c_1^{(3)} \cdot c_1^{(4)} \cdot c_1^{(5)} \cdot c_1^{(6)} = 0.5911$
2	$u_2 = c_1^{(1)} \cdot c_1^{(2)} \cdot c_1^{(3)} \cdot c_1^{(4)} \cdot c_2^{(5)} \cdot c_1^{(6)} = 0.0727$
3	$u_3 = c_1^{(1)} \cdot c_1^{(2)} \cdot c_1^{(3)} \cdot c_2^{(4)} \cdot c_1^{(5)} \cdot c_1^{(6)} = 0.0727$
4	$u_4 = c_1^{(1)} \cdot c_1^{(2)} \cdot c_1^{(3)} \cdot c_2^{(4)} \cdot c_2^{(5)} \cdot c_1^{(6)} = 0.0117$
5	$u_5 = c_1^{(1)} \cdot c_1^{(2)} \cdot c_2^{(3)} \cdot c_1^{(4)} \cdot c_1^{(5)} \cdot c_1^{(6)} = 0.0727$
6	$u_6 = c_1^{(1)} \cdot c_1^{(2)} \cdot c_2^{(3)} \cdot c_1^{(4)} \cdot c_2^{(5)} \cdot c_1^{(6)} = 0.0117$
7	$u_7 = c_1^{(1)} \cdot c_1^{(2)} \cdot c_2^{(3)} \cdot c_2^{(4)} \cdot c_1^{(5)} \cdot c_1^{(6)} = 0.0117$
8	$u_8 = c_1^{(1)} \cdot c_1^{(2)} \cdot c_2^{(3)} \cdot c_2^{(4)} \cdot c_2^{(5)} \cdot c_1^{(6)} = 0.0016$
9	$u_9 = c_1^{(1)} \cdot c_2^{(2)} \cdot c_1^{(3)} \cdot c_1^{(4)} \cdot c_1^{(5)} \cdot c_1^{(6)} = 0.0727$
10	$u_{10} = c_1^{(1)} \cdot c_2^{(2)} \cdot c_1^{(3)} \cdot c_1^{(4)} \cdot c_2^{(5)} \cdot c_1^{(6)} = 0.0117$
11	$u_{11} = c_1^{(1)} \cdot c_2^{(2)} \cdot c_1^{(3)} \cdot c_2^{(4)} \cdot c_1^{(5)} \cdot c_1^{(6)} = 0.0117$
12	$u_{12} = c_1^{(1)} \cdot c_2^{(2)} \cdot c_1^{(3)} \cdot c_2^{(4)} \cdot c_2^{(5)} \cdot c_1^{(6)} = 0.0016$
13	$u_{13} = c_1^{(1)} \cdot c_2^{(2)} \cdot c_2^{(3)} \cdot c_1^{(4)} \cdot c_1^{(5)} \cdot c_1^{(6)} = 0.0117$
14	$u_{14} = c_1^{(1)} \cdot c_2^{(2)} \cdot c_2^{(3)} \cdot c_1^{(4)} \cdot c_2^{(5)} \cdot c_1^{(6)} = 0.0016$
15	$u_{15} = c_1^{(1)} \cdot c_2^{(2)} \cdot c_2^{(3)} \cdot c_2^{(4)} \cdot c_1^{(5)} \cdot c_1^{(6)} = 0.0016$
16	$u_{16} = c_1^{(1)} \cdot c_2^{(2)} \cdot c_2^{(3)} \cdot c_2^{(4)} \cdot c_2^{(5)} \cdot c_1^{(6)} = 0.0002$

The probability of selecting the ith realization that has the vector function is given by $u_i = \prod_{k=1}^{6} c_{i_k}^{(i)}$. Table 7 lists the selection probability for each constituent BN.

It would be impractical to show all the possible transition diagrams for 16 different constituent BNs. Nonetheless, Table 8 presents the attractors of every constituent BN, which are the steady states of the PBN, where the states are listed by their decimal equivalent, with state (000000) = 1, through (111111) = 64, for simplicity. All the listed attractors are singleton attractors, or fixed point. Finding attractors in Boolean Networks is, in itself, an area of active research (Akutsu et al. 2012; Berntensis and Ebeling 2013; Dubrova and Teslenko n.d.; Guo et al. 2014; Hopfensitz et al. 2012; Pal et al. 2006; Qiu et al. 2014; Zheng et al. 2013).

Figure 3 presents the transition diagrams corresponding to one of the 16 realizations of the Pick and Place's PBN. Marked in dar1ker circles are the attractor states of the BN. Described in the figure are all of the possible attractor and non-attractor states of realization 11 of this PBN. States are listed by their decimal equivalent, with state (000000) = 1, through (111111) = 64, for simplicity. Each realization is a collection of transitions. States in these constituent BNs have transitions that lead to attractor states.

A Transition Matrix for the Pick and Place was built. With it, the constituent networks, attractors, and other elements of the PBN's dynamics can be determined. For every state, e.g.

