NON-PERIODIC INSPECTION OPTIMIZATION OF REPAIRABLE SYSTEMS

By

Yassin Hajipour, B.Sc., Ryerson University, 2014

A thesis presented to Ryerson University in partial

fulfillment of the requirements for the degree of

Master of Applied Science

in the program of

Mechanical and Industrial Engineering

Toronto, Ontario, Canada, 2016

AUTHOR'S DECLARATION FOR ELECTRONIC SUBMISSION OF A THESIS

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final version, as accepted by my examiners.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my thesis may be made electronically available to the public.

ABSTARCT

NON-PERIODIC INSPECTION OPTIMIZATION OF REPAIRABLE SYSTEMS

Yassin Hajipour
Master of Applied Science
Mechanical and Industrial Engineering, 2016
Ryerson University, Toronto, ON, M5B 2K3, CANADA

This study proposes models to find the optimal non-periodic inspection interval over a finite planning horizon for two types of multi-component repairable systems. The first system consists of hard-type and soft-type components, and the second system is a k-out-of-m system with m identical components. The failures of components in both systems follow a non-homogeneous Poisson process. The failure of soft-type components and the failure of components in a k-out-of-m system when the number of failed components is still less than m-k+1, are soft failures. Soft failures are revealed only at scheduled inspections or when an event of opportunistic inspection or a system failure occurs. The failures of hard-type components or the failure of $(m-k+1)^{th}$ failed component in a k-out-of-m system are hard failures, and cause the system to stop functioning. Hard failures are revealed immediately and the failed components are fixed. In this study, a failed component is either replaced or minimally repaired according to its age at failure time. To find the optimal inspection schedules for the systems, we minimize the total expected cost of the systems over a finite planning horizon. The total cost for the first type of system includes the costs of components' minimal repairs, replacements, downtimes, and the scheduled inspections. The total cost of a k-out-of-m system has an additional penalty cost for system failures. We consider a binary variable for a possible

scheduled inspection's time, in which 1 indicates performing a planned inspection at that time, and 0 shows no inspection to be performed. Thus, our goal is to find the optimal vector of binary decision variables which results in the minimum total cost of the system. A recursive formula is developed to calculate the expected number of minimal repairs, replacements and downtime of soft-type components. However since obtaining the expected values from the mathematical formula is cumbersome, we develop a simulation model to obtain the total expected cost for a given non-periodic inspection scheme. We then integrate the simulation model with a genetic algorithm to obtain the optimal inspection scheme.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my supervisor, Dr. Sharareh Taghipour for her remarkable support, encouragement and guidance and introducing me to Industrial Engineering and inspection optimization. Her enthusiasm into my research area has always inspired me and her willingness to share her expertise has helped me progress in my research. This thesis would not have been completed without her. Words are inadequate to express my special appreciation to Dr. Sharareh Taghipour, who helped me patiently with her excellent knowledge. Her enthusiasm and encouragement made me eager to succeed. I am also grateful for her critical reviews on this thesis.

I would also like to acknowledge the Ryerson University and the Department of Mechanical and Industrial Engineering and also graduate program director, Dr. Ahmad Ghasempoor for support and guidance.

Also, thanks to Natural Sciences and Engineering Research Council of Canada (NSERC) who has financially supported this research.

I gratefully thank Dr. Mohamad Ismail Wahab and Dr. Cory Searcy from Mechanical and Industrial Engineering Department of Ryerson University who served as the members of my research committee, and provided me with valuable comments on and insights into both the research and the presentation of the thesis.

I would like to offer my sincere gratitude to my lovely parents who have always been a source of encouragement and support in my life.

PREFACE

Chapters 2, 3 and 4 of this thesis are the extensions of the following journal and conference papers published/submitted during my Masters studies:

Journal Paper:

Y. Hajipour, S. Taghipour, "Non-Periodic Inspection Optimization of Multi-Component and k-out-of-n Systems Using Genetic Algorithm," Reliability Engineering & System Safety, under second round of review after revision.

Conference Proceedings:

Y. Hajipour, S. Taghipour, "Non-Periodic Inspection Optimization of Multi-Component System Using Genetic Algorithm," Proceedings of the IIE Annual Conference and Expo 2015 Nashville, Tennessee.

Table of Contents

List of Tables

Chapter 3

Chapter 4

List of Figures

Chapter 2

Chapter 3

Chapter 4

Chapter 1 - Introduction and the Literature Review

There are many multi-component systems in various fields of industry and human life. The significant part of capital equipment investment is usually represented by multi-component systems. They are one of the most important and complex categories of assets, in the sense that the availability and reliability of such equipment could impact the prosperity of business in different industry zones including manufacturing, services, and healthcare. Maintenance of multi-component systems is increasingly important, since the failure of a system may result in catastrophic consequences, incur huge costs and present threat to human life.

The majority of multi-component systems is either designed, or assumed to be repairable, since it is usually not economical to replace failed components instead of repairing them [1]. Systems inspection is a common maintenance strategy that many organizations implement, particularly when their equipment is complex and is subject to hidden failures. Up to 40% of failures in industrial systems are hidden failures [2]. Inspection helps to retain the uptime and availability of systems as high as possible [3, 4]. For example, computer network servers, backup power generation systems, and medical devices are required to have a maximum possible availability and uptime to be able to minimize the excessive costs and undesirable consequences.

Inspection policy could be categorized into periodic or non-periodic inspections. The components of multi-component systems are classified by type, or mode of failure. A complex

system consists of two types of components, soft-type and hard-type. Failure in these components are classified as soft (or hidden) and hard failures [5, 6]. A component which is subject to hard failures is called a hard-type component whereas a soft-type component is the one subject to soft failure. In certain systems, some components can be subject to both soft and hard failures, and therefore, they are treated as soft-type and hard-type components, separately.

