Efficient Risk Based Optimization of Large System Models using a Reduced Petri Net Methodology

Susannah Naybour

Resilience Engineering Research Group, University of Nottingham, United Kingdom.
E-mail: Susannah.Naybour@nottingham.ac.uk

John Andrews

Resilience Engineering Research Group, University of Nottingham, United Kingdom.
E-mail: John.Andrews@nottingham.ac.uk

Manuel Chiachio-Ruano

Institute for Data Science and Computational Intelligence (DaSCI) & Dept. Structural Mechanics and Hydraulics Engineering, University of Granada, Spain.

The methodology presented in this paper is a two-stage optimization approach that can be applied to large system level models, in this case using a Stochastic Petri Net (SPN) framework, to produce an equivalent model response at a reduced computational cost. The method consists of generating a reduced SPN which approximates the behavior of its large counterpart with a shorter simulation time. Parameters in this reduced structure are updated following a combined Approximate Bayesian Computation and Subset Simulation framework. In the first stage, optimization of the reduced model via a Genetic Algorithm provides a first approximation of the optimal solutions for the full system level model. In the second stage, these approximate optimal solutions then form the starting point of a short optimization of the large SPN to fine tune the results using a reduced solution space. This method is demonstrated for a sub-section of an SPN of a fire protection system. Optimization of the full model with a Genetic Algorithm is compared to the optimization through this two-stage approach to demonstrate the capability of the methodology. Results show good model agreement at a reduced computational cost.

Keywords: Petri Nets, Risk, Optimization, Genetic Algorithms, Approximate Bayesian Computation, Subset Simulation.

1. Introduction

Stochastic Petri Nets (SPNs) have been applied widely to model the failure of large and complex systems. Examples of this can be found in Andrews (2013) and Andrews and Fecarotti (2017). Simulation tools such as Monte Carlo Simulation can be used for the analysis of such models. Optimization algorithms can be applied to SPNs to provide an asset management decision making tool, to save on system life-cycle cost while ensuring safe operation. However, for large and complex SPNs these optimization algorithms are computationally costly.

The use of SPNs has been shown to be effective in Senderovich et al. (2018) but is computationally expensive due to the requirement of a large number of simulations to obtain convergence for the marking sequences of the SPN.

SPNs can be combined with optimization techniques such as Genetic Algorithms or Simulated Annealing to find an optimal maintenance and inspection strategy to reduce risk, or unavailability. This has been done in work such as Yang et al. (2015) and Lee (2014). A drawback of applying an optimization technique to an SPN is the repeated requirement for a convergent result for each trial within the optimization since, a Monte Carlo Simulation is required each time.

There have been several studies aiming to simulate SPNs more efficiently such as in Geist et al. (2005) and Yianni et al. (2018). The methods used here, though effective, require specialist hardware and software and can still be costly in terms of computational time.

This paper presents a methodology for the efficient optimization of an SPN based model, known as the reference SPN. Initially, the reference SPN is reduced in size, then a 2-stage optimization is applied implementing the optimal solution space of the reduced SPN, hence decreasing the search space for the optimization of the reference SPN, which improves the efficiency of the optimization process. The remainder of the paper is organized as follows. Section 2 gives an introduction to

Proceedings of the 29th European Safety and Reliability Conference.
Edited by Michael Beer and Enrico Zio
 Published by Research Publishing, Singapore
ISBN: 981-973-0000-00-0 :: doi: 10.3850/981-973-0000-00-0_Efficient˙Risk˙Based˙Optimization˙of˙Large˙System˙Models˙using˙a˙R

the Petri Net (PN), and SPN, methodology. Section 3 briefly overviews the overall optimization methodology. In Section 4, the fundamentals for SPN model reduction, are presented. Section 5 provides the parameter updating methodology which is further used in Section 6 for the optimization problem. The methodology is exemplified in Section 7. Finally, Section 8 gives concluding remarks.

