The Role of Expert Judgement in Optimising Preventive Maintenance and System Architecture

Shawulu Hunira Nggada
Department of Computer Science,
Polytechnic of Namibia,
Windhoek, Namibia
e-mail: snggada@polytechnic.edu.na

Yiannis Papadopoulos
Department of Computer Science
University of Hull,
Hull, United Kingdom
e-mail: Y.I.Papadopoulos@hull.ac.uk

***Abstract*-In the current industrial practice, optimisation of maintenance schedules is typically done using expert judgement but not via exhaustive exploration of all possible options for scheduling. Recently, it has been shown that search heuristics such as genetic algorithms can be used in conjunction with stochastic reliability prediction to optimise the maintenance schedules of components in a system. In this paper, we extend this framework to include the optional modelling of informed decisions by experts in terms of the time at which maintenance actions could be performed on components and decisions about which implementations of components should be in the final design. With this method, useful human knowledge and experience can be incorporated in a process that allows extensive exploration of the space of possible options for optimal or near optimal architecture and maintenance scheduling. The approach is demonstrated on a simplified model of fuel system.**

Keywords-maintenance; expert judgement; optimisation; genetic algorithm.

I. INTRODUCTION

Typically, during components inspection, more attention is given to components that have shown signs of poor condition. Such signs may be wear and tear, looseness, stiffness, low or high level of content, etc as it applies to the component in question. In a case where the component is viewed by maintenance personnel as repairable, such component undergoes maintenance actions otherwise replacement is performed. Preventive maintenance promotes carrying out maintenance intervention even before components show sign of poor condition. The time at which such intervention (maintenance actions) is performed is, however, difficult to determine.

Preventive Maintenance (PM) is normally performed periodically which implies a constant maintenance interval. The interval at which maintenance actions are performed on a given component is termed PM time T_p. Hence each component of the system will have a PM time T_{pi}, where *i* = *1..m*, *m* being the number of components of the system that have been identified for PM. Nggada et al. [1] used probabilistic method to determine the time at which a given component is maintained using the Proportional Age Reduction (PAR) model [2]. This method uses component failure data and ensures that the PM time T_{pi} for the *i-th* component is (i) not too early, incurring unnecessary cost, and (ii) not late when component reliability has significantly dropped. The probabilistic evaluation is proportional to the shortest PM time T of the system. The shortest PM time of the system is chosen such that T is less than the mean time to failure (or mean time between failures as appropriate) of the component that fails most often within the system. Hence each T_{pi} is a multiple of T as shown in Equation 1 [3].

$$T_{pi} = \alpha_i T \tag{1}$$

where: α_i is the coefficient of maintenance interval CoMI of the *i-th* component.

The CoMI α_i is an integer value ranging from 1 to α_{imax} where α_{imax} is obtained from Equation 2 below [1].

$$\alpha_{imax} = \begin{cases} Q\left(\frac{MTTF_i}{T}\right) & ;\ MTTF_i \leq RT \\ Q\left(\frac{RT}{T}\right) & ;\ MTTF_i > RT \end{cases} \tag{2}$$

where: Q is the integer quotient of the division;
RT is the system risk time, also referred to as useful life;
$MTTF_i$ is the mean time to failure for the *i-th* component;

However, a scenario may exist where the failure pattern of a given component becomes familiar over a long period of use under same condition. Another scenario that may exist is the lack of failure data which could be used to determine the PM time via probabilistic method. Under either or both scenarios, the use of expert opinion, which is informed by knowledge and experience becomes helpful.

Similarly, at the design stage of a system, each of its constituent components may have several options of its implementation. It is possible to consider all the implementation options of all the components at the system's design stage; giving rise to variants of the system. Nggada et al. [1] demonstrated how such system variants (architectures) could be optimised. The optimised set of the system variants consists of those implementation options that meet system requirements. Similar to the case of component failure, a better implementation option which should be included in the system design may be known to the expert.

