International Journal of Control and Automation
Vol.7, No.4 (2014), pp.165-174
http://dx.doi.org/10.14257/ijca.2014.7.4.15

Determining System Design Space Relative to its Architecture and Preventive Maintenance Policy

Shawulu Hunira Nggada

Department of Software Engineering,
School of Information Technology, Polytechnic of Namibia,
13Storch Street, Windhoek, Namibia

snggada@polytechnic.edu.na

Abstract

The search algorithm for optimal or near optimal solutions of preventive maintenance (PM) schedules performs the search within a confined design space. This space is an enclosure which consists of two layers; the outer and the inner layer. The outer layer is defined by set of system properties while the inner by combination of set of system properties and constraints imposed on the constituent components of the system. The total number of potential PM schedules exists within the outer layer while that of feasible PM schedules is within the inner layer. As the number of constraints increases, determining this number (in outer and inner layer) becomes more complex especially within the inner layer. A pre-knowledge of this number before optimising the system for PM schedules informs the system engineer about the size of the feasible region. This size could be used in predicting the amount of work in performing a search and also in other performance measures for a given PM optimisation problem. The calculation of the size within the feasible region is the focus in this paper.

Keywords: *Search space, preventive maintenance, optimization, feasible region, CoMI, architecture*

1. Introduction

A system design space refers to the number of potential design variants for the system. This number varies with variation in design constraints. The calculation of this number for any given system design depends on the approach employed in the design. In the context of system design optimisation, having a pre-knowledge of the number of design variants helps inform the system engineer about the design space that could be explored in the search for improved design. It could also be used in assessing the search coverage during an optimisation run. Thus for an exhaustive search, a pre-knowledge of the design space could be used as a parameter in estimating a complete search time.

Nggada *et al.* [1] established an approach for optimising preventive maintenance schedules of a system. The preventive maintenance schedule for each component could be expressed in terms of its coefficient of maintenance interval CoMI (described in Section 2.1). The CoMI for each component is determined by the constraints imposed on the optimisation. A more complex design constraint is further introduced in Nggada *et al.* [2] and includes substituting components while optimising preventive maintenance. The more the number or complexity of constraints, the more complex the optimisation becomes and the more complex it is to calculate the design space. There has been no established approach to calculate the design

ISSN: 2005-4297 IJCA

space for the preventive maintenance optimisations established by Nggada *et al.* [1, 2] and this is the focus of this paper considering the benefits of determining the design space.

The rest of this paper is structured as follows. Section 2 presents the concept of system design space while evaluation models for the design space are established in Section 3. Conclusions are drawn in Section 4.

2. System Design Space

Optimisation is concern with the search of improved or better solutions among numerous potential ones within a given boundary. The boundary is defined by the definition of the optimisation problem. The design space consists of all potential design solutions within the defined boundary. Figure 1 below similar to that which appeared in [3]illustrates this concept.

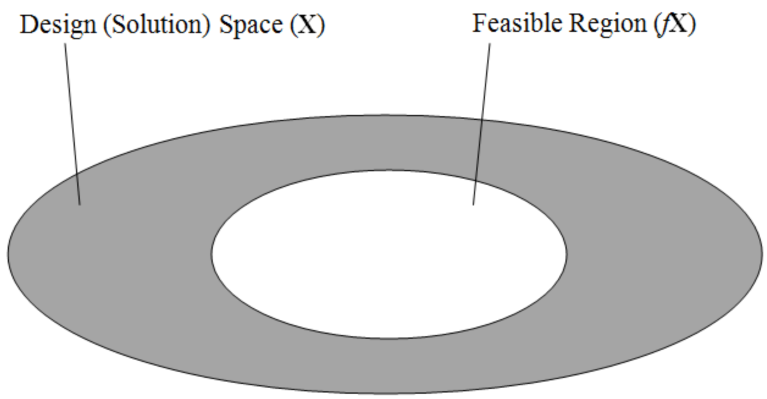

Figure 1. Solution Space and Feasible Region

The design space also referred to as solution space is denoted by **X**, while the feasible region by *f***X**. The feasible region is a boundary within the design space which consists of solutions that are potentially optimal. From the concept of Non-dominated Sorting Genetic Algorithm (NSGA) II [4, 5], the optimal solutions could actually exist at different fronts of the Pareto frontier. From Figure 1 it is trivial that *f***X** is a subset of **X**; *i.e.*, $f\mathbf{X} \subset \mathbf{X}$. Optimisation constraints guide and channel the optimisation to feasible solutions by concentrating within the feasible region. Therefore the modelling of the potential number of solutions in this paper is performed within the feasible region.