The entire system stops functioning in an event of hard failure. Hence, hard failures are self-announcing and the time of failure is known. An example of a hard-type components is central processing unit (CPU) in personal computers, since the failure of CPU results in immediate dysfunction of the computer. The power outage in a computer circuit is another example of hard-type components.

Soft failures, however, are not self-announcing and are hidden. They can only be detected and rectified at inspections [1, 7]. Soft failures typically do not result in the failure of the entire system, but likely cause a decrease in the system's reliability by eliminating redundancy and increasing the risk of breakdown or future damage. The time that soft failures occur are often unknown, since the system may continue to operate even in the presence of a soft failure. However, the performance and efficiency of the system may be decreased [6, 7]. Flaw in a computer CD-ROM is an example of a soft failure, as the computer can still operate, despite it is unable to read or write from the CD. Another example of soft-type components is liquid-level alarm in infusion pumps.

Some sort of dependency exists between the components of multi-component systems, which differentiates these systems from single-component systems [8]. This dependency could be a failure, functional, structural, or an economic dependency.

The concept of components dependency is discussed in the literature. Wang and Pham [9] propose the optimal maintenance for systems consisting of several sub-systems. They assumed that the components are economically dependent. Dekker et al. [10] surveyed the economic dependency in multi-component systems. Ozekici [11] also studies the stochastic and economic dependencies and their influence on periodic replacement policies for a multi-component system. The series systems with mixed (cold and warm) standby components using the mean time to failure (MTTF), long-term availability and cost over benefit ratio are compared by Wang and Kuo [12]. A k-out-of-m system with perfect component and repairable repair-equipment is studied by Zhang and Wu [13]. In their study, they proposed a model that minimizes the total expected costs, based on the number the components which are repaired. This model is the optimal replacement policy. Taghipour and L Kassaei [14, 15], and Taghipour [16] consider a k-out-of-m load-sharing system with some dependency between their components. In their model, the failure of each component increases the hazard level of the remaining operational components. They [14, 15] develop a model to find the optimal inspection interval minimizing the total expected cost over a finite life cycle.

For optimal maintenance of multicomponent systems several models are proposed, and each of them is subject to multiple assumptions and structures, such as hidden failures, finite and infinite planning horizons, hard-component preventive replacements and opportunistic inspections. Some work is done to summarize research in optimal maintenance of

multicomponent systems. For example, Cho and Parlar [17] review the literature of the models developed over infinite time horizon. Wang [8] provides another review of different inspection and maintenance policies for both single-unit and multi-component systems; although his focus is more on single-unit component systems. However, the models in his surveys do not consider optimization of both inspection and maintenance over a finite time horizon.

Most of the maintenance optimization models assign different probability for each type of maintenance strategy. For example, age dependent probabilities for the type of maintenance such as replacement, minimal repair, or imperfect repair are considered by Sheu and Griffith [18]. Chien and Sheu [19] consider the system's age and the number of shocks tolerated since last replacement to evaluate the probability of replacement. Makis and Jardine [20] propose a model with both imperfect and perfect repair. The probability of perfect maintenance is assumed to be dependent on the number and time of imperfect maintenance actions in a cycle.

Most of the available maintenance optimization models are constructed considering the costs per unit time over infinite planning horizon. Wang and Zhang [21] develop an optimal mixed bi-variate policy model, based on the critical reliability level and the number of system failures for a simple system to find the minimum average cost rate. An optimal replacement policy model, based on the number of failures of each component for a series system with different components type, under a geometric process is proposed by Zhang and Wang [22]. Their model is able to optimize the system's costs per unit time. Wu and Zhang [23] consider an infinite-horizon multi-variable maintenance policy model for a two-component cold-standby system subject to Poisson shocks. Their maintenance policy model depends on the number of failures of each component and the interval length between two consecutive preventive

replacements. Coria et al. [24] consider an imperfect preventive maintenance policy model over infinite planning horizon. According to a new hazard function, they provide an analytical optimization method to optimize the costs of the system per unit time. A preventive maintenance policy model by considering the improvement factor and infinite planning horizon is proposed by Pan et al. [25]. Taghipour and Banjevic [7] consider models to optimize the inspection interval for a multi-component system, which is subject to hidden failures over both finite and infinite planning horizons.

Some systems contain components that are subject to hidden failures. Protective devices usually have the highest number of hidden failures. A hidden failure is defined as a failure which is not discovered during the normal operation of the system, and is rectified only at inspection [26, 27]. For example, the failures of computer CD-ROMs are only revealed either at inspection, or whenever the protective unit is required to function, but is unavailable due to failure. The only difference between soft failures and hidden failures is that the system is still able to function even with the presence of soft failures.

Hidden failures in single-component systems have been investigated by Sheu et al. [28] They assume that the number of previous repairs, affects the probability of failure, and the maintenance policy is based on both the component's age at failure and its number of previous repairs.

The optimal inspection period which maximises the profit of a multi-mode system is determined by Baohe [29]. He considers a combination of hidden and self-announcing failures in his model, and uses the supplementary variable technique where the inspection period is a random variable.

Another topic receiving an extensive treatment in the literature is opportunistic maintenance policy. An optimisation model for opportunistic preventive maintenance of a multi-component system with a series configuration is proposed by Zhou et al. [30]. Their system is the same as a multi-component system subject to only hard failures, since a series system fails by the failure of any one of its components. Dagpunar [31] looks into opportunistic replacement of a component in a multi-component system, if the age of a failed component surpasses a specified control limit. However, in his model, he does not consider any constraints for the type of components. Zhu et al. [32] propose an opportunistic maintenance policy model for a multi-component system with hard-type and soft-type components.