Table 8 Pick and Place's PBN attractors

Realization	Attractors
1	(1, 2, 10, 12, 18, 22, 26, 28, 32, 33, 34, 42, 44, 50, 54, 58, 60, 62, 64)
2	(1, 3, 4, 12, 18, 20, 24, 28, 32, 33, 35, 36, 44, 52, 56, 64)
3	(1, 5, 6, 14, 15, 16, 22, 30, 32, 33, 37, 38, 46, 48, 54, 62, 64)
4	(1, 3, 5, 7, 8, 16, 24, 28, 32, 33, 35, 37, 39, 40, 48, 56, 64)
5	(1, 9, 10, 12, 18, 26, 28, 30, 32, 33, 41, 42, 44, 54, 58, 60, 62, 64)
6	(1, 3, 11, 12, 28, 32, 33, 35, 43, 44, 64)
7	(1, 5, 9, 13, 14, 16, 26, 28, 30, 32, 33, 37, 41, 45, 46, 48, 62, 64)
8	(1, 3, 5, 7, 11, 15, 16, 28, 32, 33, 35, 37, 39, 43, 47, 48, 64)
9	(1, 10, 17, 18, 22, 26, 28, 30, 32, 33, 44, 49, 50, 54, 58, 60, 62, 64)
10	(1, 3, 17, 19, 20, 24, 28, 32, 33, 35, 49, 51, 52, 56, 60, 64,)
11	(1, 5, 21, 22, 30, 32, 33, 37, 53, 54, 62, 64)
12	(1, 3, 5, 7, 21, 23, 24, 32, 33, 35, 37, 53, 55, 56, 64)
13	(1, 9, 17, 25, 26, 28, 30, 32, 33, 41, 49, 57, 58, 60, 62, 64)
14	(1, 3, 11, 17, 19, 27, 28, 32, 33, 35, 43, 49, 51, 59, 60, 64)
15	(1, 5, 9, 13, 14, 21, 29, 30, 32, 37, 41, 45, 53, 61, 62, 64)
16	(1, 3, 5, 7, 11, 15, 21, 23, 31, 32, 33, 35, 37, 39, 43, 47, 53, 55, 63, 64)

$(x_1 = x_2 = x_3 = x_4 = x_5 = x_6 = 0)$, or (000000), the predictors are applied, based on the selection probabilities, to obtain the state on the next time step in order to construct a transition diagram.

Attractors play a seminal role in Boolean networks, since given an initial state, within a finite number of time steps, the network will transition into a state or cycle of states, and if there are no perturbations on the network, it will continue to cycle thereafter. States that are not attractor states are called transient because these are visited at most once. Attractors characterize the steady-state behavior of BN's and PBNs.

Experimental results

This section presents results of experiments designed to test the adequacy of the proposed model. According to Dougherty (2011), model validation relies on its ability to draw predictions that can be checked against experimental observations, requiring that a relationship be established between model characteristics and observables. Moreover, the model must allow performing experimental predictions that can be related to observable phenomena so there is a correspondence between experimental and predicted values. In order to quantitatively validate the proposed model, PRISM Model Checker (Kwiatkowska et al. 2011) was used to generate the data required to conduct inferential statistical tests to determine the level of correspondence.

A control group was established, by simulating the Pick & Place's relevant components with their corresponding MTBFs, obtained from actual technical data sheets. This control group represents expected values. Two experimental groups were also established: (1) the PBN model of the system with non-deterministic control inputs (the proposed model); and (2) the PBN model of the system with the components' MTBFs as control inputs. Property verification in PRISM was used to determine the maximum probability that at least one of the Pick and Place machine components fails through verification of the following property:

$$\begin{aligned}&Pmax =?[F <= time\,(grip1 = false)\\&\mid (power1 = false) \mid (motor1A = false)\\&\mid (motor1B = false) \mid (rotary\_axis1 = false)\\&\mid (fixed\_axis1 = false)].\end{aligned}$$

The property verifies what is the maximum probability that in the future, when time reaches a certain value, either of the components in the machine is in failure. The results are plotted and shown on Figure 4 below.

A one-way ANOVA was performed using Minitab 16 to verify if there were statistically significant differences among the group means. The null hypothesis for the test is that all population means (group means) are the same or $H_o : \mu_{control} = \mu_{E1} = \mu_{E2}$. The alternative hypothesis is that one or more population means differ from the others or $H_1 : \mu_{control} \neq \mu_{E1} \neq \mu_{E2}$. The p value for the ANOVA is 0.208. Given α-level of 0.05 for the test, it is concluded that there are no significant differences in probabilities of failure between the groups. In practical terms, there is no difference between expected and observed values. Results of this test are shown on Fig. 5.

Also, differences between the control group and each of the two experimental groups were determined. A two-sample t-confidence interval and test procedure was used to make inferences about the difference between two population means based on data from two independent, random samples. The null hypothesis is that there is no difference between the delta in Control and Experimental Group 1 and the delta in Control and Experimental Group 2, or Ho :