2. Petri Nets

Petri Nets (PNs) are a digraph consisting of two types of nodes, known as places and transitions, connected by arcs. Associated with the PN structure is a rule known as the *firing rule* which allows tokens, which sit within places, to be created and destroyed to represent changes in the system being modeled, as described in David and Alla (2010). The firing rule states that if all the places that are connected to a transition are marked by a token, then firing can occur. During firing, tokens are destroyed in these input places and created in the output places, which are those connected from the transition. A full definition can be found in Murata (1989).

In practical applications, transitions are typically assigned time delays, this is useful for performance evaluation and scheduling problems of dynamical systems (Ajmone Marsan et al. (1998)). The resulting PNs are called *Timed Petri Nets* if the delays are deterministic, and *Stochastic Petri Nets* if the delays are specified by a probability model (Molloy (1982)). In such cases, a transition is fired once its time delay has passed, provided that the firing rule is satisfied, leading to a change in the marking sequence, $\mathbf{M}_k$ of the PN.

For large and complex SPNs with a variety of transition types, finding an exact analytical solution to the SPN is practically impossible. Simulation tools, such as Monte Carlo Simulation, can be employed to find the average marking sequences based on the probability model associated with the transitions. Transitions within the SPN are each assigned a probability distribution. A run of the Monte Carlo Simulation is initiated by marking the SPN with tokens; this enables firing of some transitions in the SPN. Each time a transition within the SPN is enabled, the firing time is sampled from its associated probability distribution. Upon the firing of this transition the firing time is then re-sampled if it becomes enabled again. This process is carried out for each of the transitions within the model as they are enabled. The outputs of the SPN for a given time period are then recorded and the SPN is returned to it's initial condition in order to begin the next run. This process is repeated with numerous runs until a convergent solution is reached.

3. Overview of Optimization Methodology

For a defined SPN reference structure, with an associated optimization problem, the optimization methodology presented in this paper requires the following steps:

(i) Define the key outputs of the reference SPN for comparison with a proposed reduced structure,
(ii) Define the reduced model structure,
(iii) Identify parameters in the reduced model structure for updating,
(iv) Update parameters,
(v) Validate the reduced structure by comparing the reduced model outputs to the outputs for the reference SPN,
(vi) Find the approximate optimal solution space using the reduced model structure,
(vii) Find the optimal solution space for the reference SPN by searching a reduced solution space based on the approximate optimal solution found in the previous step.

These steps are expanded in the remaining sections of this paper. A numerical example is given in Section 7.

4. The Reduced Model Structure

The aim of the reduction methodology is to develop a technique for an SPN, known as the reference SPN, to be represented by a smaller SPN, known as the reduced SPN.

Central to this reduction method is the definition of key model outputs that are present in both the reference and reduced model structures. For this methodology to be applied, the reduced SPN must have at least:

- The capacity to reproduce the key output, or outputs, of the reference SPN;
- The capacity to incorporate the behaviors requiring optimization.

A reduced SPN structure must give a reasonable approximation to the reference SPN, while sufficiently reducing the complexity. Different reduced structures can be tested to consider their level of approximation. A measure of this arises naturally during the parameter updating process.

It is recommended that the reduced structure is decided based on the knowledge of the modeler to prevent restrictions on the possible simplifications, however work can be found for the optimal simplification of PN models in Pham and Karaboga (2000) following certain rules. The combination of states corresponding to different failure mechanisms is an intuitive choice for a reduction.

5. Parameter Updating

Parameters within the probability distributions governing the reduced SPN, can be updated based on the SPN's key outputs in comparison to the outputs of the reference SPN. In this methodology, the ABC-SubSim algorithm, first published in Chiachio et al. (2014), is used to update the parameters within the reduced SPN. Before the method can be applied, a choice must be made on the number and location of parameters to update and the basis for which the suitability of the parameters will be assessed.

Multiple parameters within the reduced SPN can be updated in order to give an improved approximation to the reference SPN. However, additional computational effort is required for each parameter updated. Hence, there is a trade-off between the approximation made by the reduced model and the efficiency of the updating methodology. Good results have been found by updating parameters governing the transitions where the highest simplification of the model has been carried out.