Cui and Li [33] consider opportunistic inspections and stochastic dependency between the components of a system. Aven and Dekker [34] develop a preventive opportunistic replacement model for components whose failures follow a Poisson process. Moreover, they consider an age-based replacement action for their model. One of the models in Taghipour and Banjevic [35] only considers opportunistic inspections and minimal repairs of hard-type and soft-type components for a multi-component system over a finite planning horizon. Taghipour and Banjevic [36] consider optimal periodic inspection interval for complex multi-component systems with hard and soft failures. Their model considers two types of inspection – periodic and opportunistic. However, in their model, the maintenance action is chosen based on age dependency probabilities, and it is not subject to optimisation.

While the literature considers inspection optimization of complex systems; the focus is generally more on periodic inspections [37-40]. Kapur and Butani [41] develop a model for optimal periodic inspection of a computer system with hidden and revealed failures. Zequeira

and Berenguer [42] propose a model to optimize a preventive maintenance policy for a system involving periodic inspections. They consider three types of maintenance actions for the system, including minimal repair, imperfect maintenance, and perfect replacement in their model. Zhao et al. [43] use cumulative hazard function and asymptotic mean time to failure, and propose approximate methods for estimating failure times and optimizing maintenance and inspections policies for a parallel system. Vaurio [44] considers the optimal periodic inspection interval for a multiple-component system. His optimization model is based on the system risk and cost.

Taghipour et al. [1] consider a model to optimize the periodic inspection for a complex repairable system involving hard and soft failures. Their model optimizes the expected total cost of the system over a finite planning horizon. Taghipour and Banjevic [35, 36] consider the possibility of both periodic and opportunistic inspections to extend their previous work in this area. The periodic inspections are scheduled in advance and opportunistic inspections occur at hard failure times. They also inspect the soft-type components in addition to the failed hard-type component at the events of the opportunistic inspections. Rezaei and Imani [45] propose a model to obtain an optimal periodic inspection interval on a finite time horizon for a multi-component repairable system. Since the components of the selected system assume to be economically dependent, grouping maintenance is preferred to individual maintenance. Rezaei and Imani [45] consider a combination of inspection, repair, and downtime penalty costs as the total expected cost in their model.

Wang et al. [46] study a multi-component system, in which they model the failure of each component separately and formulate a periodic inspection interval model for the entire the system.

Wang [47] proposes a model for a joint optimization of inspection interval policy and inventory level of spare parts. He forms his optimization model based on three decision variables: “the ordering quantity”, “ordering interval” and “inspection interval”. Moreover, Wang [47] considers the delay time concept and a two-stage failure process to construct the model.

Golmakani and Moakedi [48] consider a model for a two-component system in which one of the components is assumed to be a hard-type component, and another is a soft-type component. They [48] apply the same assumptions for soft-type and hard-type components as what Taghipour et al. [1] assume in their models to find the optimal interval for periodic inspections.

Wang [49] considers a system for a production process with two types of failure. The first type of failure is a product quality shift due to minor process defects which is assumed to be identified and rectified by routine inspections. The other type of failure is the major defect caused by a mechanical or electrical problem which should be inspected and repaired immediately after its emergence. In the current thesis, similarly to Wang [49], we assume two failure types. Another similarity of this study to Wang [49] is the type of inspections which are applied to the systems. Wang [47] considers two types of inspection, including routine and opportunistic inspections, which are similar to scheduled and opportunistic inspections. On the other hand, one of the main modifications of our work to Wang [47] is that we inspect all

components and rectify and repair the failed hard-type component as well as failed soft-type components at opportunistic inspections. Another difference of our work is that we develop an optimization model for a system which is subject to non-periodic inspections, while Wang [47] considers periodic inspection. However, we both assume that the failure times of soft-type components are unknown.

Wang et al. [50] propose a model for two-level inspection policy of a single component system based on a three-stage failure process. They [50] jointly optimize the minor and major inspection intervals, as well as a threshold level for the next planned maintenance by minimizing the expected cost per unit time. Mendes et al. [51] propose a model to analyze the reliability of active and standby redundant systems, and determine their optimal periodic inspection interval using Markov chain and search technique.

Another research area related to inspection optimization of complex multi-component systems is non-periodic inspections, which have been studied in the literature [52, 53]. A model for a two-unit system, in which each unit is subject to systematic failure and is inspected by sequential non-periodic inspections is presented by Castanier et al. [54]. Castanier et al. [54] consider preventive and corrective replacements. They assume a parametric maintenance decision to control the inspection and replacement policy. However, minimal repair of the units is not considered as a possible maintenance option, which can affect the expected system cost value.

Wang and Christer [55] study three solution algorithms for non-periodic inspection of a multi-component systems. The first algorithm is assigned to calculate the system replacement time when arrival process is non-homogeneous. The second algorithm is an extension of the

first one, in which non-constant optimal inspection intervals are considered. Finally, the third algorithm is a numerical algorithm for solving an integral equation to obtain the expected time of opportunistic inspections at the system failure times.

Golmakani and Moakedi use dynamic programming and branch-and-bound methods [56] to optimize non-periodic inspections of a multi-component system. The A^* search algorithm, based on branch-and-bound method, is introduced by them [56]. The challenge of A^* search algorithm is to generate and evaluate fewer numbers of nodes by branching only the most capable nodes at each step of the examination. Computing a large number of nodes at each step is one of the most important weaknesses of A^* search algorithm.