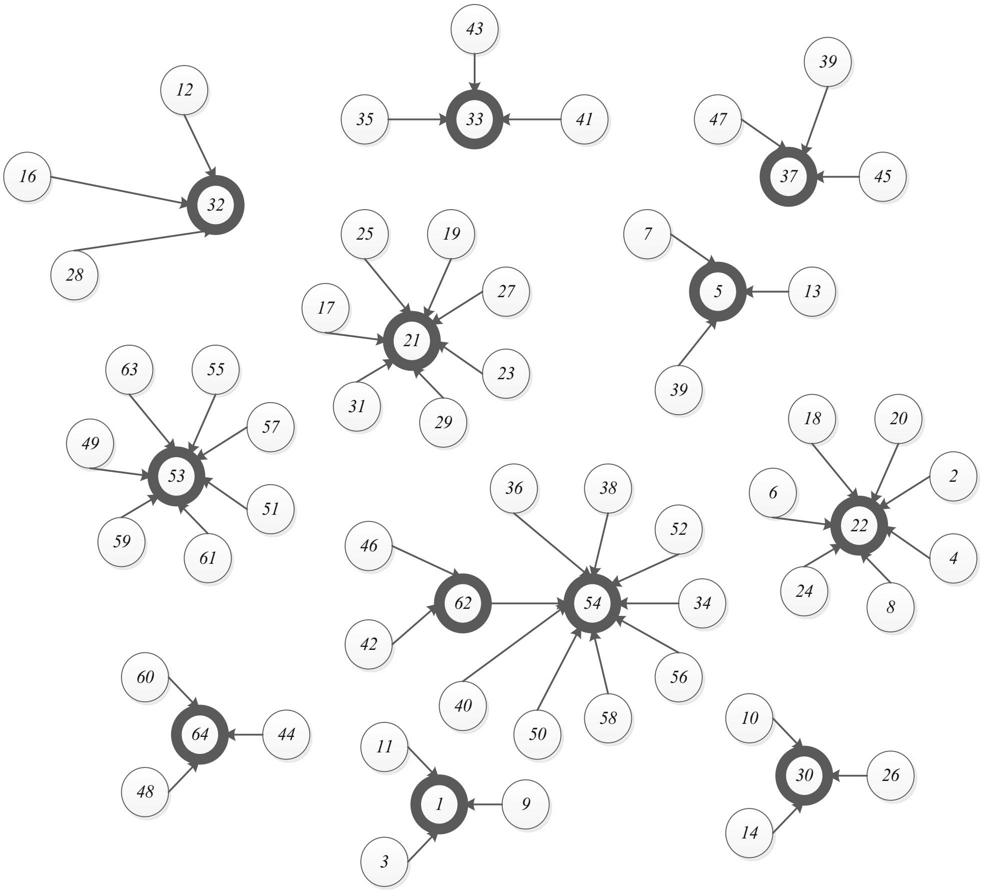


Fig. 3 Transition diagram of one of the constituent BN of the Pick and Place PBN

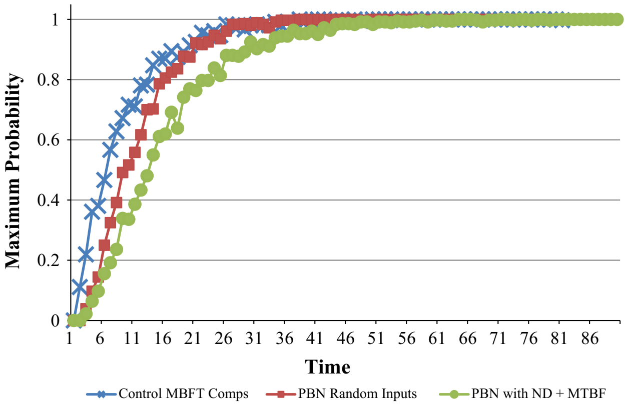
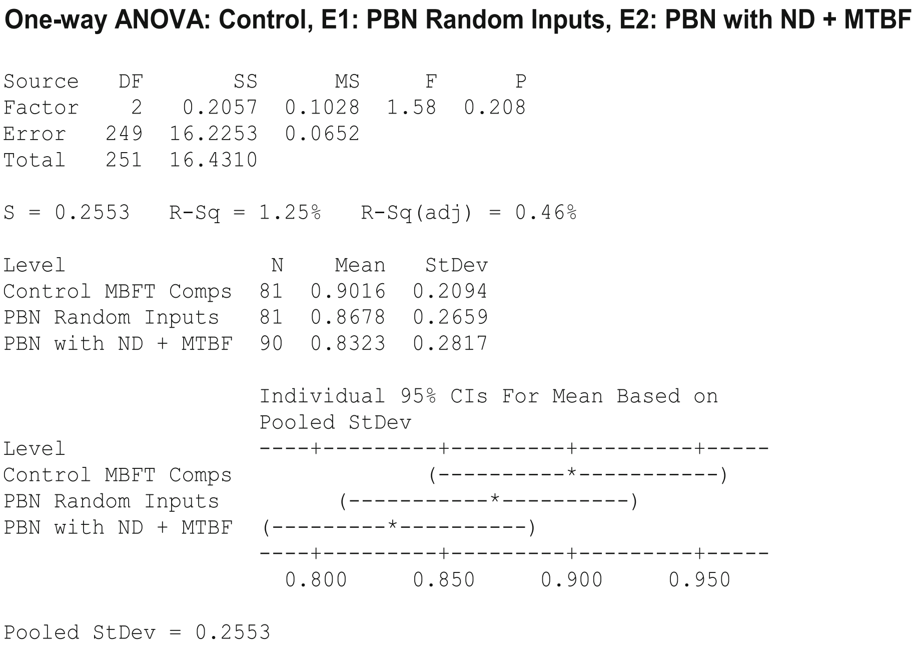


Fig. 4 Maximum probability of failure of at least one component of the Pick and place. Control and Experimental groups

$\mu\, dif\ control - E1 = \mu\, dif\ control - E2$. The alternative hypothesis is that difference between the delta in Control and Experimental Group 1 is less than the delta in Control and Experimental Group 2, or $Ho\ :\ \mu\, difference\ control - E1\ <\ \mu\, difference\ control - E2$. The p value for the hypothesis test is 0.000 and given α-level of 0.05, the null hypothesis is rejected. Therefore, the delta observed in Control and the Experimental Group 1 (the proposed model) is statistically significantly less than the one observed between Control and the Experimental Group 2. The proposed model provides values closer to the expected values; consequently, it can adequately model observable phenomena. Results of the two-sample T test are shown in Fig. 6.

