Harju, Mikko; Virtanen, Kai; Poropudas, Jirka

Simulation Metamodeling Using Dynamic Bayesian Networks with Multiple Time Scales

Published in:
Proceedings of The 9th EUROSIM Congress on Modelling and Simulation (EUROSIM 2016), The 57th SIMS Conference on Simulation and Modelling (SIMS 2016)

Published: 19/12/2018

Document Version
Publisher's PDF, also known as Version of record

Please cite the original version:
Harju, M., Virtanen, K., & Poropudas, J. (2018). Simulation Metamodeling Using Dynamic Bayesian Networks with Multiple Time Scales. In E. Juuso, E. Dahlquist, & K. Leiviskä (Eds.), *Proceedings of The 9th EUROSIM Congress on Modelling and Simulation (EUROSIM 2016), The 57th SIMS Conference on Simulation and Modelling (SIMS 2016)* (pp. 619-625). Article 090 (Linköping electronic conference proceedings; No. 142). Linköping University Electronic Press.

Simulation Metamodeling using Dynamic Bayesian Networks with Multiple Time Scales

Mikko Harju Kai Virtanen Jirka Poropudas

Department of Mathematics and Systems Analysis, Aalto University School of Science, Finland,
mikko.harju@aalto.fi, kai.virtanen@aalto.fi, jirka.poropudas@aalto.fi

Abstract

The utilization of dynamic Bayesian networks (DBNs) in simulation metamodeling enables the investigation of the time evolution of state variables of a simulation model. DBN metamodels have previously described the changes in the probability distribution of the simulation state by using a time slice structure in which the state variables are described at common time instants. In this paper, the novel approach to the determination of the time slice structure is introduced. It enables the selection of time instants of the DBN separately for each state variable. In this way, a more accurate metamodel representing multiple time scales of the variables is achieved. Furthermore, the construction is streamlined by presenting a dynamic programming algorithm for determining the key time instants for individual variables. The construction and use of the DBN metamodels are illustrated by an example problem dealing with the simulated operation of an air base.

Keywords: Bayesian networks, discrete event simulation, dynamic Bayesian networks, simulation, simulation metamodeling

1 Introduction

Discrete event simulation (DES) (e.g. Law and Kelton 2000) is a widely used methodology for modeling and analyzing stochastic dynamic systems. A DES model describes a system consisting of three types of variables (Zeigler et al., 2000). The values of input variables are given prior to the simulation and can be, e.g., parameters that determine the configuration of the system. Time variant state variables describe the time evolution of the system. Output variables obtain values after the simulation is completed and correspond to the characteristics of the system that are being investigated, such as the average waiting time in a queueing system. The main interest in the analysis of DES models is often on the relation between the input and output variables. Simulation metamodels (Friedman, 2012; Kleijnen, 2008) have been used in order to efficiently describe this relationship. See (Poropudas, 2011) for an overview of different types of metamodels as well as details of the construction and utilization of such models.

To better understand how the simulation progresses, it may be of interest to investigate the time development of the state variables of a DES model – instead of the dependence between inputs and outputs. With most simulation metamodeling techniques, such as regression modeling and stochastic Kriging (Kleijnen, 2008), it is not possible to include the state variables into the metamodel. However, the time evolution of the state variables can be analyzed by using dynamic Bayesian networks (DBNs, see, e.g., Murphy 2002) as simulation metamodels (Poropudas and Virtanen, 2007, 2011). Bayesian networks (BNs, see, e.g., Pearl 1986) are probabilistic models that describe the joint probability distribution of discrete random variables. A BN consists of a directed acyclic graph with nodes corresponding to variables and arcs indicating the dependencies between the variables. In addition, a conditional probability table (CPT) is associated with each node, describing its probability distributions conditional on the values of its parent nodes. In a DBN, individual variables are represented by multiple nodes that correspond to their value at specific time instants. In simulation metamodeling, the nodes of a DBN correspond to input, output, and state variables of a DES model. Thus, the DBN metamodel provides a representation for the joint probability distribution of the input, output, and state variables of the simulation where the state variables are considered at some fixed time instants. The DBNs are used to efficiently calculate marginal and conditional probability distributions of the state variables. The construction and utilization of the DBN metamodels are aided by available BN software (e.g., Decision Systems Laboratory). By using interpolation between the time instants of the DBN, the probability distributions are approximated for any time instants within the duration of the simulation (Poropudas and Virtanen, 2010).