Within this process, the place, or places, representing the key outputs of the reference SPN are identified. The corresponding place, or places, in the reduced SPN are also selected. The marking sequence of these comparison places forms the basis for which the reference SPN and reduced SPN can be compared. It is important to choose effective comparison places that hold the same meaning in each SPN and contain the required information from the original model.

Next, once the reduced SPN is identified, parameter updating is required to make it approximate the reference SPN. Bayesian model updating provides a methodology to make inferences about parameters of a model based on experimental data in order to find a posterior parameter region (Box and Tiao (1992)).

There are some model classes where Bayesian model updating cannot be used directly for parameter updating, such as for an SPN. In cases such as these, Approximate Bayesian Computation (ABC) methods can be used to provide a framework for parameter inference and model selection. In ABC methods, parameters θ' are used to simulate model outputs, x, and higher weights are given to regions where these values are closer to the true posterior, y. To find a region where $x \approx y$, a tolerance parameter ϵ is introduced that represents closeness of the simulated outputs to the true posterior, judged by a metric value ρ gained by a summary statistic $\eta(\cdot)$. Through this approach, the posterior is approximated to assign higher probability density to parameter values that satisfy the condition $\rho\big(\eta(x), \eta(y)\big) \leqslant \epsilon$. An algorithm to generate N samples by ABC is given in Algorithm 1.

Algorithm 1 Standard ABC

for $t = 1$ to N **do**
  repeat
    1.- Simulate θ' from $p(\theta)$
    2.- Generate $x' \sim p(x|\theta')$
  until $\rho\big(\eta(x'), \eta(y)\big) \leqslant \epsilon$
  Accept (θ', x')
end for

The success of the ABC algorithm is dependent on a good choice of the summary statistic $\eta(\cdot)$, metric choice ρ and tolerance parameter ϵ. For small posterior regions, ABC can be computationally expensive as a high quantity of simulations is required to reach a significant number of parameter values within the required tolerance. There have been several algorithms developed to decrease the computational time for the ABC algorithm, and some of these can be found in Michel et al. (2012). For this reduction methodology the ABC-SubSim algorithm has been chosen as a sufficiently good algorithm to reduce the computational effort of ABC. In this methodology the parameters are updated based on a summary statistic comparing the key outputs of the reduced SPN to the reference SPN. The ABC-SubSim algorithm combines ABC with SubSet Simulation and a full description of the ABC-SubSim algorithm can be found in Chiachio et al. (2014).

Following the updating of the parameters within the reduced SPN the outputs must be compared to ensure that the approximation made by the reduced SPN is reasonable. A measure of the difference in the key outputs can be found using the same summary statistic as in the parameter updating stage. For the reduced SPN, where approximations have been made in the reduced structure, slight differences are expected in the outputs.

6. Optimization

A Genetic Algorithm is implemented in this paper to find the optimal solutions for the maintenance and inspection intervals of a component in order to reduce the probability that the component is in the unrevealed failed state. This optimization method can be applied to a larger asset management model in order to reduce risk and life-cycle cost of the system in question as in Yianni (2017).

A full description of the Genetic Algorithm method can be found in Santos et al. (2018) and Holland (1983). In a Genetic Algorithm, an initial population is cross-bred and mutated over a number of generations, with the best candidates selected to proceed at each generation. A pseudo-code implementation is given in Algorithm 2.