Figure 7 shows a two-sample T test performed to determine differences between the control group and the proposed model. This serves to further demonstrate that there is no statistically significant difference between the control group and the proposed model, since the p value is larger than alpha.

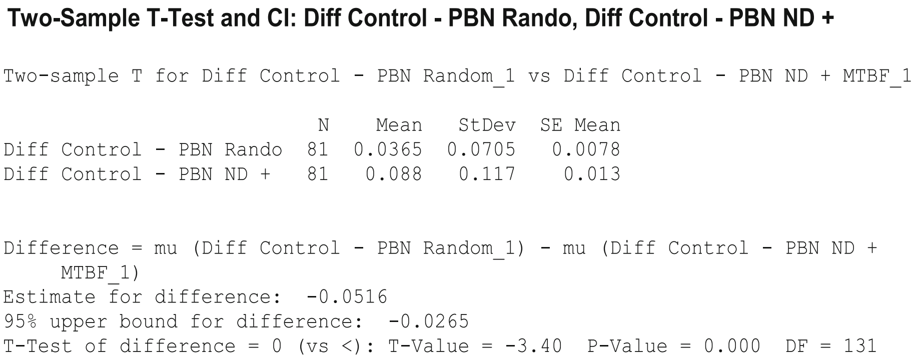
When modeling and simulating known manufacturing machines characterized as PBNs, it is expected that the model performs very close to what the machine would actually be in a production environment. Similar to other bio-inspired modeling techniques already discussed in the second section, PBNs mimic behavior exhibited in nature and provide certain advantages. PBN modeling is a methodology based on the comprehension of the logical relationships between nodes that yields state-based models that have predictive behavior. The main advantages of PBN modeling, compared to other bio-mimetic modeling methodologies, are that PBN models are straightforward to construct, provide a mechanism



One-way ANOVA: Control, E1: PBN Random Inputs, E2: PBN with ND + MTBF

```
Source   DF       SS      MS     F      P
Factor    2   0.2057  0.1028  1.58  0.208
Error   249  16.2253  0.0652
Total   251  16.4310

S = 0.2553   R-Sq = 1.25%   R-Sq(adj) = 0.46%

Level                N    Mean   StDev
Control MBFT Comps  81  0.9016  0.2094
PBN Random Inputs   81  0.8678  0.2659
PBN with ND + MTBF  90  0.8323  0.2817

                    Individual 95% CIs For Mean Based on
                    Pooled StDev
Level               ----+---------+---------+---------+-----
Control MBFT Comps              (----------*-----------)
PBN Random Inputs          (-----------*----------)
PBN with ND + MTBF  (---------*----------)
                    ----+---------+---------+---------+-----
                      0.800     0.850     0.900     0.950

Pooled StDev = 0.2553
```

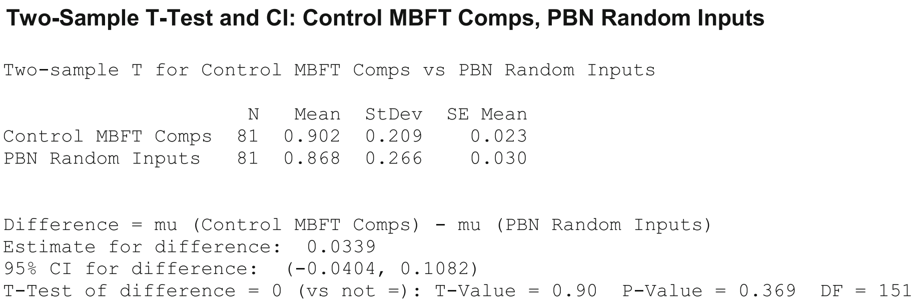
Fig. 5 One-way ANOVA: control versus experimental groups



Two-Sample T-Test and CI: Diff Control - PBN Rando, Diff Control - PBN ND +

```
Two-sample T for Diff Control - PBN Random_1 vs Diff Control - PBN ND + MTBF_1

                        N    Mean   StDev  SE Mean
Diff Control - PBN Rando  81  0.0365  0.0705   0.0078
Diff Control - PBN ND +   81   0.088   0.117    0.013


Difference = mu (Diff Control - PBN Random_1) - mu (Diff Control - PBN ND +
     MTBF_1)
Estimate for difference:  -0.0516
95% upper bound for difference:  -0.0265
T-Test of difference = 0 (vs <): T-Value = -3.40  P-Value = 0.000  DF = 131
```

Fig. 6 Two-sample *T* test: delta between experimental groups and control



Two-Sample T-Test and CI: Control MBFT Comps, PBN Random Inputs

```
Two-sample T for Control MBFT Comps vs PBN Random Inputs

                     N   Mean  StDev  SE Mean
Control MBFT Comps  81  0.902  0.209    0.023
PBN Random Inputs   81  0.868  0.266    0.030


Difference = mu (Control MBFT Comps) - mu (PBN Random Inputs)
Estimate for difference:  0.0339
95% CI for difference:  (-0.0404, 0.1082)
T-Test of difference = 0 (vs not =): T-Value = 0.90  P-Value = 0.369  DF = 151
```