 ISBN: 978-1-61208-340-7

The use of a stochastic method and expert opinion in deciding PM time and implementation option of selected components and the overall evaluation of the system model is the focus of this paper. Hence, the reminder of the paper is structured as follows. Section II discusses expert judgement in PM time while Section III discusses expert judgement in implementation option. Section IV discusses the effect of expert judgement on the system and maintenance scheduling optimisation algorithm and process. The modelling and system optimisation process have been implemented in HiP-HOPS, a state-of-the-art tool [11] that is the result of more than fifteen years of research on model-based system dependability analysis and architecture and maintenance optimisation. A case study on maintenance scheduling using this tool is presented in Section V. Evaluations are presented in Section VI, while conclusions are drawn in Section VII.

II. EXPERT JUGDEMENT IN MAINTENANCE TIME

Expert judgement as defined in this paper refers to the elicitation of informed opinions from persons with particular expertise. Expert judgement has been applied in several areas. For instance, expert judgement has been utilised in specifying the number of failures for a component within time interval. The elicitation of such lifetime data from several experts are combined into a consensus distribution which is then updated with failure data. Such combination is done through defined procedures [4]. Another area where expert judgement has been used is reliability prediction in early stages of product development process. Elicitation of expert opinion on lower bound (belief) and upper bound (plausibility) of failure time interval is performed [5]. An increased use of risk assessment in organisations has also increased the role of expert judgement in providing information for safety related decision making. Under such a decision making, expert judgement is required in most of the steps of risk assessment, for instance hazard identification, risk estimation, risk evaluation and analysis options [6].

Under preventive maintenance as used in this paper, expert opinion is used in determining the regular time interval for which a component is to be maintained. In this paper, the time specified by an expert at which a given component is to be maintained is referred to as Expert PM time ($EPMT$). In complying with the expert judgement it is considered appropriate for the PM time of the component to be less or equal to its $EPMT$; $T_{pi} \leq EPMT_i$. The rationale for this is straightforward: so that the likelihood of the component to experiencing unplanned maintenance is minimised. The definition is thus as shown in Equation 3.

$$\left(T_{pi} \leqslant \begin{cases} EPMT_i & ; EPMT_i \leq RT \\ RT & ;\ ET_i > RT \end{cases}\right) \tag{3}$$

where: $EPMT_i$ is the expert PM time specified for the *i-th* component

In order to follow a similar pattern of evaluation as for components with non-expert specified PM times, the maximum CoMI for the *i-th* component under expert judgement is obtained as shown in Equation 4.

$$\alpha_{iemax} = \begin{cases} Q\left(\dfrac{EPMT_i}{T}\right) & ;\ EPMT_i \leq RT \\ Q\left(\dfrac{RT}{T}\right) & ;\ EPMT_i > RT \end{cases} \tag{4}$$

where: α_{iemax} is the maximum CoMI of the *i-th* component under expert judgement.

Therefore, the PM time of a component under expert judgement is evaluated similar to Equation 1 shown in Equation 5.

$$T_{pi} = \alpha_{iemax} T \tag{5}$$

III. EXPERT JUDGEMENT IN IMPLEMENTATION OPTION

Assume the following sub-system with two components shown in Fig. 1. X_1 and Y_1 are the implementations of their respective component types X and Y. Fig. 2 shows a variant of Fig. 1 where each component type consists of 3 implementation options.