In the methodology presented in this paper, a two-stage approach is used to find optimal solu-

Algorithm 2 Pseudo-code implementation for a Genetic Algorithm

Inputs:
$P_0 = p_1^0, p_2^0, ..., p_N^0$ {The initial population of N vectors, where p_n^0 is a vector of length L}.
J, {Fitness function, in this case the risk of the system and cost of interventions calculated via Monte Carlo Simulation of the Petri Net model}
I, {Number of iterations}
$S(P)$ {Selection operator, based on the fitness of the solution}
$X(p_a, p_b)$ {Crossover operator}
$M(P)$ {Mutation operator}
Algorithm:
for $j : 0, \ldots, (I-1)$ **do**
  $P = P_j$ {set the population at each iteration}
  for $k : 0, \ldots, N$ **do**
    $F_k = J(p_k)$
  end for
  $P_{parent} = S(P)$, where R is the number of parents following selection
  $P_{j+1} \leftarrow P_{parent}$
  for $q = 0, \ldots, R/2$ **do**
    $p_{child1}, p_{child2} = X(p_a, p_b)$, where p_a, p_b are vectors from the parent population and each parent is only used once
    $P_{j+1} \leftarrow p_{child1}, p_{child2}$
  end for
  $P_{j+1} = M(P_{j+1})$
end for

tions to the reference SPN, with an intermediate step applied to the reduced model. The first step in the optimization process is to find an optimal solution space for the reduced SPN. This is found by completing a number of generations of a Genetic Algorithm applied to the reduced SPN. The second step of the optimization approach is to define a region encompassing these solutions to give an approximate solution space. The third step is to use the approximate solution space as the initial population of a Genetic Algorithm, and to apply a low number of generations of the algorithm to the reference SPN to gain the optimal solution space. In summary, the optimization process is mostly performed on a smaller more efficient model, with the larger model reintroduced in the latter stages to fine-tune the solutions.

There are several decisions to be accounted for when implementing this approach in addition to those required when applying a Genetic Algorithm. Firstly, the number of generations that are applied to the reduced SPN before re-introducing the reference SPN must be decided. Secondly, the definition of the approximate solution space, given the population obtained from the reduced SPN optimization, must be defined. These decisions are dependent on the problem in question and more research should be completed into an automated approach for this. It is recommended that the number of generations applied to the reduced SPN is sufficient to see a good level of convergence in the solutions so that the search space can be adequately reduced. It is also recommended that the optimal solution space uniformly covers all values gained from the reduced SPN optimization, with some values outside of this range. This is recommended to allow the algorithm to explore the approximate space and to check for values outside, but close to, the values found from the reduced SPN optimization. Both these assumptions are made from the expectation that the optimal solutions from the reduced SPN will approximate the optimal solutions for the reference SPN.

7. Example

The reference SPN used as an illustrative example is given in Figure 1. This SPN can be used to model a repairable component and is taken from a wider model simulating a fire protection system. The input data used in this model can be found in Appendix A.

In this SPN, place $P1$ corresponds to the working state of the component and place P_4 corresponds to the failed state of the component. There are two pathways that can result in a failure, firstly through the age of the component modeled by transitions t_1, t_2 and t_3, and secondly through a randomly occurring failure modeled by transitions t_1 and t_5. Place P_6 corresponds to a revealed failure and place P_7 corresponds to a scheduled maintenance action, either due to a revealed failure, or the age of the component, represented by transitions t_6 and t_8 respectively. Place P_8 counts the number of maintenance actions.

The shaded regions are those that will be reduced and place P_4 is emphasized as its marking pattern over time is used as the key output for the SPN.

The optimization problem for this SPN aims to reduce the time that the component is in the unrevealed failed state by finding the optimal inspection and age-based maintenance intervals for the component, within a given cost constraint. This corresponds to reducing the time that place P_4 is marked, representing the unrevealed failed state, by altering the parameter values of the distributions governing transitions t_4 for the inspection interval and t_8 for the age-based maintenance.