Lam and Banjevic [57] propose a model to decide at an inspection point whether a deteriorating system should be replaced immediately or the replacement should be postponed until later. They [57] also decide on when to schedule the next inspection. Barker and Newby [58] propose a model for a multi-variable stochastic process system. Their [58] model is designed to obtain non-periodic inspection and maintenance policy for a multi-component system in which the state of failure is modeled using a Markov stochastic process. Zhao et al. [59] develop another non-periodic inspection model for a complex multi-component system which is affected by a dynamic environment. They [59] use covariates process to define environment effects, and a stochastic univariate process is used to model the system's failure. Zhao et al. [59] derive the optimal maintenance threshold to minimize the expected maintenance cost per time unit and find the optimal non-periodic inspection sequence.

One of the most common types of multi-component complex systems which is used in industry for the last decade is a k-out-of-m system. A k-out-of-m system is an active redundant

system consisting of m components, in which a minimum of k components must be operational for the system to function [60-69].

The optimization of inspection policies for a k-out-of-m system is the main focus of many publications. One of the research papers in this area is Taghipour and Kassaei [15], which offers a model to optimize the periodic inspection interval for a k-out-of-m system assuming load-sharing between all identical components. In the current research, we also consider a k-out-of-m system, but we do not consider the load concept for the components. The main difference of the current work with the work of Taghipour and Kassaei [15] is that we consider a non-periodic inspection strategy for the system.

Bjarnason et al. [68 -71] propose a joint optimisation model for a k-out-of-m redundant system, and minimise the total costs of both maintenance and inventory policies. They consider hidden failures for the system. In another work, Bjarnason and Taghipour [72] find the optimal maintenance and (s, S) inventory policies for a k-out-of-m system subject to hidden failures. They search through a three-dimensional objective function by using genetic algorithm. As all components of k-out-of-m systems are identical, the analysis is much easier than a system containing various hard-type and soft-type components, because each component in the latter may require a special treatment, which makes the analysis much harder.

One of the models of our study is an extension of the models proposed by Taghipour and Banjevic [35, 36], but we develop an optimization model for non-periodic inspection of a system subject to soft and hard failures, and opportunistic inspections. Since the failure rates of the components are increasing as they get older, it is more reasonable to have less frequent inspections when the components are relatively new, and have more frequent inspections as

the probability of failure increases for the components. Our decision variables are binary variables indicating possible scheduled inspections' times for the system. Thus, the optimal non-periodic inspection scheme is the vector of the binary variables which results in the minimum total expected cost of the system over a finite time. We develop a simulation model to obtain the total expected cost of the system for a non-periodic inspection scheme, and combine this model with the genetic algorithm to obtain the optimal scheme more efficiently. At the next stage, we apply the same model for a k-out of-m system with similar assumptions.

For both models we assume minimal repair and replacement as possible maintenance actions for both hard-type and soft-type components and components of a k-out-of-m system. It is not generally possible to obtain an analytical solution for the optimal inspection interval, even in the simpler case of optimizing system availability regardless of the costs. For this reason, simulation is used to calculate the required expected values and to exhaustively search for the optimal solution in the case of a complex system.

Overall, inspection optimization models can be used as valuable tools in providing the safe and reliable operation of various equipment. Such models can also have strong managerial implications, since in practice, it is usually important to justify and support managerial decisions with both qualitative and quantitative analysis in order to make them robust. Simulation models are particularly useful in this regard, as they can cover a great number of possible scenarios and provide the results both for a particular and the most general case. Using the proposed inspection optimization model, the decision-makers gain an opportunity to observe the outcomes of their managerial decisions and to find the combination of decisions that is most likely to result in the greatest cost savings without sacrificing the required reliability and

availability. For example, based on the model's output for a given system with particular component parameters, it may not require as frequent inspection as previously thought because of the accounted effect of the additional opportunistic inspections. This would result in cost savings, which would be especially significant, if the costs of inspection were particularly high.

Thus, the contributions of this thesis is two-fold: developing a model for non-periodic inspection of multi-component and k-out-of-m systems, and proposing a solution algorithm using Monte Carol simulation and genetic algorithm which can be efficiently applied in practice to obtain the optimal non-periodic inspection policy for industrial systems. The remaining chapters of this thesis are organized as follows: Chapter 2 presents the problem description and the models' assumptions. Chapter 3 describes the system with soft-type and hard-type components. Chapter 4 presents a k-out-of-m system. Finally, Chapter 5 concludes this study and proposes some possible extensions of the work. In Appendix A, genetic algorithm and simulation Matlab code for both models are provided.

Chapter 2 - Problem Description and the Models Assumptions

In this study, we consider two models, which there are similarity and differences between them. For the first model we assume a system consisting of hard-type and soft-type components. The number of hard-type components and soft-type components are shown by m_1 and m_2 , respectively. A repairable component can be a single part such as battery or line cord, or a subsystem, such as circuit breaker or charger in an infusion pump. The failures of both soft-type and hard-type components for the first two models follow a non-homogenous Poisson process with power law hazard rate $\lambda_j(t) = \frac{\beta_j}{\eta_j}\left(\frac{t}{\eta_j}\right)^{\beta_j - 1}$, $j = 1, \dots, m_1 + m_2$,where β_j and η_j are the parameters of the power law intensity function. The second model considered is a k-out-of-m with m identical components. The failures of m components also follow a non-homogenous Poisson process with power law hazard rate $\lambda(t) = \frac{\beta}{\eta}\left(\frac{t}{\eta}\right)^{\beta - 1}$.