The nodes of DBNs are partitioned into sets corresponding to particular time instants. The sets are called time slices and, typically, all the time slices include nodes corresponding to each of the variables. In the context of DBN metamodels, this means that all state variables are considered at the same time instants. The common time instants are not necessarily ideal because they may ignore changes that are specific to only some variables or, alternatively, include redundant information about others. Such situations arise, e.g., when the changes of one variable occur at a faster pace than others or the changes in the variables take place in distinct time intervals of the simulation.

It is also possible that one variable is considered more important than another for the purposes of the analysis and therefore needs to be treated in more detail.

In this paper, these issues are resolved by utilizing multiple time scales for state variables. In addition, a dynamic programming (DP, e.g., Bertsekas 1995) algorithm similar to (Gluss, 1962) is used to determine the time scales. When multiple time scales are considered in DBN metamodels, the time instants in the DBN are selected independently for each variable. This offers an improvement for the structure of the DBN metamodel. The application of multiple time scales results in a more accurate representation of the time evolution of the simulation without increasing the size of the metamodel.

The paper is organized as follows. The construction of DBN metamodels from simulation data is introduced in Section 2. The utilization of the DBN metamodels in simulation analysis is briefly presented in Section 3. Examples of a DBN metamodel with multiple time scales as well as its application are given in Section 4 where a DBN metamodel is used for probabilistic inference regarding the operation of a simulated air base.

2 Construction of DBN metamodels

The first step in the construction of a DBN metamodel is the selection of variables. While a DES model includes all the variables that significantly affect the behavior of the system, the subset of variables included in the DBN metamodel is selected based on how the DBN is going to be utilized. Assume now that state variables $x_1(t),\dots,x_n(t)$, where t refers to time, as well as input and output variables $u_1,\dots,u_m$ and $z_1,\dots,z_\ell$ of the DES model are included in the metamodel.

The second step in the construction is the design of experiment. Only discrete variables are allowed in DBN metamodels. The values of input variables are therefore discretized, which is discussed in more detail below. When constructing a DBN metamodel, a number of simulation replications are performed for all the combinations of the values of the inputs. A lower limit for the number of replications is calculated based on the objectives of the analyses (for details, see Poropudas and Virtanen 2011). If the number of data is found insufficient later on in the validation step of the construction, additional replications can be performed. The third step of the construction, i.e., the simulation, is performed once the values of the input variables and the number of replications are determined.

Due to the nature of DBNs, the values of state variables x_k are restricted to discrete sets X_k. Thus, the discretization is the fourth step of the construction. The elements of X_k and the manner in which the actual values of x_k are mapped onto them is decided on a case by basis by taking advantage of prior knowledge of the system. If no natural discretization of the variables is available, the values are mapped into a set of discrete bins with the help of general clustering algorithms such as k-means (Hartigan and Wong, 1979). The same procedure is applied to the input and output variables. The input variable u_k obtains values from the discrete set denoted by U_k and the output variable z_k from the discrete set denoted by Z_k.

DBNs are discrete time models where each state variable is considered at a finite number of time instants. In this paper, the time instants are allowed to vary from variable to variable and they are selected separately for each one. This constitutes the fifth step in the construction. The state variable x_k is considered at the time instants

$$T_k = \left\{t_0, t_1^k, t_2^k, \dots, t_f\right\}, \tag{1}$$

where t_i^k are chosen from the interval (t_0, t_f). Here t_0 and t_f refer to the starting and terminating times of the simulation, respectively, which are assumed to be identical for every replication. The DBN metamodel considers the joint probability distribution of all the variables at all the time instants. The estimate for the probability of the variable x_k obtaining the value $j \in X_k$ at time instant $t \in T_k$, i.e., $P(x_k(t) = j)$, provided by the DBN metamodel is denoted by $\hat{p}_j^k(t)$. A linear interpolation technique is used to construct estimates at the probabilities for time instants that are not included in the DBN. This results in estimates of the form

$$\hat{p}_j^k(t) := \hat{p}_j^k(t_-) + \frac{t - t_-}{t_+ - t_-}\left(\hat{p}_j^k(t_+) - \hat{p}_j^k(t_-)\right), \tag{2}$$

where $t \notin T_k$, $t_- = \max\{v \in T_k | v \le t\}$, and $t_+ = \min\{v \in T_k | v \ge t\}$.