A reduced Petri Net structure, for the reference SPN given in Figure 1, is presented in Figure 2. Here, place P_1' corresponds to the working state of the component and place P_2' corresponds to the unrevealed failed state of the component. Place P_3' corresponds to the revealed failed state

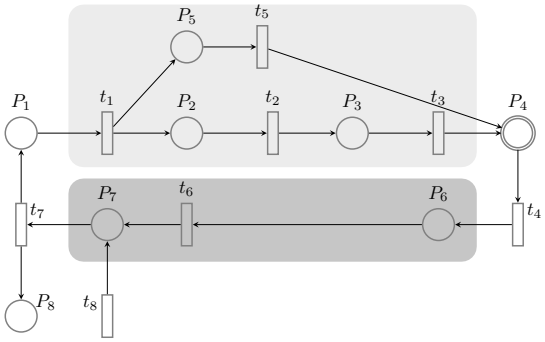

Fig. 1. Reference SPN used for illustration throughout this paper

of the component and place P_4' counts the number of maintenance actions, either due to a revealed failure or the age of the component. Transition t_4' represents periodic age-based maintenance. Place P_2' corresponds to the unrevealed failed state of the component represented by place P_4 in the reference SPN model. The shaded areas in the reduced SPN correspond to the shaded areas in the reference SPN and highlight the following reductions:

(i) The intermediate states of the component that lie between the working and failed states are absorbed into the single transition $t'1$;
(ii) The maintenance scheduling delay on failure is assumed to be much less than the inspection interval and so this delay, and the state corresponding to scheduled maintenance, are absorbed into the place $P'3$ to represent a state where failure is revealed and maintenance is scheduled.

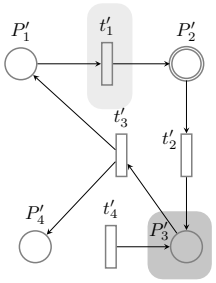

Fig. 2. A reduced SPN used for approximation of the SPN in Figure 1

For the reduced SPN presented in Figure 2 in comparison to the reference SPN presented in Figure 1, the parameters governing the Transition t_1' were updated. A 2-parameter Weibull distribution was assigned to this transition. The ABC-SubSim algorithm was implemented to find the region in the parameter space where the parameters governing this Weibull distribution resulted in the most similar reduced model output to that of the reference SPN.

The output used here was the marking of the place corresponding to an unrevealed failure in both SPNs. In this case, the centroid linkage was used as the summary statistic, alternatives can be found in Desa and Desa (2009). The metric value used for updating the parameters was the squared sum of the centroid linkage, for the marking of this place at each time.

Four levels of the ABC-SubSim algorithm were applied to update the parameters from a uniform prior region. This can be seen in Figure 3, where the prior region is enclosed in the dotted lines and each of the circles represents the pair of parameter values within the estimated posterior region at each level. The evolution of the parameters within each level was chosen adaptively to maximize convergence to the posterior region. Further discussion on this can be found in Chiachio et al. (2014). With repeated further levels of the algorithm, there was limited reduction in the posterior region, showing that after a point the approximation made by the reduced model could not become any more exact. Different structures or choices of parameters to update might yield a more exact approximation. This could be used to compare the approximations made by different model configurations.

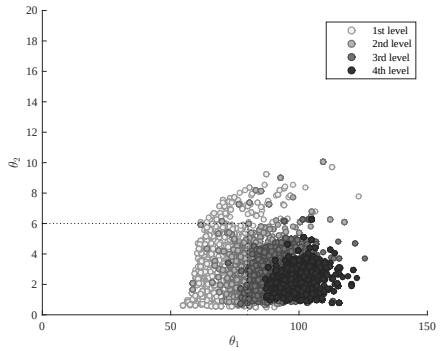

Fig. 3. The posterior region at each level of the ABC-SubSim algorithm; θ_1 gives the values for η and θ_2 gives the parameter values for β in the Weibull distribution.

The probability that the component is in the unrevealed failed state at each time, taken from each of the reference and reduced models, is given in Figure 4. It can be seen here that the behavior of the reduced SPN closely follows that of the reference SPN.