The lifecycle length of all three systems is shown by T. The failure of soft-type components is revealed by inspecting the system non-periodically. Once a soft-type component fails, it stays in the same state until it gets fixed at the first approaching inspection, which could be a non-periodic scheduled inspection, or an opportunistic inspection assumed to be at a hard failure's time. Alternatively for the k-out-of-m model, if the number of failed components is less than $m - k + 1$, we consider them as soft failures; and if the number of failed components reaches $m - k + 1$, we consider the $(m - k + 1)^{st}$ failure as a hard failure, since the system

stops immediately, and the system failure provides an opportunity to inspect all the components.

The maintenance action applies to failed components; depending on the age of the component at failure time, the component either requires a minimal repair or replacement. It is assumed that there is no dependency between the failures of a component to other components. It is also assumed that soft failures or the combination of soft failures cannot change to a hard failure and hard components cannot fail at the same time. Both types of inspection are considered to be perfect inspections with negligible time for inspection, replacement and minimal repair.

In a k-out-of-m system, we also assume that when a component fails, it stays in the same condition until it gets fixed either at a system failure time, or at a scheduled inspection. We consider a penalty cost for the downtime of each component, as well as a penalty cost for each time the system fails.

The minimum time unit is assumed to be τ, which means a scheduled inspection can be only performed at $l\tau(l = 1, 2, 3 \dots \mathrm{n})$. In this way, the time interval between two uninterrupted scheduled inspections cannot be less than τ. Each τ can be measured as a day, a week, or a month. The objective of this model is to find the minimum total expected cost of the system over lifecycle length T, which is the result of the optimal non-periodic inspection scheme.

Given τ, the system potentially can be inspected at times $l\tau(l = 1, 2, 3 \dots \mathrm{n})$, where $n = \frac{T}{\tau} - 1$, if T is divisible by τ, and $n = \left[\frac{T}{\tau}\right]$, otherwise. In other words, at each scheduled inspection time $l\tau$, we should decide whether the system should be inspected or not. We also

assume that at the time of a hard failure, all the soft-type components and the failed hard-type component are inspected opportunistically (Figure 2.1).

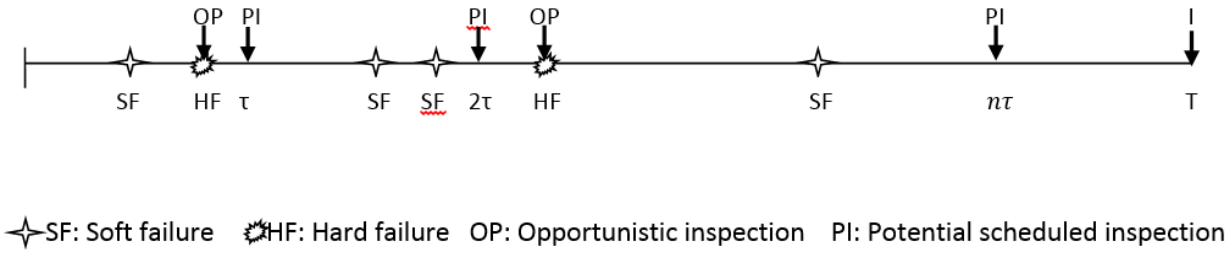

Figure 2.1. Sample soft and hard failures, and potential and opportunistic inspections over cycle T.

Similarly, we assume that at the time of the system failure, all the failed components are inspected opportunistically (Figure 2.2).

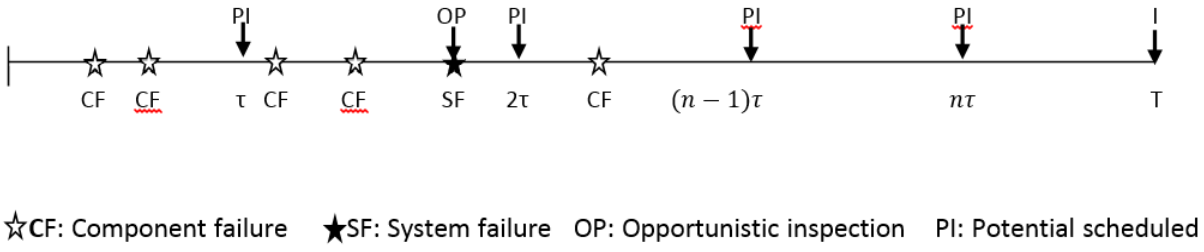

Figure 2.2. Sample of the failures of a k-out-of-m system, and potential and opportunistic inspections over cycle T.

We assume at time T the last scheduled inspection is performed to prepare the system for the next cycle. The preparation could be the renewal of the entire system, or replacement of some of the components and minimal repairs of the others. In this case, the components will

start with different initial ages. Let us define x_l $(l = 1, 2, 3 \dots \mathrm{n})$ as a binary decision variable, which is 1 if inspection is performed at scheduled inspection time $l\tau$, and is 0, otherwise. In this case, x_l $(l = 1, 2, 3 \dots \mathrm{n})$ construct the vector $\vec{x} = (x_1, x_2, , \dots x_n)$, which describes a possible scheduled inspection scheme for the system. Moreover, we define X_k^+ and $\mathrm{y}_{\mathrm{x}_k^+}$ as follows:

$$\forall \boldsymbol{x_k} \neq \mathbf{0} : \ \boldsymbol{x_k^+} = \sum_{\boldsymbol{l=1}}^{\boldsymbol{k}} \boldsymbol{x_l}, \tag{2.1}$$

$$\boldsymbol{y_0} = \mathbf{0}, \quad \boldsymbol{y_{x_k^+}} = \boldsymbol{l\tau} - \sum_{\boldsymbol{l=0}}^{\boldsymbol{x_k^+ - 1}} \boldsymbol{y_l} \ . \tag{2.2}$$

In fact, x_k^+ is counting the number of scheduled inspections up to time $l\tau$ (including $l\tau$), and $y_{x_k^+}$ is the interval between $(x_k^+)^{th}$ and $(x_k^+ - 1)^{th}$ inspections.