2.1. Coefficient of Maintenance Interval (CoMI)

Preventive maintenance is normally performed at periodic intervals. Assuming the established preventive maintenance policy states that maintenance for all components should be performed at an interval of at least T = 100 time units. This implies that for any given component C, its PM time T_p (the time at which maintenance actions are performed) could be 100 (1 x T), 200 (2 x T), 300 (3 x T), ..,t_p(n x T). Where t_p is the maximum potential PM time and n is the maximum multiple of T. It can be observed that the potential PM time for component C is in the form $T_p = \alpha T$, α is referred to as coefficient of maintenance interval [1]. Since T is a constant, the genetic encoding for preventive maintenance schedules can be represented by the CoMI. This implies that the potential preventive maintenance times for component C can be represented as {1, 2, 3, ..,n}. For any given component the maximum CoMI is obtained by equation 1 below [2].

$$n = \alpha_{max} = \begin{cases} Q\left(\frac{MTTF}{T}\right) & ;\ MTTF \leq RT \\ Q\left(\frac{RT}{T}\right) & ;\ MTTF > RT \end{cases} \quad (1)$$

Where: Q is the integer quotient of the division
RT is the system risk time, also referred to as useful life
MTTF is the mean time to failure for the component

Equation 1 above assumes a perfect preventive maintenance policy; for imperfect preventive maintenance policy *MTTF* is replaced with *MTBF* (mean time between failures).

3. Preventive Maintenance Design Space

Nggada *et al.* [1] performed a preventive maintenance optimisation which finds optimal PM schedules by considering range of CoMIs based on equation 1; which formed part of primary constraints. The primary constraints as established by Nggada *et al.* [1] ensures that (i) maintenance is not performed later than the component that fails most often in the system, and (ii) maintenance is not carried out too early thereby incurring unnecessary cost or too late when reliability of the component has dropped significantly. In a later work Nggada *et al.* [2] performed a more complex PM optimisation by combination PM and architecture (component substitution). Hence two forms of modelling for the number of potential solutions in the feasible region will be performed in this paper. Firstly, for the optimisationthat is based on primary constraints and secondly for component substitution. The component substitution in [2] forms part of secondary constraints.

3.1. Feasible Region Modelling under Primary Constraints

To be able to model the number of potential PM schedules, 3 cases of potential PM schedules are assumed and enumerated. The modelling of the feasible region uses the approach from first principle.

3.1.1. Case 1: Figure 2 below shows a system with 2 components each with a maximum CoMI of 2.

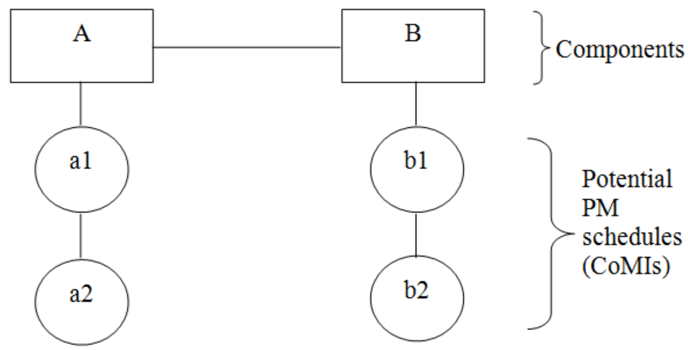

Figure 2. 2 component system with same number of respective potential PM schedules

To enumerate each potential PM schedule which implies the total number of potential PM schedules in Figure 2, the following procedure applies.

For each CoMI in component A Do
    For each CoMI in component B Do
    Add the CoMI of A and B to a new set

Using the procedure, the set of potential PM schedules (PMS1) is as follows.