The selection of the time instants T_k begins with the discretization of the time interval $[t_0, t_f]$ into the equally spaced instants

$$T_k^* = \{t_0, t_0 + \delta_k, \dots, t_0 + (m_k - 1)\delta_k, t_f\}, \tag{3}$$

where m_k is the number of segments for variable x_k and $\delta_k = (t_f - t_0)/m_k$. The time instants T_k are selected from among the time instants T_k^*. The probability estimates $P(x_k(t) = j) = p_j^k(t)$ of the variable x_k, which are based on the simulation data, are calculated for each time instant T_k^* and each value $j \in X_k$. The time evolution of these probabilities is referred to as the probability curves of the variable x_k. Now, the objective is to select the time instants T_k in such a manner, that the corresponding probabilities provided by the DBN metamodel follow the probability curves closely, while keeping the number of the time instants T_k low.

To quantify the accuracy of the DBN metamodel, the sum of squared error

$$M_k(T_k) = \sum_{t \in T_k^*} \sum_{j \in X_k} \left(p_j^k(t) - \hat{p}_j^k(t)\right)^2, \tag{4}$$

is used. The DBN is constructed so that $p_j^k(t) = \hat{p}_j^k(t)$ for all time instants $t \in T_k$. This means that the probabilities

$\hat{p}_j^k(t)$ are known prior to the construction of the DBN. The problem of selecting the time instants T_k then consists of selecting the number of time instants to include and finding their optimal location which minimizes $M_k(T_k)$.

Since the squared error is summed over all the time instants T_k^*, and the probability estimate given by the DBN for any single time instant depends only on the preceding and the following time instant in T_k, the error is calculated for one segment between consecutive time instants in T_k at a time. The errors are then aggregated to provide the total error $M_k(T_k)$. Thus, it is not necessary to evaluate every potential T_k as a whole because only each pair of time instants needs to be considered separately.

Optimal time instants are found by using dynamic programming in a manner similar to (Gluss, 1962). The algorithm iterates through all the pairs of time instants in T_k^* and calculates the total error for the segment from the one instant $t_0 + a\delta_k$ to another $t_0 + b\delta_k$ by assuming that there are no time instant $t \in T_k$ between them. The total error for such a segment is denoted as

$$D(a,b) = \sum_{i=a}^{b} \sum_{j \in X_k} \Big(p_j^k(t_0 + i\delta_k) - \left(\frac{b-i}{b-a} p_j^k(t_0 + a\delta_k) + \frac{i-a}{b-a} p_j^k(t_0 + b\delta_k) \right) \Big)^2. \quad (5)$$

The optimization problem is solved by considering subproblems where subsets of the form $\{t_0, t_0 + \delta_k, \ldots, t_0 + b\delta_k\}$ with $0 \leq b \leq m_k$ are considered. Optimal selections of time instants within each such a set are determined. Let $f_l(b)$ denote the resulting minimum error in such a subproblem when l time instants are used to estimate the probability curves between t_0 and $t_0 + b\delta_l$. The time instants t_0 and $t_0 + b\delta_k$ must always be included in the solution, so $l \geq 2$. For all l, $f_l(0) = 0$ and, for all $b \leq m_k$, $f_2(b) = D(0,b)$. For other values of l and b, the value of $f_l(b)$ is determined by the equation

$$f_l(b) = \min_{0 \leq i < b} \{ f_{l-1}(i) + D(i,b) \}, \quad (6)$$

where $D(i,b)$ is given by Eq. (5). $f_l(b)$ is evaluated for each value of l from 2 to the maximum value l_k. Then, the value of b is increased by one and $f_l(b)$ is again evaluated for each value of l. This is repeated until b has gone from 1 to m_k, at which point the algorithm terminates. The optimal time instant sets covering the entire time interval $[t_0, t_f]$ and consisting of any number of time instants up to the maximum l_k have then been calculated. The optimal sets containing different numbers of time instants are compared and the most suitable one is identified.

The structure of the DBN metamodel, consisting of nodes and arcs between them, is determined once T_k is chosen for each variable x_k. The construction is aided by BN software such as GeNIe (Decision Systems Laboratory). In the DBN, nodes are included for each variable x_k at all of the time instants T_k. A node is also associated with each input and output variable. If prior information about the system under consideration is available, the sixth step consists of using this information to define the known dependencies between nodes. Arcs implying dependence between specific nodes can be included regardless of the simulation data. In order to maintain causality, arcs going from a state variable to an input variable, from an output variable to an input variable, from an output variable to a state variable and from a state variable to a state variable at an earlier time instant are not allowed.