For the last inspection at the end of cycle time T we have:

$$\boldsymbol{x_{n+1}^+} = \mathbf{1} + \boldsymbol{x_n^+} \ , \tag{2.3}$$

$$\boldsymbol{y_{x_{n+1}^+}} = \boldsymbol{T} - \sum_{\boldsymbol{l=0}}^{\boldsymbol{x_n^+}} \boldsymbol{y_l} \ . \tag{2.4}$$

Thus, the total expected cost of the system over T can be formulated as follows:

$$\boldsymbol{E\left[C_{S,\vec{x}}^{T}\right] = C_I x_{n+1}^{+} + \sum_{j=1}^{m_2} \left(C_j^M M_T^j\left(\vec{x}, T, t_j, \vec{\theta}\right) + C_j^R R_T^j\left(\vec{x}, T, t_j, \vec{\theta}\right) + C_j^D \left(T - D_T^j\left(\vec{x}, T, t_j, \vec{\theta}\right)\right)\right),} \tag{2.5}$$

where at the beginning of cycle T, the soft-type component j has the initial age t_j . The initial ages of the hard-type components at the beginning of cycle T is given by vector $\vec{\theta} = (\theta_1, \theta_2, \dots . , \theta_{m_1})$. The total number of scheduled inspections by the end of cycle T is provided by x_{n+1}^+. The expected number of minimal repairs and replacements, and the expected downtime of soft-type component j over cycle T are denoted by

$M_T^j(\vec{x}, T, t_j, \vec{\theta}), R_T^j(\vec{x}, T, t_j, \vec{\theta})$, and $D_T^j(\vec{x}, T, t_j, \vec{\theta})$, respectively. The costs of a minimal repair, replacement, and the downtime penalty per unit time, respectively, for component j are C_j^M, C_j^R, and C_j^D . C_I is the cost of a scheduled inspection. Our objective is to find the optimal scheme $\vec{x}$ which minimizes the total expected cost of the system $E\left[C_{S,\vec{x}}^T\right]$.

The costs of minimal repairs and replacements of hard-type components, and the opportunistic inspections have no impact on the optimal inspection scheme; thus, these costs are not included in $E\left[C_{S,\vec{x}}^T\right]$.

The total expected cost of the k-out-of-m system over cycle length T for each scheme is shown by $E\left[C_{S,\vec{x}}^T\right]$ and is calculated as follows:

$$\begin{aligned} \boldsymbol{E\left[C_{S,\vec{x}}^T\right] = C^I(\ total\ number\ of\ schduled\ inspections\ ) +} \\ \boldsymbol{C^D(\ total\ expected\ downtime) +} \\ \boldsymbol{C^M(\ total\ expected\ number\ of\ minimal\ repairs) +} \\ \boldsymbol{C^R(\ total\ expected\ number\ of\ replacements) +} \\ \boldsymbol{C^{SD}(expected\ number\ of\ system\ failures)}, \end{aligned} \quad \textbf{(2.6)}$$

where C^I, C^M, C^R, C^D, and C^{SD} denote the scheduled inspection cost, cost of minimal repair of a component, cost of replacement of a component, the penalty cost for the downtime of a components per unit time, and system failure cost, respectively. As shown, we consider the cost of system failures as being additional.

Chapter 3 – The System with Soft-Type and Hard-Type Components

To calculate the expected cost of the system, we are required to obtain the expected number of minimal repairs, replacements, and expected downtime of each soft-type components stated in (2.5). First, we develop a recursive mathematical formula to calculate these expected values for a single soft-type component. The recursive formula is designed with a placeholder function to be able to calculate different required expected values. However, calculating the expected values from the recursive formula is computationally intensive, because it involves multi-dimensional integrals which must be obtained numerically by discretization and solving systems of equations.

We assume all hard-type components are considered as a subsystem of the whole system in a series configuration. The intensity function of hard-type component j at time z is $\lambda_j(z|\theta_j) = \lambda_j(\theta_j + z),\ \ j = 1,2, \dots . m_1$, where θ_j is the initial age of the hard-type component. Thus, the intensity of the hard subsystem at time z is $\lambda_H(z|\boldsymbol{\theta}) = \sum_{j=1}^{m_1} \lambda_j(\theta_j + z)$, $\boldsymbol{\theta} = (\theta_1, \theta_2, \dots, \theta_{m_1})$.

Let us define the densities of the failure of the soft-type component and hard subsystem as follows:

$$f^{X}(x \mid t) = \lambda(t+x)e^{-\int_{t}^{t+x}\lambda(s)ds}, \quad f^{Z}(z \mid \boldsymbol{\theta}) = \lambda_H(z \mid \boldsymbol{\theta})e^{-\int_{0}^{z}\lambda(s|\boldsymbol{\theta})ds}.$$

The reliability functions of the soft-type component and the hard subsystems are $R^X(x|t)$ and $R^Z(z|\boldsymbol{\theta})$, respectively. The probability of the failure of the hard-type component j with initial age of θ_j at time $Z = z$ is equal to:

$$q^j(z) = q^j(z|\boldsymbol{\theta}) = \frac{\lambda_j(z|\theta_j)}{\lambda_H(z|\boldsymbol{\theta})}.$$

For the soft-type component, we have three possible events in one scheduled inspection interval. We use I to show all the possible cases.

$$I^X = \begin{cases} 0, & failure\ and\ then\ minimal\ repair\ at\ the\ first\ inspection, \\ 1, & failure\ and\ then\ replacement\ at\ the\ first\ inspection, \\ 2, & no\ failure\ in\ the\ first\ inspection\ interval. \end{cases}$$