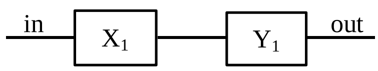

Figure 1. Two components sub-system

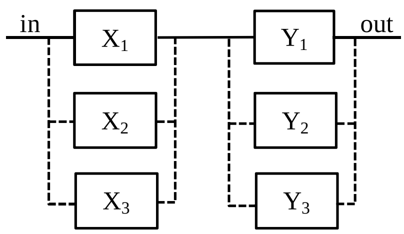

Figure 2. Two sub-system with implementation options

In Fig. 1, X = {X_1} and Y = {Y_1} while in Fig. 2, X = {X_1, X_2, X_3} and Y = {Y_1, Y_2, Y_3}. In Fig. 2 X_1 is the active implementation of component X and similarly Y_1 is the active implementation of component Y. The set of components {X_2, X_3} and {Y_2, Y_3} are the alternative implementations of components X and Y respectively. Fig. 2 implies that the sub-system is a possible combination of any of the implementation options in X and Y. Thus, if the sub-system is represented by S_{sub}, then the potential design models of the sub-system are shown below.

$$\begin{aligned} S_{sub} = \{ & \{X_1, Y_1\}, \{X_1, Y_2\}, \{X_1, Y_3\}, \\ & \{X_2, Y_1\}, \{X_2, Y_2\}, \{X_2, Y_3\}, \\ & \{X_3, Y_1\}, \{X_3, Y_2\}, \{X_3, Y_3\} \} \end{aligned}$$

Each of the subsets of potential sub-system design models is referred to as a variant of the sub-system. This scenario also applies to a full system model. At the infancy design stage of a system, as illustrated by Nggada et al. [1], a stochastic process could be used in determining the set of variants which meet design requirements. The engineer could then select one (or more as appropriate) of these variants which will be implemented.

In certain scenarios, the engineer would like to specify which components are to be included in the system model. Such action is an expert opinion that is informed by the engineer's knowledge and, or experience of the system over time. Hence, in addition to stochastically determining the implementation option of a given component type it is also helpful to provide the engineer with an option to specify which of the implementation options should be included in the system model. To achieve this, the engineer would simply select the active implementation of the component while the alternative options are excluded by using a flag.

IV. Effect of Expert Judgement on Optimisation

An optimisation problem could have single or multi objective, depending on the problem and the approach to optimising the solutions. The work in this paper is multi-objective in nature and maintains a multi-objective approach to optimising the solutions. In general a multi-objective optimisation problem is defined as follows [7][8].

max $\mathbf{F}(x) = \{ f_1(x), f_2(x), f_3(x), .., f_{z-1}(x), f_z(x) \}$
such that:
$x \in \mathbf{X}$
$g_j(x) \leqslant b_j$; $j = 1..k$

where: $f_1, f_2, f_3, .., f_{z-1}, f_z$ are objective functions.
$\mathbf{X}$ is the solution space of all potential solutions.
$x = (x_1, x_2, .., x_{m-1}, x_m)$. In genetic algorithm terminology x is referred to as decision variable vector, while each x_i is referred to as decision variable.
$g_j(x) \leqslant b_j$ is referred to as constraint, where k is the number of constraints imposed on the optimisation.

The left hand side of the constraint; $g_j(x)$ is a real value function, whereas b_j could either be a predefined value or the result of another real value function. $\mathbf{F}(x)$ is referred to as decision vector. The goal of the optimisation problem could be either maximisation (max) or minimisation (min) of the decision vector. The decision vector consists of objective functions as seen in the definition. The objective functions are attributes of the system design and normally include cost and one or more of the following: reliability, availability, safety, weight, etc. The equation $g_j(x) \leqslant b_j$ is known as the inequality constraint. If this is in the form $g_j(x) = b_j$, then it is referred to as equality constraint [8]. When a constraint is present, the optimisation must conform to it. A solution $x \in \mathbf{X}$ which satisfies the constraints is said to be a feasible solution. A collection of all potential feasible solutions defines the feasible region.

Hence, to define the PM optimisation in this paper, the effect of expert judgement will be considered as constraints to the optimisation. The constraints are defined in the next section.

A. Constraints of the optimisation

The constraints guide search algorithm towards the feasible region. When an expert specifies PM time or what implementation option is active and excludes the rest in the optimisation, the size of the feasible region is altered. The size of the feasible region is resizable and without any defined constraint the size is same as the solution space. Firstly, the constraint under PM time is defined followed by component substitution.