Fig. 7 Two-sample *T* test: control and proposed model

of predicting the behavior of a system, and produces accurate results. Through the results obtained, it is statistically demonstrated that when modeling and simulating a known manufacturing machine characterized as PBN, the model performs very close to what the machine would actually be in a production environment. Therefore, PBN modeling, along with model checking and simulation, facilitates experimenting with or examining a machine's possible behavior without having to use it directly. The added value of this research, compared to other methods, is that with PBN modeling the designer or engineer can have a semantically correct model that is simple to define that allows to observe the evolution of the system and predict its future state. Given that the model capability to reproduce results very close to expected actual values, complexity is reduced and allows the designer to make accurate design decisions. Model Checking permits formal verification of the model and its state sequence/evolution. Besides being a model definition easy to create, formal verification provides the assurance that it is computationally/mathematically correct model.

According to Ueda et al. (1997), BMS deal with dynamic changes in external and internal environments using biologically inspired ideas, such as self-growth, self-organization, adaptation and evolution. As other BMS, systems modeled using PBNs are able to evolve through time, and its evolution through time is oriented towards its steady-states or attractors. PBNs are able to adapt to noise or fluctuations, and knowledge of those mechanisms that govern them provide the means to control their behavior. Guiding network dynamics in such a way is called intervention. Similar to other bio-inspired methodologies, such as swarms, ant colonies and multi-agents, PBNs exhibit emergent characteristics, such as self-organization, and that interact with other individual entities in the system. In manufacturing systems, swarm intelligence technology can be expressed as evolutionary algorithms, as in Anghinolfi et al. (2007), Moon et al. (2014) and Zainal et al. (2014), or as multi-agent systems (Leitão and Restivo 2002; Sahin et al. 2015; Barbosa and Leitão 2011; Xiong and Fu 2015).

Guided Self-Organization (Prokopenko 2009) steers the self-organizing dynamics of a system to a favored configuration, balancing design and self-organization. PBNs exhibit self-organizing characteristics (Kauffman 1993). BN and PBN dynamics self-organize towards attractors, be it attractor cycles or point attractors, and they reduce system complexity. As an example, there are over 30,000 genes (nodes) in the human genome, and only around 300 cell types, or attractors, cells that self-organize towards a limited sub-set of possible states (Kauffman 1993). When a system has a group of preferred states, or attractors, the system will self-organize toward them. When two levels of representation are present and there is a relationship or interaction between these, the system can be self-organizing and the interactions of the lower level change the properties of the higher level (e.g. bee-swarm, ant-colony, gene-cell). In PBNs the mechanism of intervention can be used as a guided self-organizing technique that steers the evolution of the system towards a desired operational or chosen state. The criticality (balance between ordered and chaotic behavior) of a system or network depends on many factors, and these can be advantageous to engineers and designers to steer the evolution of the system. The evolution of an industrial machine, like the Pick and Place, can be guided towards a preferred state (normal operation), and the machine's evolution in time will oscillate (criticality) between ordered dynamics (the operating state of the machine) and chaos (states leading to machine failure). Every time a component fails and steers the system into chaotic behavior, the system can eventually organize and correct its behavior to reach a preferred state. The topics of perturbation and intervention in the systems of interest will be addressed in future research articles.

Conclusions

This paper presented a bio-inspired, stochastic modeling methodology for an industrial machine using probabilistic Boolean networks. The methodology aids the development and validation of a bio-inspired model from which statistically valid predictions about its behavior were obtained. A Pick and Place machine was modeled using the proposed approach and, coupled with validation and verification, experiments were conducted to perform empirical predictions of system behaviors. Experimental data showed results that were congruent with expected observable events. This research demonstrates that PBN modeling of industrial machines is appropriate because of the similarities in characteristics between both: probabilistic, dynamic systems, with rule-based state-transition behavior. Moreover, this research pioneers the application of PBNs outside the well-studied area of GRNs, thus serving as a basis for future research on PBN modeling in industrial processes.

Findings from this research suggest that the proposed approach can be repeated for multiple machines in a major system, which can afterwards be characterized as a PBN. The predictors for the system can be determined in the same way, studying the relationship between the nodes to determine relevant nodes, logical equations, among others. The predictor selection probability is also determined from a probability analysis. Future research can also be focused on "interventions", or deliberate perturbation of a network to achieve a desired response, in order to identify those conditions that can attract "healthy" machine states, thus improving its reliability and efficiency.

References

Akutsu, T., Kosub, S., Melkman, A., & Tamura, T. (2012). Finding a periodic attractor of a Boolean network. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *9*(5), 1410–1421.

Anghinolfi, D., Boccalante, A., Grosso, A., Paolucci, M., Passadore, A., & Vecchiola, C. (2007). A swarm intelligence method applied to manufacturing scheduling. In *Proceedings of the WOA* (pp. 65–70).

Arnosti, D. N., & Ay, A. (2012). Boolean modeling of gene regulatory networks: Driesch redux. *Proceedings of the National Academy of Sciences*, *109*(45), 18239–18240.

Assef, Y., Bastard, P., & Meunier, M. (1996). Artificial neural networks for single phase fault detection in resonant grounded power distribution sytems. In *Proceedings of the 1996 transmission and distribution conference*. http://ieeexplore.ieee.org.ezproxy.library.wisc.edu/xpls/abs_all.jsp?arnumber=547573.