The seventh step in the construction consists of finalizing the structure of the network and determining its CPTs. The realized values of each state variable x_k at all of the time instants T_k are recorded for every replication of the simulation model, as are the values of all input and output variables. The structure is completed by applying learning algorithms (Heckerman et al., 1995) on the simulation data. The CPTs are constructed in accordance to the relative frequencies of the values in the data (Poropudas and Virtanen, 2011).

For the input variables, the relative frequencies in the simulation data do not necessarily reflect the actual probability distributions in question because they can be modified as part of the design of experiment step in order to collect a broader set of data. The probability distributions of the input variables are adjusted after the construction of the DBN metamodel in order to represent input certainty (Pousi et al., 2013). The distributions can be modified only after validating the metamodel, because the adjusted distributions for the inputs are not consistent with the validation data.

3 Utilization of DBN metamodels

The constructed DBN metamodel provides the joint probability distribution of the input, output, and state variables. The DBN is applied for various what-if analyses where conditional probabilities and probability distributions are studied. In these analyses, the values of some variables at given time instants are fixed and the conditional probability distributions of the other variables are updated using readily available algorithms implemented in BN software such as GeNIe (Decision Systems Laboratory). When considering conditional probabilities, the chronological order of the time instants is irrelevant, i.e., the conditional probability distributions can be calculated also for conditions related to later time instants.

The most basic application of the DBN metamodel is to determine the marginal probability distribution of a state variable as a function of time. Such a distribution consists of the probabilities of the state variable obtaining a given value at a given time, i.e., $P(x_k(t) = j)$. The marginal probabilities for outputs $P(z_k = j)$ can also be obtained. Conditional probabilities for the state variables are also obtained by setting conditions for inputs $P(x_k(t) = j | u_{k'} = j')$, state variables $P(x_k(t) = j | x_{k'}(t') = j')$, outputs $P(x_k(t) = j | z_{k'} = j')$, or any combination of these. Conditions can even be set for the same state variable that is being investigated, as long as the time instants are different. Conditional probabilities for output variables are calculated similarly. To create input-output mappings, conditions are set for input variables and the conditional probability distributions of the outputs, e.g., $P(z_k = j|u_{k'} = j')$, are studied. Conditional distributions of the outputs can also be studied by fixing the values of state variables, e.g., $P(z_k = j|x_{k'}(t) = j')$. If the metamodel includes multiple output variables, conditions can be set for some of them as well resulting in conditional probabilities such as $P(z_k = j|z_{k'} = j')$. The conditional probability distributions for input variables are investigated by setting conditions for the state variables $P(u_k = j|x_{k'}(t) = j')$, output variables $P(u_k = j|z_{k'} = j')$, or both. This inverse reasoning can be used to investigate, e.g., which combination of input values is most likely to lead to a certain outcome.

If the analysis involves probability distributions related to time instants not included in the DBN, the interpolation discussed in Section 2 is applied. The interpolation can also be applied to conditions taking place at time instants not included in the DBN. The details of the interpolation scheme are presented in (Poropudas and Virtanen, 2010).

4 Example analysis - simulated operation of air base

In this example, a DBN metamodel is constructed according to the guidelines discussed in Section 2 and used in the simulation analysis of the operation of an air base. In the model, aircraft go a through a cycle consisting of mission assignment, mission execution, repair of possible damage obtained during the mission, and standard service such as fueling. There are three queues for the aircraft: one for mission assignment, one for repair, and one for service. The repair and service personnel can only work on one aircraft at a time. The aircraft that have not been damaged move directly from the mission to the service queue. An aircraft is released from the mission assignment queue every time a new mission is to be executed. If there are no aircraft in this queue, a backlog of missions is formed and the aircraft are assigned to the missions as soon as they arrive from the service. A flowchart of the simulation model is presented in Fig. 1.

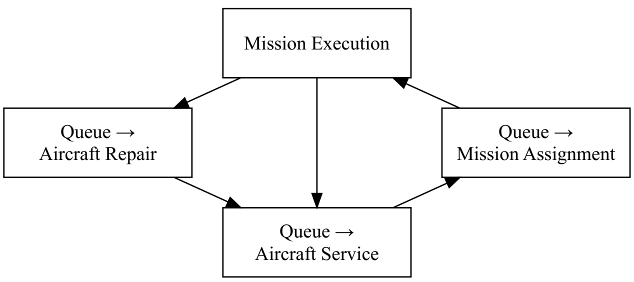


Figure 1. Flowchart of the simulation model.