A Genetic Algorithm was applied to the SPNs presented in this paper. For comparison, the

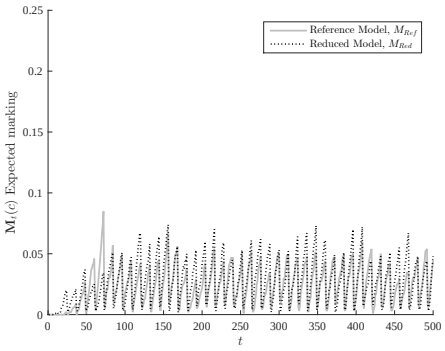

Fig. 4. The probability that the component is in the failed state at each time, taken from both the reference and reduced SPNs.

algorithm was applied to the reference SPN in isolation and the reduced SPN in isolation. Finally the Genetic Algorithm was applied across both SPNs using the two-stage approach.

In this example, the Genetic Algorithm had a population of 100 vectors and a mutation rate of 1 in 100. The initial population of 100 vectors was defined for parameter values where entries ranged from 1 to 100. For example, the first member of the population was a vector where all entries had the value 1 and for the last population member, all entries had the value of 100. For the reference SPN in isolation and the reduced SPN in isolation, eight generations of the Genetic Algorithm were completed. For the two-stage approach, five levels of the Genetic Algorithm were completed for the reduced SPN to find the approximate solution space. Following this, three levels of the Genetic Algorithm were applied to the reference SPN using the approximate solution space as the initial population. The selection operator was weighted such that the fittest individuals had a higher probability of selection.

An arbitrary cost was assigned to each of the inspection and maintenance actions and this was constrained to within 1000 units over the time period in question. The time that the component was in an unrevealed failed state was minimized subject to this constraint.

The approximate solution space was found by taking the maximum and minimum value of each variable in the population gained at the 5th generation of the Genetic Algorithm for the reduced SPN. The maximum and minimum values for each variable were found along with the difference between these. The approximate solution space, for this case, is defined by the region given by Eq.(1) and Eq.(2) for the two variables θ_1 and θ_2.

$$d_{\theta_n} = \sigma_n^{max} - \sigma_n^{min} \tag{1}$$

$$\sigma_n^{min} - d_{\theta_n} \leq \theta_n \leq \sigma_n^{max} + d_{\theta_n} \tag{2}$$

where σ_n^{max} is the maximum value for parameter n, and σ_n^{min} is the minimum value for parameter n, as found in the 5th generation of the Genetic Algorithm optimization of the reduced SPN, for values of $n = 0, 1$.

The initial population of 100 vectors was initiated within this region for the second stage of the optimization using the reference SPN. For this example, the nth entry of element, p_i of the population is defined as in Eq.(3).

$$p_i(n) = i \cdot \left\lfloor \left(\frac{\sigma_n^{max} - \sigma_n^{min} + 2d_{\theta_n}}{i_{max}} \right) \right\rfloor + \sigma_n^{min} - d_{\theta_n} \tag{3}$$

for a population member p_i, where i is in the range $[1, 100]$ and $i_{max} = 100$.

The results for the two-level optimization are given in Figure 5 for the optimal inspection interval and age-based maintenance interval. Each mark within the generation represents the parameter value of the individual population member with the horizontal bar representing the mean value of the population for the variable in question. The mean of the population for the optimization applied to the reference SPN in isolation was 9.21 time units and 9.03 time units for the inspection and maintenance interval, respectively. The mean of the population for the optimization applied to the reduced SPN in isolation was 11.00 time units for the inspection interval and 11.00 time units for the maintenance interval. The mean of the population for the two-level approach was 8.87 time units and 9.84 time units for the inspection and maintenance interval respectively.

The optimal solution for the reduced SPN occurs in a region that approximates the solution space of the reference SPN. However, by using the two-stage approach, the optimal solutions obtained more closely recreate the solutions for the reference SPN Genetic Algorithm optimization. In the two-stage approach, the impact of the method for finding the approximate region can be seen in the population values at the 6th generation by the more uniform nature of the members of the population in comparison to the previous population.