Ayhan, M. B., Aydin, M. E., & Öztemel, E. (2013). A multi-agent based approach for change management in manufacturing enterprises. *Journal of Intelligent Manufacturing*. doi:10.1007/s10845-013-0794-2.

Babiceanu, R. F., & Chen, F. F. (2006). Development and applications of holonic manufacturing systems: A survey. *Journal of Intelligent Manufacturing*, *17*(1), 111–131.

Bane, V., Ravanmehr, V., & Krishnan, A. R. (2012). An information theoretic approach to constructing robust Boolean gene regulatory networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *9*(1), 52–65.

Barbosa, J. & Leitão, P. (2011). Simulation of multi-agent manufacturing systems using agent-based modelling platforms. In *Proceedings of the 9th IEEE international conference on industrial informatics (INDIN), Lisbon, Portugal* (pp. 477–482).

Barghash, M. A., & Santarisi, N. S. (2004). Pattern recognition of control charts using artificial neural networks—Analyzing the effect of the training parameters. *Journal of Intelligent Manufacturing*, *15*(5), 635–644.

Beaudin, M. (1990). *Manufacturing systems analysis*. Englewood Cliffs, NJ: Yourdon Press.

Berntensis, N., & Ebeling, M. (2013). Detection of attractors of large Boolean networks via exhaustive enumeration of appropriate subspaces of the state space. *BMC Bioinformatics*, *14*, 361.

Booker, L., Goldberg, D., & Holland, J. H. (1989). Clasifier systems and genetic algorithms. *Artificial Intelligence*, *40*(1–3), 235–282.

Chaib-Draa, B., Moulin, B., Mandiau, R., & Millot, P. (1992). Trends in distributed artificial intelligence. *Artificial Intelligence Review*, *6*, 35–66.

Chaouiya, C., Ourrad, O., & Lima, R. (2013). Majority rules with random tie-breaking in Boolean gene regulatory networks. *PLoS ONE*, *8*(7), e69626.

Chen, H., & Sun, J. (2014). Stability and stabilisation of context-sensitive probabilistic Boolean networks. *IET Control Theory & Applications*, *8*(17), 2115–2121.

Chen, X., Jiang, H., & Ching, W.-K. (2012). On construction of sparse probabilistic Boolean Networks. *East Asian Journal on Applied Mathematics*. doi:10.4208/eajam.030511.060911a.

Cheng, X., Sun, M., & Socolar, J. E. S. (2013). Autonomous Boolean modelling of developmental gene regulatory networks. *Journal of the Royal Society Interface*, *10*(78), 20120574.

Ching, W.-K., Chen, X., & Tsing, N.-K. (2009). Generating probabilistic Boolean networks from a prescribed transition probability matrix. *IET Systems Biology*, *3*, 453–464.

Ching, W.-K., Zhang, S.-Q., Jiao, Y., Akutsu, T., Tsing, N.-K., & Wong, A.-S. (2009). Optimal control policy for probabilistic Boolean networks with hard constraints. *IET Systems Biology*, *3*(2), 90–99.

Cicirello, V., & Smith, S. (2001a). Improved routing wasps for distributed factory control. In *Proceedings of the workshop on artificial intelligence and manufacturing*. Presented at the workshop on artificial intelligence and manufacturing.

Cicirello, V., & Smith, S. (2001b). Wasp nests for self-configurable factories. In *Proceedings of the 5th international conference on autonomous agents*.

Corry, P., & Kozan, E. (2004). Ant colony optimisation for machine kayout problems. *Computational Optimization and Applications*, *28*(3), 287–310.

Didier, G., & Remy, E. (2012). Relations between gene regulatory networks and cell dynamics in Boolean models. *Discrete Applied Mathematics*, *160*(15), 2147–2157.

Dorigo, M. (1992). *Optimization, learning, and natural algorithms* (Doctoral Thesis). Politecnico di Milano, Milan, Italy.

Dorigo, M., & Blum, C. (2005). Ant colony optimization theory: A survey. *Theoretical Computer Science*, *344*(2–3), 243–278.

Dougherty, E. R. (2011). Validation of gene regulatory networks: Scientific and inferential. *Briefings in Bioinformatics*, *12*(3), 245–252.

Dougherty, E. R., Kim, S., & Chen, Y. (2000). Coefficient of determination in nonlinear signal processing. *Signal Processing*, *80*, 2219–2235.

Dubrova, E., & Teslenko, M. (2011). A SAT-based algorithm for finding attractors in synchronous Boolean networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *8*(5), 1393–1399. doi:10.1109/TCBB.2010.20.

Ebeling, C. E. (1997). *An introduction to reliability and maintainability engineering*. New York: McGraw-Hill.

Gao, Y., Xu, P., Wang, X., & Liu, W. (2013). The complex fluctuations of probabilistic Boolean networks. *BioSystems*, *114*(1), 78–84.

Gershenson, C. (2007). *Design and control of self-organizing systems*. CopIt Arxives, Mexico. http://tinyurl.com/DCSOS2007.

Ghanbarnejad, F. (2012). *Perturbations in Boolean networks as model of gene regulatory dynamics (doctoral thesis)*. Leipzig: University of Leipzig.

Gu, J.-W., Ching, W.-K., Siu, T.-K., & Zheng, H. (2013). On modeling credit defaults: A probabilistic Boolean network approach. *Risk and Decision Analysis*, *4*(2), 119–129.