$PMS_1 = \{ \{a1, b1\}, \{a1, b2\}, \{a2, b1\}, \{a2, b2\} \}$

In order to distinguish the feasible region under primary constraints and that under component substitution, $f\mathbf{X}_p$ will be used in this paper to represent the feasible region under primary constraints. Thus, the number of potential PM schedules in its feasible region will be represented by $\#(f\mathbf{X}_p)$. Hence the $\#(f\mathbf{X}_p)$ of Figure 2 as enumerated in PMS_1 is 4.

3.1.2. Case 2: Figure 3 below also shows a system with 2 components, component A has 2 CoMIs while B has 3. Its enumerated PM schedules are shown in PMS_2.

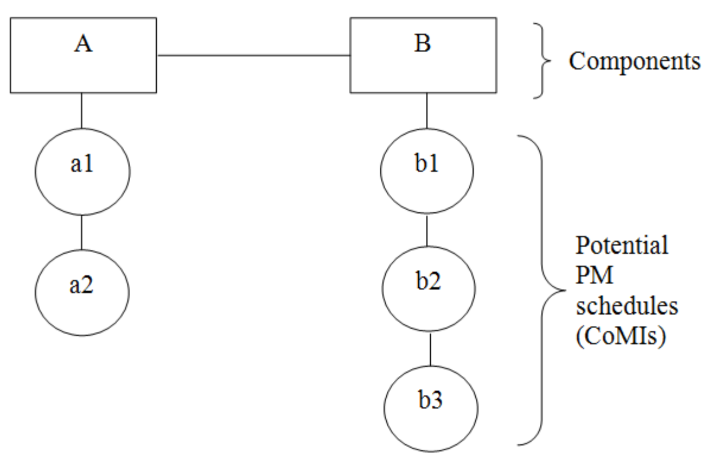

Figure 3. 2 component system with different number of respective potential PM schedules

PMS_2 = { {a1, b1}, {a1, b2}, {a1, b3}, {a2, b1}, {a2, b2}, {a2, b3} }

From PMS_2, $\#(f\mathbf{X}_p) = 6$.

3.1.3. Case 3: Figure 4 shows a system with 3 components. Each component has 3 CoMIs. The potential PM schedules are enumerated in PMS_3 below. The procedure used is illustrated below which is similar to the one that was used in cases 1 and 2.

For each CoMI in component A Do
For each CoMI in component B Do
For each CoMI in component D
Add the CoMI of A, B and D to a new set

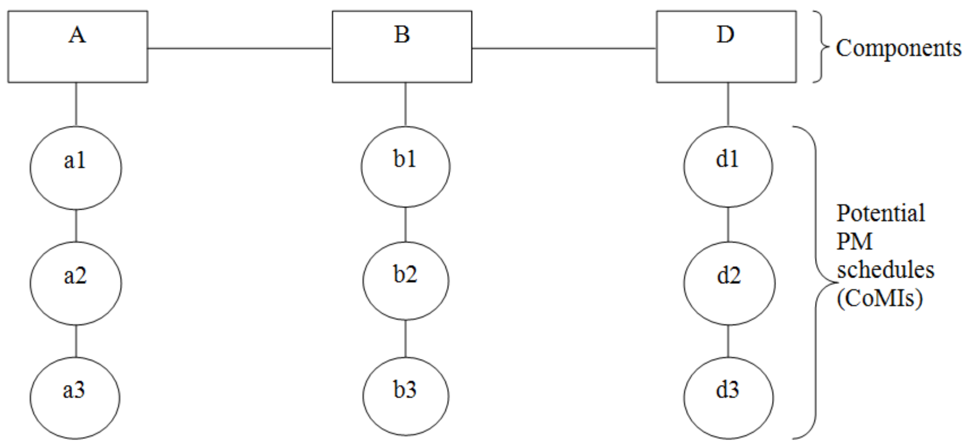

Figure 4. 3 component system with same number of respective potential PM schedules

PMS_3 = {a1, b1, d1}, {a1, b1, d2}, {a1, b1, d3}, {a1, b2, d1}, {a1, b2, d2}, {a1, b2, d3},
{a1, b3, d1}, {a1, b3, d2}, {a1, b3, d3},
{a2, b1, d1}, {a2, b1, d2}, {a2, b1, d3}, {a2, b2, d1}, {a2, b2, d2}, {a2, b2, d3},
{a2, b3, d1}, {a2, b3, d2}, {a2, b3, d3},
{a3, b1, d1}, {a3, b1, d2}, {a3, b1, d3}, {a3, b2, d1}, {a3, b2, d2}, {a3, b2, d3},
{a3, b3, d1}, {a3, b3, d2}, {a3, b3, d3} }

From PMS_3, $\#(f\mathbf{X}_p) = 27$.