The missions are categorized into patrol missions and combat missions. The patrol missions are assigned regularly with the time between consecutive missions sampled from a uniform probability distribution. The aircraft are unlikely to be damaged during a patrol mission. Combat missions are assigned as a Poisson process with a time dependent arrival intensity. They are on average shorter than the patrol missions but the aircraft have a much higher probability of being damaged. The repair time of a damaged aircraft is exponentially distributed. The service time is deterministic and depends on the length and type of the preceding mission.

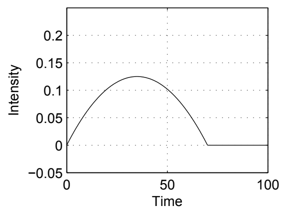
Table 1. Variables of the metamodel.

	Type	Range	Interpretation
u	Input	$\{1,\dots,4\}$	Scenario
x_1	State	$\{0,\dots,4\}$	Aircraft in assignment
x_2	State	$\{0,\dots,4\}$	Aircraft in repair
z	Output	$\{0,1\}$	Insufficient aircraft available

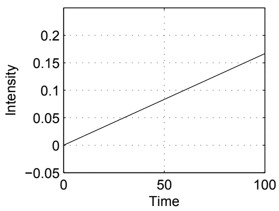
The input variable of the simulation model, denoted by u, determines the time dependent intensity of the occurrence of the combat missions. In this example, four alternative scenarios are studied. The number of aircraft in each of the four locations are considered as state variables. Two of the state variables, i.e., the number of aircraft in mission assignment, denoted by $x_1(t)$, and in aircraft repair, denoted by $x_2(t)$, are included in the DBN metamodel. The output variable of the simulation model is an indicator, denoted by z, that determines whether or not at any time during the simulation no aircraft are available to execute an incoming mission. The variables included in the DBN and their ranges are summarized in Table 1.

In order to acquire data for construction of the DBN, four scenarios are simulated. In the first one, the arrival intensity of the combat missions starts at 0, peaks early in the simulation, and returns to 0 later on. In the second scenario, the intensity slowly increases throughout the simulation. In the third one, the intensity is constant. In the fourth scenario, there are no combat missions. The non-zero arrival intensities of the first three scenarios are illustrated in Fig. 2. The four scenarios occur with equal probability. In every simulation replication, four aircraft are included. The data is collected by running 2000 simulation replications for each scenario. The duration of each replication is 100 units of time. A quarter of the data is reserved for validation. Since all the variables under consideration are discrete, there is no need for their discretization.

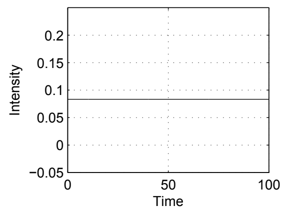
The probability curves of the state variables calculated from the simulation data are shown in Fig. 3. The advantage of utilizing multiple time scales is evident. The probability distribution of x_1 changes repeatedly due to the regularly scheduled patrol missions while the distribution of x_2 changes more slowly. Determining the optimal time instants for the state variables using the DP algorithm discussed in Section 2, a suitable number of time instants for



(a) Scenario $u = 1$.

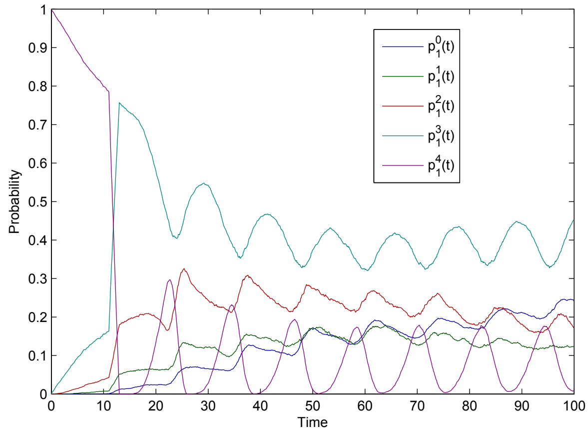


(b) Scenario $u = 2$.

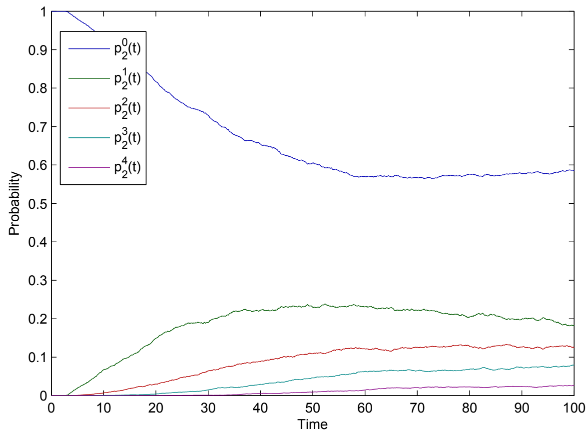


(c) Scenario $u = 3$.