It is notable that the reduced SPN gives solutions that are close to the reference SPN, and that for some modeling scenarios it may be suitable to solely take the optimal values from the reduced SPN. However, the reduced SPN may not closely reproduce the time that the component is in the failed state given the parameter changes to the system. This is due to dependencies within the

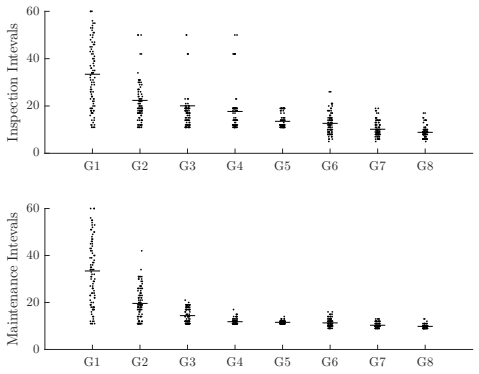

Fig. 5. The population for the two-level approach with each generation of the Genetic Algorithm when considering the inspection interval (above) and the maintenance interval (below) as the variables

model that are absorbed during the parameter fitting process. To clarify, the reduced SPN may be sufficient to mimic trends in the behavior of the reference SPN for different parameter values, to enable an approximation to be made in an optimization process, but may not be able to reproduce exactly the key model outputs when parameters are changed within the model.

A graph showing the different simulation times for the different optimization approaches is given in Figure 6. A reduction in the computational time for the optimization process can be seen for this simple example.

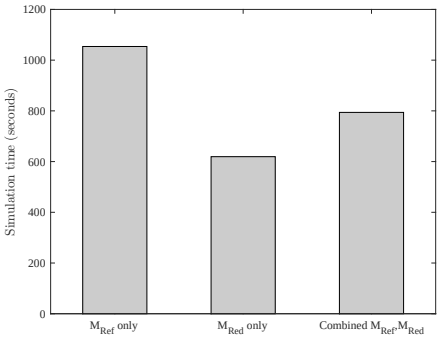

Fig. 6. A bar chart showing the computational time for optimization of: the reference SPN, (M_{Ref}), in isolation, the reduced SPN, (M_{Red}), in isolation and the 2-level approach combining both the reference SPN and the reduced SPN

7.1. *Discussion*

This methodology shows a reduction in time for the optimization of a simple SPN model, whereby the optimal solutions are closely recreated using an approximation of the SPN within a reduced structure. It is expected that a larger level of reduction in this approach will lead to a greater reduction in computational cost. This could be especially useful if there are multiple repeated components that can be modeled by the same reduced structure. One of the limitations of this method is the computational effort required to reduce the model size, in order to gain the reduction in computational time for the optimization. Further work can be completed to find an automatic way to reduce the structure of a large SPN and to decide on which parameters should be updated. Secondly, further work can explore the definition of the approximate optimal region and how this impacts the convergence of the population of the Genetic Algorithm.

8. Conclusion

This paper has presented a two-stage approach to the optimization of an SPN. Initially, a reduction of the model within an ABC-SubSim framework is completed followed by an optimization using this reduced model structure as an intermediate step towards the desired solution space. This is undertaken by initially optimizing the reduced structure to find an approximate region and using this approximate region as the basis for a shorter optimization of the full SPN structure. An illustrative example has been demonstrated in the paper.

Acknowledgements

This work was supported by the Lloyd's Register Foundation, a charitable foundation in the U.K. helping to protect life and property by supporting engineering-related education, public engagement, and the application of research. John Andrews is the Lloyd's Register Foundation Director of the Resilience Engineering Research Group and also the Network Rail Professor of Infrastructure Asset Management at the University of Nottingham. Manuel Chiachio has been partially funded by the ROBIN project reference PPJI2018.04 founded by the University of Granada. The authors gratefully acknowledge the support of these organizations.

Appendix A. Model Input Values

Table 1 gives the model inputs for the SPNs given in Figure 1 and Figure 2.

Table 1. The data used in the SPN models given in this paper.