3.1.4. General Representation: The summary of cases 1, 2 and 3 is shown in Table 1, where A_{CoMI}, B_{CoMI} and $D_{CoMI}$is the number of CoMIs for component A, B and D respectively.

Table 1. Summary of enumerated PM Schedules of cases 1, 2 and 3

Case	Components	Number of CoMIs	Number of Enumerated PM Schedules	Modelled Number of PM Schedules
1	A	$A_{CoMI} = 2$	4	A_{CoMI} x B_{CoMI}
	B	$B_{CoMI} = 2$		
2	A	$A_{CoMI} = 2$	6	A_{CoMI} x B_{CoMI}
	B	$B_{CoMI} = 3$		
3	A	$A_{CoMI} = 3$	27	A_{CoMI} x B_{CoMI} x D_{CoMI}
	B	$B_{CoMI} = 3$		
	D	$D_{CoMI} = 3$		

Hence by mathematical induction, Table 1 suggests that the mathematical modellingfor the number of potential PM schedules in a given feasible region $f\mathbf{X}_p$ is as shown in equation 2.

$$\#(f\boldsymbol{X_p}) = \prod_{i=1}^{m} \#(CoMI_i) \qquad (2)$$

Where: m is the number of components in the system
$\#(CoMI_i)$is the number of CoMIs for the *i-th* component

3.2. Feasible Region Modelling under Component Substitution

The component substitution allows for variants of the system architecture to be evaluated against set requirements. A combination of the primary and component substitution constraints implies that while architecture is being optimised, the preventive maintenance optimisation of the architecture is also taken into account for the overall evaluation of the system objectives [2]. Optimisation under primary constraints as considered in an earlier work [1] refers to only a single instance of the system architecture and the number of potential PM schedules for this instance is what is modelled in equation 2.

Hence having modelled the number of potential PM schedules for a given system model under primary constraints has simplified the complex nature of modelling such under the constraint of component substitution. To model the number of potential PM schedules under component substitution, the following task is performed.

(i) Model the number of variants of the system architecture

(ii) For each variant of the system architecture, the number of its potential PM schedules can be modelled using equation 2.

(iii) Model the number of potential PM schedules under component substitution as the summation of the potential PM schedules of each variant of the system architecture.

To model the number of variants of the system architecture, modelling from first principle is also applied under three cases.

3.2.1. Case 1: Let Figure 5 represent a system with 2 components. Figure 6 shows the same system but with its implementation options which when chosen will result into a variant of the system model.

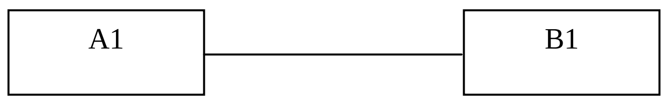

Figure 5. 2 components system

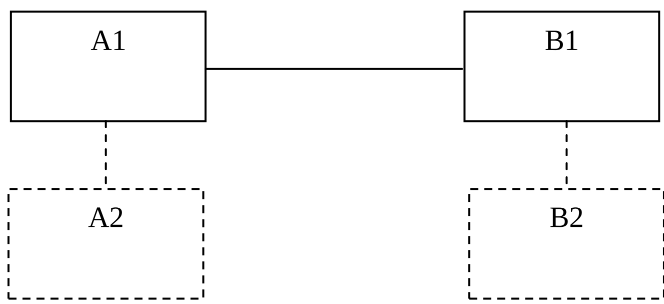

Figure 6. 2 component system showing their respective implementation options

Figure 5 shows that the system configuration is {A1, B1}, hence only one variant is possible. Figure 6 shows one additional implementation option for each of the components. Thus, an enumeration of the configurations is shown in C_1 (configurations for case 1).