Guo, W., Yang, G., Wu, W., He, L., & Sun, M. (2014). A parallel attractor finding algorithm based on Boolean satisfiability for genetic regulatory networks. PLoS One .

Hopfensitz, M., Müssel, C., & Maucher, M. (2012). Attractors in Boolean networks: a tutorial. *Computational Statistics*. http://www.springerlink.com.ezproxy.library.wisc.edu/index/NR1671N55Q3365Q5.pdf

Hsieh, F.-S., & Lin, J.-B. (2013). A self-adaptation scheme for workflow management in multi-agent systems. *Journal of Intelligent Manufacturing*.

Hsieh, F.-S., & Lin, J.-B. (2014). Context-aware workflow management for virtual enterprises based on coordination of agents. *Journal of Intelligent Manufacturing*, *25*(3), 393–412.

Huang, Y., McMurran, R., Dhadyalla, G., & Jones, R. P. (2008). Probability based vehicle fault diagnosis: Bayesian network method. *Journal of Intelligent Manufacturing*, *19*(3), 301–311.

Jamhour, A., & García, C. (2012). Automation of industrial serial processes based on finite state machines. *Presented at the 20th international congress of chemical and process engineering*, Prague, Czech Republic.

Kauffman, S. A. (1969a). Homeostasis and differentitation in random genetic control networks. *Nature*, *224*, 177–178.

Kauffman, S. A. (1969b). Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, *22*, 437–467.

Kauffman, S. A. (1993). *The origins of order: Self-organization and selection in evolution*. NewYork: Oxford University Press.

Kobayashi, K., & Hiraishi, K. (2010). Reachability analysis of probabilistic Boolean networks using model checking. *Presented at the SICE annual conference 2010, proceedings of* (pp. 829–832). http://library.uprm.edu:2055/stamp/stamp.jsp?tp=&arnumber=5604207.

Koestler, A. (1967). *The ghost in the machine*. New York: Macmillan.

Kumar, A., & Dhingra, A. K. (2012). Optimization of scheduling problems: A genetic algorithm survey. *International Journal of Applied Science and Engineering Research*, *1*(1), 11–25.

Kwiatkowska, M. Z., Norman, G., & Parker, D. (2011). PRISM 4.0: Verification of probabilistic real-time systems. In G. Gopalakrishnan & S. Qadeer (Eds.), *Computer Aided Verification*. Lecture Notes in Computer Science (Vol. 6806, pp. 585–591). Berlin, Heidelberg: Springer.

Leger, R., Garland, W., & Poehlman, W. F. S. (1998). Fault detection and diagnosis using statistical control charts and artificial neural networks. *Artificial Intelligence in Engineering*. http://www.sciencedirect.com.ezproxy.library.wisc.edu/science/article/pii/S0954181096000398.

Leitão, P. (2008). Self-organization in manufacturing systems: Challenges and opportunities. In *Procedings of the second IEEE international conference on self-adaptive and self-organizing systems workshops* (pp. 174–179). Presented at the 2nd IEEE international conference on self-adaptive and self-organizing systems.

Leitão, P., & Restivo, F. (2002). Agent-based holonic production control. In *Proceedings of the 13th international workshop on database and expert systems applications* (pp. 589–596).

Li, P., Zhang, C., Perkins, E. J., Gong, P., & Deng, Y. (2007). Comparison of probabilistic Boolean network and dynamic Bayesian network approaches for inferring gene regulatory networks. *BMC Bioinformatics*, *8*(13).

Liang, R., Qiu, Y., & Ching, W.-K. (2014). Construction of Probabilistic Boolean Network for Credit Default Data. In *Proceedings of the seventh international joint conference on computational science and optimization*. Presented at the Seventh International Joint Conference on Computational Science and Optimization.

Liu, X., Yi, H., & Zhong-hua, N. (2013). Application of ant colony optimization algorithm in process planning optimization. *Journal of Intelligent Manufacturing*, *24*(1), 1–13.

Liu, Y., Ling, X., Zhewen, S., Mingwei, L., Fang, J., & Zhang, L. (2011). A survey on particle swarm optimization algorithms for multimodal function optimization. *Journal of Software*, *6*(12), 2449.

Markov, A. A. (1954). The theory of algorithms. *Academy of Sciences of the USSR*, *42*, 3–375.

Moon, I., Lee, S., Shin, M., & Ryu, K. (2014). Evolutionary resource assignment for workload-based production scheduling. *Journal of Intelligent Manufacturing*.

Moore, K., & Gupta, S. M. (1996). Petri net models of flexible and automated manufacturing systems: A survey. *International Journal of Production Research*, *34*(11), 3001–3035.

Mosallam, A., Medjaher, K., & Zerhouni, N. (2014). Data-driven prognostic method based on Bayesian approaches for direct remaining useful life prediction. *Journal of Intelligent Manufacturing*. doi:10.1007/s10845-014-0933-4.

Nouiri, M., Bekrar, A., Jemai, A., Niar, S., & Ammari, A. C. (2015). An effective and distributed particle swarm optimization algorithm for flexible job-shop scheduling problem. *Journal of Intelligent Manufacturing*. doi:10.1007/s10845-015-1039-3.

Pal, R., Ivanov, I., Datta, A., Bittner, M. L., & Dougherty, E. R. (2006). Synthesizing Boolean networks with a given attractor structure. Genomic Signal Processing and Statistics, 2006. GENSIPS'06. *IEEE International Workshop on*, 73–74, doi:10.1109/GENSIPS.2006.353162.