B. PM Time Constraints

Constraints guide the selection of individuals within feasible region, where solutions meeting design requirements exit. The constraint of PM time under expert judgement is simply a modification of Equation 3. The time at which maintenance actions are performed when an expert specifies time is the product of the maximum CoMI and the system's shortest PM interval. Thus, the following constraint applies.

$$\left(T_{pi} \leqslant \begin{cases} EPMT_i & ; EPMT_i \leq RT \\ RT & ; EPMT_i > RT \end{cases} \right) \Leftrightarrow \big((expert\_judgement_i = true) \wedge (EPMT_i \geq T) \big) \quad (C1)$$

where: *expert_judgement$_i$* is a Boolean variable that is flagged true for a component that is identified for expert judgement and false otherwise.

When an expert specifies a PM time, Equation 3 ensures that this time is a multiple of T, and if not then converted to such. Additionally constraint $C1$ is enforced if the expert PM time is not less than the system's shortest PM interval. Constraint $C1$ only applies to components that are subjected to expert judgement. Thus for other components the constraint is same as those defined in [10] also shown below.

$$\mathrm{T} < \frac{1}{\lambda_H} \quad (C2)$$

$$\alpha_i \mathrm{T} \leqslant \frac{1}{\lambda_i} \quad (C3)$$

Where: λ_H is the failure rate of the component that fails most frequently.
λ_i is the failure rate of the *i-th* component.

Constraint $C2$ implies that the shortest PM interval T must be smaller than the mean time to failure (MTTF) of the component that fails most often in the system. The second constraint defines that for every component *i*, its PM interval must be smaller than its MTTF. These two constraints ensure that maintenance is effective and is not scheduled too late when the reliability of components has dropped too much.

C. Component Substitution Constraint

Component substitution refers to the process of replacing the current active implementation with one of its alternative options. The role of expert judgement under component substitution is to select and to specify the active implementation of the component, and disable such substitution. The constraint used by Nggada et al. [1], which was implemented in HiP-HOPS, could be reused to achieve this, and is as shown below.

$$substitute\_component(i, k_i) \Leftrightarrow \begin{pmatrix} (substitute_i = true) \\ \wedge \\ (k_i > 0) \end{pmatrix} \qquad (C4)$$

Component substitution is performed by a function called $substitute\_component$ and has two parameters. The first is the index *i* of the component under consideration, and then a second index k_i of its current active implementation. The details of the function are contained in Nggada et al. [1]. Constraint $C4$ implies that a component is substituted if and only if the *i-th* component's Boolean parameter $substitute_i$ is flagged $true$ and that there exist at least one alternative implementation option. This therefore entails that during component failure annotation in HiP-HOPS the engineer would simply disable substitution for the *i-th* component once an active implementation is selected. Hence for a component which the expert has selected to be fixed throughout the optimisation, the $substitute_i = true$ is replaced with $substitute_i = false$.

D. Defined PM Optimisation

Having defined the optimisation constraints, the optimisation is, therefore, defined as follows.

$min\mathbf{F}(\boldsymbol{\alpha}) = \{ U(\boldsymbol{\alpha}), C(\boldsymbol{\alpha}) \}$

such that: $\boldsymbol{\alpha} \in \mathbf{A}$, $C1$, $C2$, $C3$, $C4$.

Where: $\boldsymbol{\alpha}$ is a decision variable vector consisting of CoMIs of constituent components of the system.
A is the PM solution space
$C1$, $C3$, $C3$ and $C4$ are the defined constraints.
U and C are the objective functions; unavailability and cost respectively.
The goal of the optimisation is to minimise the objective functions.