To derive a formula that can be generally used to obtain the expected value of any random variable, we consider a placeholder function $\psi(x, I^X, z)$, in which x represents the failure time of the soft-type component, and z represents the failure time of hard–type component j, and I^X represents the failure/survival/maintenance action on the soft-type component. Therefore, the general formula remains the same for all random variables, and only the place holder function changes accordingly as follows:

For the number of minimal repairs of the soft-type component:

$$\psi(x, I^X, z) = \begin{cases} 1, & \text{if } I^X = 0 \\ 0, & \text{otherwise} \end{cases}.$$

For the number of replacement of the soft-type component:

$$\psi(x, I^X, z) = \begin{cases} 1, & \text{if } I^X = 1 \\ 0, & \text{otherwise} \end{cases}.$$

For the downtime of the soft-type component:

$$\psi(x, I^X, z) = \begin{cases} x, & \text{if } x < \text{z} \\ \text{z}, & \text{otherwise} \end{cases}.$$

The probability of minimal repair of soft-type component with age x at failure is $r^X(x)$, and $\bar{r}^X(x) = 1 - r^X(x)$ is the probability of replacement. The probability of minimal repair of hard-type component j with age z at failure is r_j^Z, and $\bar{r}_j^Z = 1 - r_j^Z$ is the probability of replacement.

Let us assume there is only one scheduled inspection interval at the end of cycle length T, i.e. $n = 0$ or $y_{x_{n+1}^+} = y_1 = 1$. We derive a recursive formula to obtain the expected number of minimal repairs, replacements and downtime for a soft-type component over the inspection interval y_1. Let G_{y_1} denote the expected value of a random variable of interest at the end of the inspection interval y_1. Assuming the initial ages of the soft-type component and the hard subsystem at the beginning of the inspection interval are t and $\boldsymbol{\theta}$, respectively, and by conditioning on the first failure time of the soft-type component, i.e. x and the first hard failure time, i.e. z, $G_{y_1}(t, \boldsymbol{\theta})$ can be recursively obtained as follows:

$$G_{y_1}(t, \boldsymbol{\theta}) = \int_0^{y_1} \int_0^z \sum_{j=1}^{m_1} \Big\{ \Big([\psi(x, 0, z) + G_{y_1 - z}(t + x, \boldsymbol{\theta} \oplus z)] r^X(t + x)$$

$$+ [\psi(x, 1, z) + G_{y_1 - z}(0, \boldsymbol{\theta} \oplus z)] \bar{r}^X(t + x) \Big) r_j^Z(\theta_j + z) + \Big([\psi(x, 0, z) + G_{y_1 - z}(t + x, (\boldsymbol{\theta} \oplus z)^{(0_j)}] r^X(t + x)$$

$$+ [\psi(x, 1, z) + G_{y_1 - z}(0, (\boldsymbol{\theta} \oplus z)^{(0_j)})] \bar{r}^X(t + x) \Big) \bar{r}_j^Z(\theta_j + z) \Big\} \times q^j(z) f^X(x \mid t, z) f^Z(z \mid \boldsymbol{\theta}) dx dz$$

$$+\int_0^{y_1}\sum_{j=1}^{m_1}\{[\psi(z,2,z)+G_{y_1-z}(t+z,\boldsymbol{\theta}\oplus z)]r_j^Z(\theta_j+z)$$
$$+[\psi(z,2,z)+G_{y_1-z}(t+z,(\boldsymbol{\theta}\oplus z)^{(0_j)})]\bar{r}_j^Z(\theta_j+z)\}\times q^j(z)R^X(z\mid t)f^Z(z\mid\boldsymbol{\theta})dz$$

$$+\int_0^{y_1}\{\psi(x,0,y_1)r^X(t+x)+\psi(x,1,y_1)\bar{r}^X(t+x)\}f^X(x\mid t,y_1)dxR^Z(y_1\mid\boldsymbol{\theta})$$
$$+\psi(y_1,2,y_1)R^X(y_1\mid t)R^Z(y_1\mid\boldsymbol{\theta}), \qquad \textbf{(3.1)}$$

where $\boldsymbol{\theta}\oplus z$ shows the addition of scalar z to the elements of vector $\boldsymbol{\theta}$, and $(\boldsymbol{\theta}\oplus z)^{(0_j)}$ indicates that the jth element of vector $\boldsymbol{\theta}\oplus z$ is set to zero.

In general, we develop a recursive formula to obtain the expected values for other inspection intervals $l=x_{n+1}^+,\ldots,2$:

$$G_{y_l}(t,\boldsymbol{\theta})=\int_0^{y_l}\int_0^{z}\sum_{j=1}^{m_1}\Big\{\Big([\psi(x,0,z)+G_{y_l-z}(t+x,\boldsymbol{\theta}\oplus z)]r^X(t+x)$$

$$+[\psi(x,1,z)+G_{y_l-z}(0,\boldsymbol{\theta}\oplus z)]\bar{r}^X(t+x)\Big)r_j^Z(\theta_j+z)+\Big([\psi(x,0,z)+G_{y_l-z}(t+x,(\boldsymbol{\theta}\oplus z)^{(0_j)})]r^X(t+x)$$

$$+[\psi(x,1,z)+G_{y_l-z}(0,(\boldsymbol{\theta}\oplus z)^{(0_j)})]\bar{r}^X(t+x)\Big)\bar{r}_j^Z(\theta_j+z)\Big\}\times q^j(z)f^X(x\mid t,z)f^Z(z\mid\boldsymbol{\theta})dxdz$$