Transition	Firing model	Parameters
t_1	Normal PDF	$\mu = 20, \sigma = 7$
t_2	Normal PDF	$\mu = 13, \sigma = 4$
t_3	Normal PDF	$\mu = 8, \sigma = 2$
t_4	Interval (Global)	$I = 6$
t_5	Uniform PDF	$C = 0.005$
t_6	Normal PDF	$\mu = 1, \sigma = 0.5$
t_7	Normal PDF	$\mu = 0.1, \sigma = 0.01$
t_8	Normal PDF	$\mu = 30, \sigma = 10$
t'_1	2-ParameterWeibull PDF	$\eta = 100.29, \beta = 1.64$
t'_2	Interval (Global)	$I = 6$
t'_3	Normal PDF	$\mu = 1, \sigma = 0.5$
t'_4	Normal PDF	$\mu = 30, \sigma = 1$

Note: Here the Interval (Global) firing model generates a firing time based on the difference between the global time and the time until the next integer value multiplied by interval I

References

Ajmone Marsan, M., A. A. Bobbio, and S. Donatelli (1998). *Petri nets in performance analysis: An introduction. In: Reisig W., Rozenberg G. (eds) Lectures on Petri Nets I: Basic Models: Advances in Petri Nets*. Springer Berlin Heidelberg.

Andrews, J. (2013). A modelling apporach to railway track asset management. *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit 227*, 56–73.

Andrews, J. and C. Fecarotti (2017). System design and maintenance modelling for safety in extended life operation. *Reliability Engineering & System Safety 163*, 95 – 108.

Box, G. and G. Tiao (1992). *Bayesian Inference in Statistical Analysis*. John Wiley and Sons, Inc.

Chiachio, M., J. Beck, J. Chiachio, and G. Rus (2014). Approximate bayesian computation by subset simulation. *SIAM Journal of Scientific Computing 36*, A1339–A1358.

David, R. and H. Alla (2010). *Discrete, Continuous, and Hybrid Petri Nets* (2 ed.). Springer-Verlag Berlin Heidelberg.

Desa, M. M. and E. Desa (2009). *Encyclopedia of Distances*. Springer-Verlag Berlin Heidelberg.

Geist, R., J. Hicks, and M. Smotherman (2005). Parallel simulation of petri nets on desktop pc hardware. In *Proceedings of the 37th Conference on Winter Simulation*, pp. 374–383. Winter Simulation Conference.

Holland, J. (1983). *Adaptation in Natural and Artificial Systems* (2 ed.). MIT Press.

Lee, B. L. H. (2014). Modelling railway bridge asset management. PhD Thesis, University of Nottingham.

Michel, J. M., P. Pudlo, C. Robert, and R. Ryder (2012). Approximate bayesian computation methods. *Statistics and Computing 22*, 1167–1180.

Molloy, M. K. (1982). Performance analysis using stochastic petri nets. *IEEE Transactions on Computers 31*, 913–917.

Murata, T. (1989). Petri nets: Properties, analysis and applications. *Proceesings of the IEEE 77*, 541–580.

Pham, D. and D. Karaboga (2000). *Intelligent Optimisation Techniques* (2 ed.). Springer-Verlag Berlin Heidelberg.

Santos, F. P., A. P.Teixeira, and C. Guedes Soares (2018). Modeling, simulation and optimization of maintenance cost aspects on multi-unit systems by stochastic petri nets with predicates. In *Simulation:Transactions of the Society for Modeling and Simulation International*, pp. 1–18.

Senderovich, A., A. Shleyfman, M. M. Weidlich, A. A. Gal, and A. Mandelbaum (2018). To aggregate or to eliminate? optimal model simplification for improved process performance prediction. *Information Systems 78*, 96–111.

Yang, C., R. Remenyte-Prescott, and J. Andrews (2015). Pavement maintenance scheduling using genetic algorithms. *International Journal of Performability Engineering 11*, 135–152.

Yianni, P. C. (2017). A modelling approach to railway bridge asset management. PhD Thesis, University of Nottingham.

Yianni, P. C., L. Neves, and J. Andrews (2018). Accelerating petri-net simulations using nvidia graphics processing units. *European Journal of Operational Research 265*, 361–371.