V. CASE STUDY

The case study used in Nggada et al. [1] is adopted on which the defined PM optimisation problem is evaluated. The case study is a simplified model of the fuel oil service system (FOSS) which supplies fuel to the main engine of a ship. The FOSS is shown in Fig. 3 and its description is same as in [1]. The system incorporates a service tank which contains stored fuel oil. The booster pump conveys fuel oil to the mixing tank through a filter and flow meter. If the pressure level in the mixing tank exceeds a threshold level, fuel oil is released back into the service tank through a pipe connecting the two. The circulation pump then conveys fuel oil to the main engine through a heater, viscosity meter and a filter. Excess fuel oil not used in the main engine is released to the service tank via the mixing tank.

In order to analyse the model of the fuel oil service system, its constituent components were annotated with HiP-HOPS failure behaviour data. Due to space limitation, a detailed presentation of the annotations is impossible, however the component failure behaviour is simple; each component has a single failure mode which causes omission of outputs while input failures propagate to the outputs of the components.

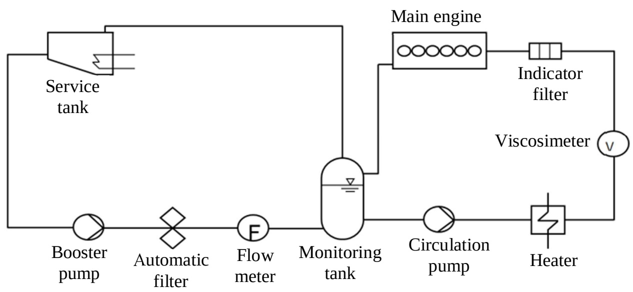

Figure 3. Fuel oil service system (FOSS)

Similarly due to space limitation, short names are also used to represent the actual component names for the FOSS. These short names are as shown in Table I.

TABLE I. COMPONENTS AND THEIR RESPECTIVE SHORT NAMES

Component	Short Name
Automatic filter	af
Booster pump	bp
Circulation pump	cp
Flow meter	fm
Heater	ht
Indicator filter	if
Main engine	me
Mixing tank	mt
Service tank	st
Viscosimeter	vm

Table II shows the components that were subjected to expert judgement on PM time. The table also shows the corresponding EPMTs.

TABLE II. COMPONENTS AND EXPERT PM TIMES

Components	Expert PM Time
Main engine	1500
Service tank	1260
Viscosimeter	870

All the components have implementation options; however, those that were subjected to component substitution are shown in Table III.

TABLE III. COMPONENTS, IMPLEMENTATIONS AND SHORT NAMES

Component	Implementations	Short Name
Heater	Heater_1	ht_1
	Heater_2	ht_2
	Heater_3	ht_3
	Heater_4	ht_4
Mixing tank	Mixing_tank_1	mt_1
	Mixing_tank_2	mt_2
	Mixing_tank_3	mt_3
	Mixing_tank_4	mt_4
Flow meter	Flow_meter_1	fm_1
	Flow_meter_2	fm_2
	Flow_meter_3	fm_3
	Flow_meter_4	fm_4
	Flow_meter_5	fm_5

The components which the expert specified as fixed throughout the optimisation are shown in Table IV.

TABLE IV – FIXED COMPONENTS THROUGHOUT THE OPTIMISATION

Component	Short Name
Automatic filter	af
Booster pump	bp
Circulation pump	cp
Indicator filter	if

VI. EVALUATIONS

In order to evaluate the defined PM optimisation on the case study, the evaluation models for the objective functions need to be defined. Similarly the optimisation algorithm that would incorporate the constraints needs to be defined. These definitions are discussed next.