Figure 2. Intensity of the generation of combat mission in three scenarios.



(a) Number of aircraft in mission assignment, x_1.



(b) Number of aircraft in aircraft repair, x_2.

Figure 3. Time evolution of the marginal probability distribution of the state variables as estimated from the simulation data.

x_1 is found to be 25. For x_2, seven time instants are used.

Prior knowledge is used to add arcs to each node in the DBN from the most recent node corresponding to each variable. Arcs originating from nodes corresponding to the time instant 0, except those leading to the following node corresponding to the same variable, are ignored, since the initial state of the simulation is always same. Arcs are also added from the input variable to all other nodes and from every other node to the output variable. The arcs determined in this manner are sufficient to describe the entire system because no additional dependencies are evident in the simulation data.

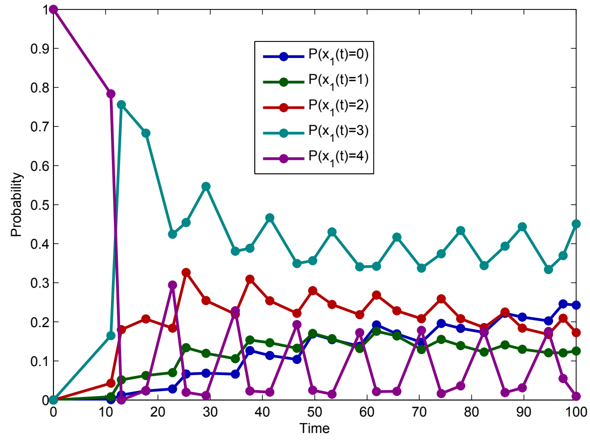
Fig. 4 depicts the unconditional time evolution of the simulation, i.e., the time evolution of the marginal probability distributions of the two state variables provided by the DBN metamodel. The distributions resemble the probability curves in Fig. 3. The periodical nature of x_1, caused by the regular patrol missions, is evident in Fig. 4a. This is also the main reason why the concept of multiple time scales is useful in this example. The patrol missions directly affect the number of aircraft available for missions, but have little impact on the number of aircraft needing repair.

Next, alternative what-if analyses allowed by the DBN metamodel are illustrated. The first of the four scenarios is examined in Fig. 5 by setting the condition $u = 1$. When comparing to Fig. 4, there are fewer aircraft ready for missions and more in need of repair during the middle of the time interval but the situation is reversed by the end of it. This is consistent with the intensity of the generation of combat missions presented in Fig. 2.

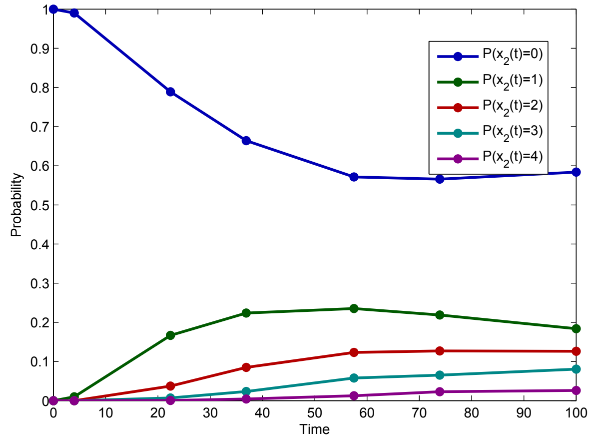
In order to further investigate conditional properties of the simulation model, the condition regarding u is removed and the condition $x_1(100) = 0$ is instead added, i.e., every aircraft is either on a mission, being repaired, being serviced or in a queue waiting for one of the latter two activities at the end of the simulation. The probability of this event is $P(x_1(100) = 0) = 0.24$. Fig. 6 presents the time evolution of the conditional probability distributions of the state variables. The likely number of aircraft ready for mission decreases steadily in Fig. 6a. The expected number of aircraft in need of repair increases conversely in Fig. 6b, but the most likely values of $x_2(100)$ are 2 and 3 which means that one or two aircraft are probably still either carrying out a mission, being serviced, or waiting for service.