$C_1 = \{ \{A1, B1\}, \{A1, B2\}, A2, B1\}, \{A2, B2\} \}$

Let $\#(V_c)$ denote the number of variants of the system model under component substitution. Then from C_1, $\#(V_c) = 4$.

3.2.2. Case 2: Figure 7 also represent a system with 2 components but with different number of implementation options.

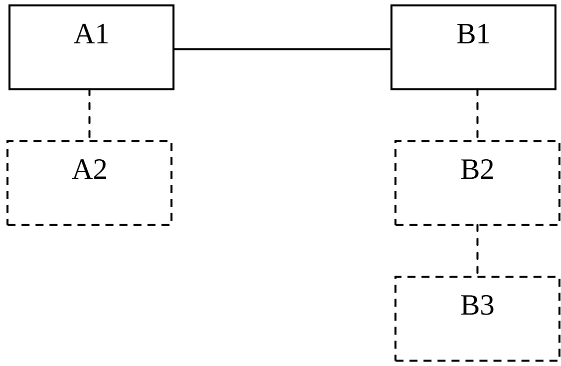

Figure 7. 2 component system each with different number of implementation options

The possible configurations of Figure 7 are shown in C_2.

C_2 = { {A1, B1}, {A1, B2}, {A1, B3}, {A2, B1}, {A2, B2}, {A2, B3} }

From C_2, $\#(V_c) = 6$.

3.2.3. Case 3: Figure 8 shows a system with 3 components and each with 3 number of implementation options.

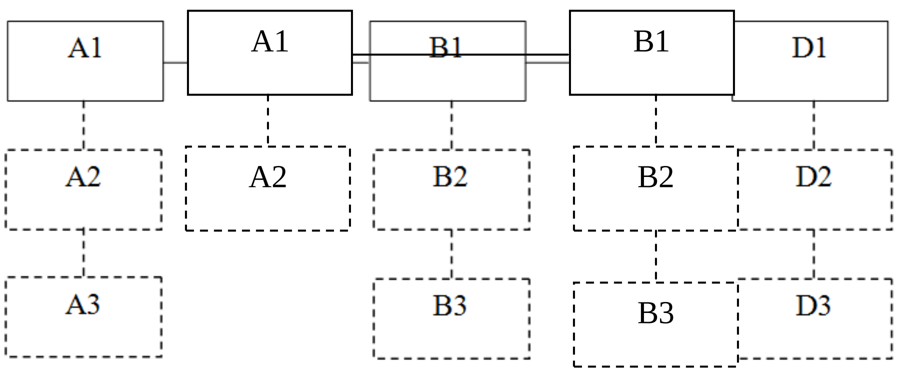

Figure 8. 3 component system with same number of implementation options

The possible configurations of Figure 8 are shown in C_3.

C_3 = { {A1, B1, D1}, {A1, B1, D2}, {A1, B1, D3}, {A1, B2, D1}, {A1, B2, D2}, {A1, B2, D3},
{A1, B3, D1}, {A1, B3, D2}, {A1, B3, D3},
{A2, B1, D1}, {A2, B1, D2}, {A2, B1, D3}, {A2, B2, D1}, {A2, B2, D2}, {A2, B2, D3},
{A2, B3, D1}, {A2, B3, D2}, {A2, B3, D3},
{A3, B1, D1}, {A3, B1, D2}, {A3, B1, D3}, {A3, B2, D1}, {A3, B2, D2}, {A3, B2, D3},
{A3, B3, D1}, {A3, B3, D2}, {A3, B3, D3}, }

From C_3, $\#(V_c) = 27$.

3.2.3. General Representation: The summary of cases 1, 2 and 3 is shown in Table 2. Where A_{IO}, B_{IO} and $D_{IO}$are the number of implementations of component A, B and D respectively.