Park, H.-S., & Tran, N.-H. (2010). An intelligent manufacturing system with biological principles. *International Journal of CAD/CAM*, *10*(1), 39–50.

Prokopenko, M. (2009). Guided self-organization. *HFSP Journal*, *3*(5), 287–289.

Qiu, Y., Tamura, T., Ching, W.-K., & Akutsu, T. (2014). On control of singleton attractors in multiple Boolean networks: Integer programming-based method. *BMC Systems Biology*, *8*, S7.

Rausand, M., & Høyland, A. (2004). *Systems reliability theory: Models, statistical methods, and applications* (2nd ed.). Hoboken, NJ: Wiley.

Ristevski, B. (2013). A survey of models for inference of gene regulatory networks. *Nonlinear Analysis-Modeling and Control*, *18*(24), 444–465.

Sahin, C., Demitras, M., Erol, R., Baykasoğlu, A., & Kaplanoğlu, V. (2015). A multi-agent based approach to dynamic scheduling with flexible processing capabilities. *Journal of Intelligent Manufacturing*. doi:10.1007/s10845-015-1069-x.

Samanta, B., & Nataraj, C. (2009). Application of particle swarm optimization and proximal support vector machines for fault detection. *Swarm Intelligence*, *3*(4), 303–325.

Shmulevich, I., Dougherty, E., & Kim, S. (2002). Probabilistic Boolean networks: A rule-based uncertainty model for gene regulatory networks. *Bioinformatics*. http://bioinformatics.oxfordjournals.org.ezproxy.library.wisc.edu/content/18/2/261.short

Shmulevich, I., & Dougherty, E. R. (2010). *Probabilistic boolean networks: Modeling and control of gene regulatory networks*. Philadelphia, PA: SIAM.

Shmulevich, I., Dougherty, E. R., Kim, S., & Zhang, W. (2002). From Boolean to probabilistic Boolean networks as models of genetic regulatory networks. *Proceedings of the IEEE*, *90*, 1778–1792.

Skitt, P. J. C., Javed, M. A., Sanders, S. A., & Higginson, A. M. (1993). Process monitoring using auto-associative, feed-forward artificial neural networks. *Journal of Intelligent Manufacturing*, *4*(1), 79–94.

Sood, A. (2013). Artificial neural networks-growth & learn: A survey. *International Journal of Soft Computing and Engineering*, *3*(3), 103–104.

Sun, T.-H., Tien, F.-C., Tien, F.-C., & Kuo, R.-J. (2014). Automated thermal fuse inspection using machine vision and artificial neural networks. *Journal of Intelligent Manufacturing*. doi:10.1007/s10845-014-0902-y.

Tchangani, A. P. (2004). Decision-making with uncertain data: Bayesian linear programming approach. *Journal of Intelligent Manufacturing*, *15*(1), 17–27.

Trairatphisan, P., Mizera, A., Pang, J., Tantar, A. A., Schneider, J., & Sauter, T. (2013). Recent development and biomedical applications of probabilistic Boolean networks. *Cell Communication and Signaling*, *11*, 46.

Ueda, K. (1992). A concept for bionic manufacturing systems based on DNA-type information. In *Proceedings of the 8th international PROLAMAT conference* (pp. 853–863).

Ueda, K. (1993). A genetic approach toward future manufacturing systems. In J. Peklenik (Ed.), *Flexible manufacturing systems: Past–present–future* (p. 211). Ljubljana, Slovenia: CIRP.

Ueda, K. (1994). *Biological manufacturing systems*. Tokyo: Kogyochosakai.

Ueda, K., Vaario, J., & Ohkura, K. (1997). Modelling of biological manufacturing systems for dynamic reconfiguration. *Annals of the CIRP*, *46*(1), 343–346.

Vahedi, G. (2009). *An engineering approach towards personalized cancer therapy*. Retrieved from http://gradworks.umi.com.ezproxy.library.wisc.edu/33/84/3384337.html

Wang, X., Wang, H., & Qi, C. (2014). Multi-agent reinforcement learning based maintenance policy for a resource constrained flow line system. *Journal of Intelligent Manufacturing*. doi:10.1007/s10845-013-0864-5.

Wooldridge, M. (2002). *An introduction to multi-agent systems*. New York: Wiley.

Wu, C.-H., Wang, D.-Z., Ip, A., Wang, D.-W., Chan, C.-Y., & Wang, H.-F. (2009). A particle swarm optimization approach for components placement inspection on printed circuit boards. *Journal of Intelligent Manufacturing*, *20*(5), 535–549.

Xiong, W., & Fu, D. (2015). A new immune multi-agent system for the flexible shop scheduling problem. *Journal of Intelligent Manufacturing*. doi:10.1007/s10845-015-1137-2.

Zainal, N., Zain, A. M., Razi, N. H. M., & Othman, M. R. (2014). Glowworm swarm optimization (GSO) for optimization of machining parameters. *Journal of Intelligent Manufacturing*. doi:10.1007/s10845-014-0914-7.

Zheng, D., Yang, G., Li, X., Wang, Z., Liu, F., & He, L. (2013). An efficient algorithm for computing attractors of sychronous and asynchronous Boolean networks. *PLoS One, 8*(4), e60593. doi:10.1371/journal.pone.0060593.