The condition $x_1(100) = 0$ also affects the output variable z. The probability distribution of z without and with the condition is presented in Table 2. The condition greatly increases the probability of z obtaining the value 1. This is as expected since a mission with no aircraft available to carry it out can only occur if x_1 obtains the value 0 at some point.

This example demonstrates just some of the capabilities of DBN metamodels with multiple time scales. With more variables and replications of the simulation model, more elaborate what-if analyses can be performed. By fully utilizing existing BN software, this can be done programmat-

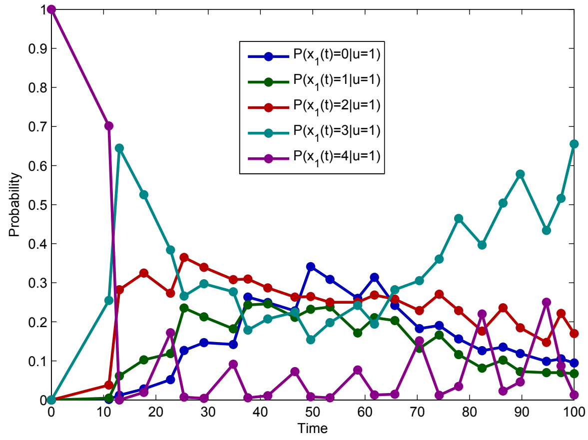


(a) Number of aircraft in mission assignment, x_1.

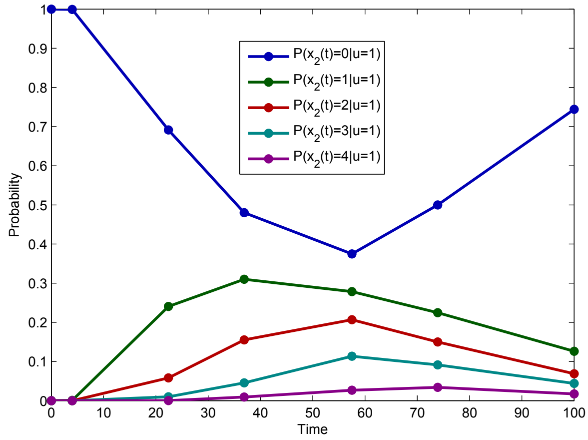


(b) Number of aircraft in aircraft repair, x_2.

Figure 4. Time evolution of the marginal probability distributions of the state variables provided by the DBN metamodel.



(a) Number of aircraft in mission assignment, x_1.



(b) Number of aircraft in aircraft repair, x_2.

Figure 5. Time evolution of the conditional probability distributions of the state variables provided by the DBN metamodel conditional on $u = 1$.

ically, which greatly enhances the number of probability estimates that can be calculated in a reasonable amount of time.

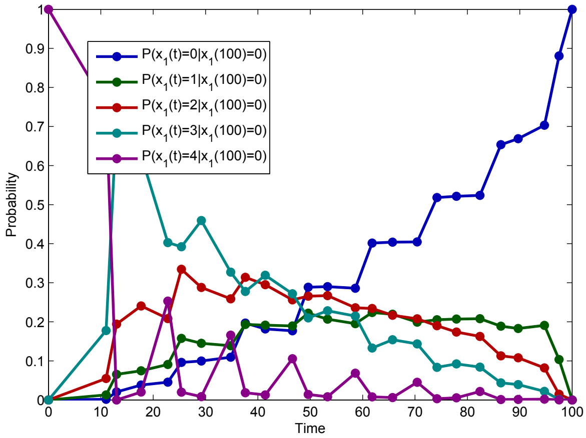
5 Conclusions

The time evolution of DES can be analyzed in a transparent manner by using DBNs as simulation metamodels. The DBN metamodels offer an effective way for conducting various what-if analyses. In the previous literature, the structure of DBN metamodels has consisted of time slices, i.e., the networks have had a rigid structure where all state variables of the model are considered at each of the time instants represented by the DBN. In this paper, the concept of multiple time scales is introduced to the DBN metamodeling, i.e., the changes in the probability distributions of the state variables are allowed to occur independently in the DBN and the time evolution of individual state variables is studied at its own pace. By employing multiple time scales, the different temporal changes in the behavior of the state variables are described more accurately without unnecessary increase in the size of the DBN. The paper also presents an algorithm based on dynamic programming for the optimal selection of time instants represented by the DBN. The construction and utilization of the DBN metamodel with multiple time scales are demonstrated with an example analysis involving the operation of an air base.