Table 2. Summary of enumerated architecture configurations in cases 1, 2 and 3

Case	Components	Number of Implementation Options	Number of onfigurations	Modelled Number of Configurations (Number of Variants)
1	A	$A_{IO} = 2$	4	A_{IO} x B_{IO}
	B	$B_{IO} = 2$		
2	A	$A_{IO} = 2$	6	A_{IO} x B_{IO}
	B	$B_{IO} = 2$		
3	A	$A_{IO} = 2$	27	A_{IO} x B_{IO} x D_{IO}
	B	$B_{IO} = 2$		
	D	$D_{IO} = 3$		

Hence by mathematical induction, Table 2 suggests that the mathematical modelling of the number of variants of the system model is as shown in equation 3. This is similar to equation 2.

$$\#(\mathrm{V}) = \prod_{i=1}^{m} \#(IO_i) \tag{3}$$

Where: m is the number of components in the system
$\#(IO_i)$is the number of implementation options for the *i-th* component

A revisit to the identified procedure in section 3.2 required to model the number of potential PM schedules under component substitution, implies that task (i) is captured in equation 3. Task (ii) and (iii) can be capture in single model and are dependent on task (i); since task (i) models the number of variants of the system model for which the number of PM schedules for each is to be modelled. Hence equation 4 captures this scenario.

$$\#(f\boldsymbol{X_c}) = \sum_{j=1}^{\#(V_c)} \left(\prod_{i=1}^{m} \#(CoMI_{ji}) \right) \tag{4}$$

Where: $\#(V_c)$ is the number of variants of the system model as stipulated in equation 3
m is the number of components in the system
$\#(CoMI_{ji})$is the number of CoMIs for the *i-th* component of the *j-th* variant of the system model

4. Conclusions

Optimisation involves searching a space which contains potential solutions to a given problem. It usually defines objective functions for which a potential solution is evaluated against for possible consideration as a solution. Typically, constraints are also imposed to direct the search to potential solutions within the feasible region. Complex optimisation problems coupled with huge solution space could take longer to complete even with start-of-the-art infrastructure. It is therefore helpful to know the number of potential solutions within the feasible region prior to running the optimisation problem. This paper has addressed this problem within the areas of (i) optimising preventive maintenance schedules, and (ii) combined optimisation of preventive maintenance schedules and architecture. It focused on the proportional age reduction (PAR) model. Two respective models were developed which could be used to inform the system engineer of the scale of the problem in terms of search space to be covered. This could be compared against the actual space covered by the optimisation problem to know whether the feasible region has exhaustively been covered.

Further work will need to be carried out on modelling the number of potential PM schedules where the objective functions use other maintenance models other than the PAR.

References

[1] S. H. Nggada, D. J. Parker and Y. I. Papadopoulos, “Dynamic Effect of Perfect Preventive Maintenance on System Reliability and Cost Using HiP-HOPS”, IFAC-MCPL 2010 5th Conference on Management and Control of Production and Logistics, Coimbra, Portugal, **(2010)** September 8 – 10.

[2] S. H. Nggada, Y. I. Papadopoulos and D. J. Parker, “Combined Optimisation of System Architecture and Maintenance, 4th IFAC Workshop on Dependable Control of Discrete Systems, University of York, UK, **(2013)** September 4 – 6.

[3] S. H. Nggada, “Optimising Maintenance Intervals for a Component using a New Hill-Climbing Algorithm”, International Journal of Advanced Science and Technology, vol. 37, ISSN: 2005-4238**, (2011)**, pp. 1-14.

[4] K. Deb, A. Pratab, S. Agarwal and T. Meyarivan, “A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II”, IEEE Transactions on Evolutionary Computation, vol. 6, issue 2, **(2002)**, pp. 182-197.

[5] S. Favuzza, M. G. Ippolito and E. R. Sanseverino, “Crowded Comparison Operators for Constraints Handling in NSGA-II for Optimal Design of the Compensation System in Electrical Distribution Networks”, Advanced Engineering Informatics, vol. 20, issue 2, **(2006)**, pp. 201-211.

Author

Shawulu Hunira Nggada

Shwulu Hunira Nggada has PhD in Computer Science and PGDp in Research Training both from the University of Hull – UK, MSc Software Engineering from the University of Bradford – UK and B. Tech (Hons) in Computer Science from AbubakarTafawaBalewa University, Nigeria. He is a Charted IT Professional with BCS The Charted Institute for IT, Member Nigeria Computer Society and IEEE Reliability Society. He is currently a Senior Lecture in Software Engineering at the Polytechnic of Namibia (Namibia University of Science and Technology).