A. Objective Functions Model

The maintenance policy assumed in this paper is perfect preventive maintenance and therefore same evaluation models found in [1] is used. Equation 6 is used to evaluate component reliability where its failure characteristic is assumed to follow the Weibull distribution [1].

$$U_{pc}(t) = 1 - \ exp\left[-n\left(\frac{T_p}{\Theta}\right)^{\beta}\right] exp\left[-\left(\frac{t - nT_p}{\Theta}\right)^{\beta}\right] \quad (6)$$

$$nT_p \leqslant t \leqslant (n+1)T_p$$

where: U_{pc} is component unavailability under PPM
t is the age of the component
n is the number of PM stages
β is the Weibull shape parameter
θ is the Weibull scale parameter

The system unavailability (U) is evaluated using the Esary-Proschan approximation [12]. The total PM cost of the system is the summation of the individual components' cost as shown in Equation 7.

$$C = \sum_{i}^{m} \left(n_i C_{ppmi} + C_{ci}\right) \quad (7)$$

Where: m is the number of system components
C is the system cost under PPM
C_{ppmi} is the cost of performing PPM for the *i-th* component
C_{ci} is the unit cost of the *i-th* component
n_i is the total number of PM stages for the *i-th* component, n_i is evaluated using Equation 8

The number of PM stages n for each component is evaluated using Equation 8.

$$n = \begin{cases} Q\left(\frac{MTTF}{T_p}\right) & ; \ MTTF \le RT \\ Q\left(\frac{RT}{T_p}\right) & ; \ MTTF \ \ge \ RT \end{cases} \quad (8)$$

Additionally, the following parameter values were assumed.

Weibull shape parameter β = 2
Weibull scale parameter θ = 1500
FOSS shortest PM interval T = 180
Maintenance improvement factor f = 0.875
Maximum optimization generation = 5120

The improvement factor is simply the effectiveness of the maintenance action. The details of which could be found in Nggada et al. [10].

B. Optimisation Algorithm

To optimise the PM schedules of the FOSS, a variant of the Non-dominated Sorting Genetic Algorithm (NSGA) II [13] is developed. It takes into account the defined constraints and objective functions. The mechanics of the adapted algorithm using HiP-HOPS are here discussed. The algorithm first generates a random initial population P of N number of PM individuals, with each individual represented as *p*. The following steps are then executed:

 ISBN: 978-1-61208-340-7

1. Set population index t = 1.
2. Set front index i = 1.
3. Randomly generate an initial population P_t of N number of PM individuals. This is performed in any of two steps as follows (1) If a given component *i* qualifies for component substitution, then $\alpha_{i,k_i} = substitute\_component(i, k_i)$, (2) if the component qualifies for expert judgement then $\alpha_{i,k_i} = \alpha_{ie,k_i}$ else $\alpha_{i,k_i} = random(1..\alpha_{imax,k_i})$.
4. $\forall p \in P_t$, configure the variant of the system model with *p* by using the encoding to set the CoMI of each component and then evaluate the unavailability and cost (objective functions) of the system by calling the automatic fault tree synthesis and analysis functions of HiP-HOPS.
5. $\forall p \in P$, find n_p number of solutions that dominate *p*, and S_p set of solutions for which *p* dominates.
6. Add all *p* with $n_p = 0$ into the set F_i (the i-th front) and assign domination rank $R_p = i$.
7. For each $p \in F_i$ assign crowding distance to *p*.
8. Increment front index by 1; i.e. i = i + 1.
9. For each $p \in F_{i-1}$, visit each $q \in S_p$ and decrement n_q by 1, if by doing so, n_q becomes 0 then add *q* into the set F_i (*q* belonging to front i, $R_q = i$).
10. Repeat step 8 to find subsequent fronts.
11. Perform recombination as follows (“a – j” below)

(a) Set child population $Q_t = \emptyset$.

(b) Use binary tournament selection to select two parents from population P_t.

(c) With probability P_c, perform uniform crossover on the selected parents to evolve with a child p.

(d) With probability P_m, perform mutation in one of the following ways; (1) if the selected locus *i* corresponds to a component that has been flagged for expert judgement (i.e. *expert_judgement*$_i$ = *true*) and $ET_i \geq T$ then exit to step “e” below, else (2) perform normal mutation.

(e) Add p to Q_t; i.e. $Q_t = Q_t \cup p$.

(f) If the size of Q_t is not equal to N, then go to step “b”.