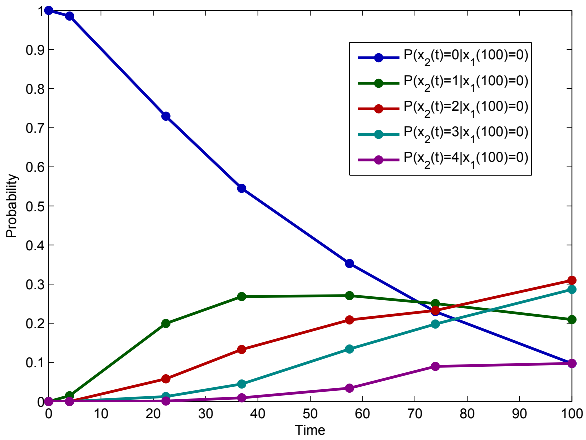
Table 2. Marginal and conditional probability distributions of the output variable when $x_1(100) = 0$.

j	$P(z=j)$	$P(z=j\|x_1(100)=0)$
0	0.63	0.17
1	0.37	0.83

Simulation studies using DBN metamodels can be performed with software designed for the analysis of BNs. Unfortunately, the dynamic programming algorithm and the interpolation technique used for approximative reasoning are beyond the scope of such software and, thus, the calculations presented in this paper have been carried out using MATLAB. In order to alleviate future studies, it is worthwhile to develop an automated tool designed for the construction and utilization of DBN metamodels.



(a) Number of aircraft in mission assignment, x_1.



(b) Number of aircraft in aircraft repair, x_2.

Figure 6. Time evolution of the conditional probability distributions of the state variables provided by the DBN metamodel when $x_1(100) = 0$.

The DBN metamodels have also been used in simulation-based optimization as a part of influence diagram metamodels (Poropudas and Virtanen, 2009). In such metamodels, the DBN reveals the consequences of decision alternatives, i.e., the time evolution of a simulated system with given values of simulation parameters. Clearly, the concept of multiple time scales could also be applied in the construction of influence diagram metamodels from simulation data.

References

D.P. Bertsekas. *Dynamic programming and optimal control*. Athena Scientific Belmont, MA, 1995.

Decision Systems Laboratory. GeNIe (graphical network interface). Available via https://www.bayesfusion.com/ [accessed November 8, 2017].

L.W. Friedman. *The simulation metamodel*. Springer Science & Business Media, 2012.

B. Gluss. Further remarks on line segment curve-fitting using dynamic programming. *Communications of the ACM*, 5(8): 441–443, 1962.

J.A. Hartigan and M.A. Wong. Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

D. Heckerman, D. Geiger, and D.M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine learning*, 20(3):197–243, 1995.

J.P.C. Kleijnen. *Design and analysis of simulation experiments*, volume 20. Springer, 2008.

A.M. Law and W.D. Kelton. *Simulation modeling and analysis*. McGraw Hill Boston, 2000.

K.P. Murphy. *Dynamic bayesian networks: representation, inference and learning*. PhD thesis, University of California, Berkeley, 2002.

J. Pearl. Fusion, propagation, and structuring in belief networks. *Artificial intelligence*, 29(3):241–288, 1986.

J. Poropudas. *Bayesian Networks, Influence Diagrams, and Games in Simulation Metamodeling*. Doctoral dissertation, Aalto University School of Science, 2011. Available via http://lib.tkk.fi/Diss/2011/isbn9789526042688/ [accessed November 8, 2017].

J. Poropudas and K. Virtanen. Analyzing air combat simulation results with dynamic Bayesian networks. In *Proceedings of the 2007 Winter Simulation Conference*, pages 1370–1377. IEEE Press, 2007.

J. Poropudas and K. Virtanen. Influence diagrams in analysis of discrete event simulation data. In *Proceedings of the 2009 Winter Simulation Conference*, pages 696–708. Winter Simulation Conference, 2009.

J. Poropudas and K. Virtanen. Simulation metamodeling in continuous time using dynamic bayesian networks. In *Proceedings of the 2010 Winter Simulation Conference*, pages 935–946. Winter Simulation Conference, 2010.

J. Poropudas and K. Virtanen. Simulation metamodeling with dynamic Bayesian networks. *European Journal of Operational Research*, 214(3):644–655, 2011.

J. Pousi, J. Poropudas, and K. Virtanen. Simulation metamodelling with Bayesian networks. *Journal of Simulation*, 7(4): 297–311, 2013.

B.P. Zeigler, H. Praehofer, and T.G. Kim. *Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems*. Academic press, 2000.