(g) $\forall p \in Q_t$, configure the variant of the system model with p. The values of objective functions (unavailability and cost) are also calculated.

(h) P_t and Q_t are combined into B_t; i.e. $B_t = P_t \cup Q_t$ and B_t is sorted based on non-domination.

(i) From 2N solutions (combination of P_t and Q_t) in B_t, N best solutions are selected using the crowding calculation and comparison to form P_{t+1}.

(j) Increment population index by 1; i.e. t = t + 1.

12. If maximum generation is not reached then go to step 4 else terminate giving the set of PM individuals in the first front F_1 as the solution.

C. Results

The Pareto frontier of the optimisation is shown in Fig. 4. A total of 206 optimal PPM schedules were found, with the last found in generation 1722. For the components subjected to substitution Heater_2, Mixing_tank_2 and Flow_meter_3 dominated the optimal solutions. The result indicates that an engineer could choose an optimal design option relative to cost and unavailability requirements. Typically, the optimisation is done manually, which, therefore presents only fewer options.

Table V shows the first and last 5 out of the 206 PPM schedules. It shows that the components Main_engine, Service_tank and Viscosimeter subjected to expert judgement have fixed CoMIs in all the optimal PPM schedules. Similarly none of the alternative options of the components that were not subjected to component substitution appears in the optimal set.

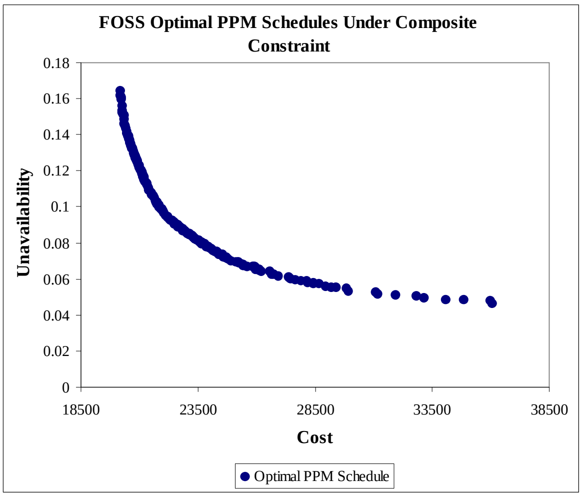

Figure 4. Pareto frontier of FOSS PPM schedules

TABLE V. A SUBSET OF OPTIMAL PPM SCHEDULES; A TABULAR REPRESENTATION

Optimal PPM Schedule	Cost	Unavailability	Generation
af(8) bp(4) cp(6) ft.ft_3(4) ht.ht_2(1) if(3) me(8) mt.mt_2(5) st(7) vm(4)	21866	0.099808	26
af(8) bp(4) cp(6) ft.ft_3(4) ht.ht_2(3) if(3) me(8) mt.mt_2(5) st(7) vm(4)	21746	0.102454	26
af(6) bp(4) cp(6) ft.ft_3(4) ht.ht_2(2) if(3) me(8) mt.mt_2(4) st(7) vm(4)	22217	0.094253	29
af(5) bp(3) cp(6) ft.ft_3(4) ht.ht_2(2) if(3) me(8) mt.mt_2(4) st(7) vm(4)	22678	0.089516	29
af(6) bp(3) cp(6) ft.ft_3(4) ht.ht_2(2) if(3) me(8) mt.mt_2(5) st(7) vm(4)	22237	0.093825	30

af(6) bp(4) cp(6) ft.ft_3(4) ht.ht_2(1) if(3) me(8) mt.mt_2(4) st(7) vm(4)	22307	0.092919	1439
af(1) bp(1) cp(1) ft.ft_3(1) ht.ht_2(1) if(1) me(8) mt.mt_2(2) st(7) vm(4)	34859	0.048226	1453
af(8) bp(4) cp(6) ft.ft_3(4) ht.ht_2(2) if(3) me(8) mt.mt_2(8) st(7) vm(4)	21600	0.105751	1463
af(1) bp(1) cp(1) ft.ft_3(1) ht.ht_2(2) if(1) me(8) mt.mt_2(1) st(7) vm(4)	36001	0.047708	1631
af(8) bp(3) cp(6) ft.ft_3(4) ht.ht_2(2) if(3) me(8) mt.mt_2(5) st(7) vm(4)	21972	0.098662	1722

VII. CONCLUSION AND FUTURE WORK

In the design of engineering systems it is generally helpful to enable systematic and automated exploration of design options using heuristics whilst maintaining the possibility of certain decisions to be taken by informed expert opinion. This paper has illustrated an approach to this in which expert judgement can be integrated in system architecture and maintenance optimisation method where optimisation is driven by dependability and cost. Constraints to represent expert judgement on maintenance time and selection of components were developed and a variant of the NSGA II was adapted within the HiP-HOPS tool to enable the proposed approach. Initial results suggest that the approach is valid and promising. The method is currently being extended to enable more sophisticated ways for incorporating important maintenance constraints related to the geometry and topology of the system.

REFERENCES

[1] 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, 4- 6 September 2013, University of York - UK, IFAC Proceedings Volumes (IFAC-PapersOnline), vol. 4, part 1, pp. 25-30, doi: 10.3182/20130904-3-UK-4041.00026

[2] M. Shafiee, and M. J. Zuo, "Adapting an age-reduction model to extend the useful-life duration," International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering (ICQR2MSE), 15-18 June 2012, pp. 1055-1059, doi: 10.1109/ICQR2MSE.2012.6246403

[3] S. H. Nggada, Y. I. Papadopoulos, and D. J. Parker, “Extending HiP-HOPS with capabilities of planning preventative maintenance, in Strategic Advantage of Computing Information Systems in Enterprise Management, M. Sarrafzadeh, Ed, pp. 231-242, ISBN: 978-960-6672-93-4

[4] J. M. van Noortwijk, R. Dekker, R. M. Cooke, and T. A. Mazzuchi, “Expert jugement in maintenance optimization,” IEEE Transactions on Reliability, vol. 41, no. 3, pp. 427-432, 1992

[5] A. Mannhart, A. Bilgic, and B. Bertsche, "Modeling expert judgement for reliability prediction - comparison of methods," Reliability and Maintainability Symposium, RAMS '07, 22-25 January 2007, pp. 1-6, doi: 10.1109/RAMS.2007.328099

[6] T. Rosqvist, “On the use of expert judgement in the qualifcation of risk assessment,” Helsinki University of Technology, PhD Thesis, Finland, 2003.

[7] H. Huang, Z. Tian, and M. J. Zuo, "Intelligent interactive multiobjective optimization method and its application to reliability optimization," IIE Transactions, vol. 37, issue 11, pp. 983-993, 2005

[8] A. Konak, D. W. Coit, and A. E. Smith, "Multi-objective optimization using genetic algorithms: a tutorial," Reliability Engineering and System Safety, vol. 91, issue 9, pp. 992-1007, 2006

[9] M. Gen, and R. Cheng, "Genetic algorithms and engineering design," New York, John Wiley & Sons, 1997

[10] 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, September 2010, Coimbra – Portugal, IFAC Proceedings Volumes (IFAC-PapersOnline), pp. 204-209, ISSN: 14746670

[11] Y. I. Papadopoulos, and J. A. McDermid, “Hierarchically performed hazard origin and propagation studies,” In: 18th International Conference in Computer Safety, Reliability and Security, Toulouse, France, pp. 139-152, 1999

[12] T. Jin, and D. W. Coit, “Approximating network reliability estimates using linear and quadratic unreliability of minimal cuts,” Reliability Engineering and System Safety, vol. 82, issue 1, pp. 41-48, 2003

[13] 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, pp. 182-197, 2002