$$+\int_0^{y_l}\sum_{j=1}^{m_1}\{[\psi(z,2,z)+G_{y_l-z}(t+z,\boldsymbol{\theta}\oplus z)]r_j^Z(\theta_j+z)$$
$$+[\psi(z,2,z)+G_{y_l-z}(t+z,(\boldsymbol{\theta}\oplus z)^{(0_j)})]\bar{r}_j^Z(\theta_j+z)\}\times q^j(z)R^X(z\mid t)f^Z(z\mid\boldsymbol{\theta})dz$$

$$+\int_0^{y_l}\{[\psi(x,0,y_l)+G_{y_{l-1}}(t+x,\boldsymbol{\theta}\oplus y_l)]r^X(t+x)$$

$$+[\psi(x,1,y_l)+G_{y_{l-1}}(0,\theta\oplus y_l)]\bar{r}^X(t+x)\}f^X(x\,|\,t,y_l)dxR^Z(y_l\,|\,\theta)$$

$$+[\psi(y_l,2,y_l)+G_{y_{l-1}}(t+y_l,\theta\oplus y_l)]R^X(y_l\,|\,t)R^Z(y_l\,|\,\theta). \quad \textbf{(3.2)}$$

Since calculating the expected values from the recursive formula is computationally intensive, in the next section, we develop a simulation model to obtain the expected costs in equation (2.5) and equation (2.6) for a given non-periodic inspection scheme $\vec{x}$.

3.1. Simulation model to obtain $\mathbf{E}\left[\mathbf{C}_{\mathbf{S},\vec{\mathbf{x}}}^{\mathbf{T}}\right]$

The inputs of the simulation model are an inspection scheme $\vec{x}, T, \tau, \vec{\theta}, C_j^M, C_j^R, C_j^D, \beta_j$ and η_j, where $j = 1, \dots, m_1 + m_2$, and the initial age t_j of soft-type components j, $j = 1, \dots, m_2$. The output of the simulation model is $E\left[C_{S,\vec{x}}^T\right]$. We also define the age-dependent function $r(t_j) = a_j e^{-b_j t_j}$, which specifies the probability of minimal repair for a component with age t_j . $1 - r(t_j)$ is the probability of replacement of the component. We assume that a_j and b_j are given. We generate the first failure times for all components $j = 1, 2, \dots , m_1 + m_2$, given their initial ages**.** The first failure time for component j is obtained from $\eta_j \left(\left(\frac{t_j}{\eta_j} \right)^{\beta_j} - \log(z) \right)^{\frac{1}{\beta_j}} - t_j$, in which z is a random variable from a uniform distribution over (0, 1).

We add an element 1 to the end of vector $\vec{x}$ to include the last scheduled inspection which is always performed at time T to prepare the system for the next cycle. Then, from (1) and (2), we obtain $y_{x_k^+}$ for $k = 1, 2, 3 \dots \mathrm{n}$, which are the time between two consecutive scheduled inspections. In the next step, we find the minimum of the first hard failure times and compare it with y_1. From this comparison, we have the following two possible cases:

Case 1: If y_1 is more than the minimum hard failure's time, we experience an opportunistic inspection within y_1. We then compare the first failure time for each soft-type component j with the minimum hard failure's time. As a result, the two following cases may happen:

Case 1-1: If the minimum hard failure's time is greater than the first failure for soft-type component j, then the soft failure is detected at the opportunistic inspection's time (i.e. hard failure's time). At this time, we minimally repair or replace the soft-type and hard-type

components according to $r(x)$, where x is the age of the component at the failure time. We update the variables which are keeping track of the number of minimal repairs, replacements and downtime of soft-type component j. We then generate the time to the next failure for soft-type component j.

Case 1-2: If the minimum hard failure's time is less than the first failure for soft-type component j, then at this time, we minimally repair or replace the hard-type component according to $r(x)$, where x is the age of the component at the repair time.

In both *Cases 1-1* and *1-2*, we move forward the simulation clock to the minimum hard failure's time, and subtract the minimum hard failure's time from y_1, and the first failure times of the survived components (soft-type and hard-type). We also generate the time to the next failure for the failed hard-type component with the minimum hard failure's time.

Case 2: If y_1 is less than the minimum hard failure's time, the next inspection time is y_1. We then compare y_1 with the first failure time of each soft-type component j. As the result, two possible cases may happen:

Case 2-1: If y_1 is greater than the first failure of soft-type component j, the soft failure is detected at y_1. At this time, we minimally repair or replace soft-type component j according to $r(x)$, where x is the age of the component at the failure time. We update the variables which are keeping track of the number of minimal repairs, replacements and downtime of soft-type component j. We then generate the time to the next failure for the soft-type component.

Case 2-2: If the first failure for soft-type component j is greater than y_1, we just follow the steps described below.

In both *Case 2-1* and Case *2-2*, we move forward the simulation clock to y_1, and reduce this time from the first failure times of the survived components (soft-type and hard-type). We then use y_2 as the next scheduled inspection interval.

We follow the steps described above, until the simulation clock reaches T.

Each simulation run returns the estimates for the expected number of minimal repairs, replacements and downtime for each soft-type component j. The average of these estimates from multiple simulation runs is $M_T^j(\vec{x}, T, t_j, \vec{\theta}), R_T^j(\vec{x}, T, t_j, \vec{\theta})$, and $D_T^j(\vec{x}, T, t_j, \vec{\theta})$. We then obtain $E[C_{S,\vec{x}}^T]$ based on equation (2.5). The simulation model should be run for all possible inspection schemes $\vec{x}$, which are 2^n cases, starting from $\vec{x} = (0,0,\dots,0)$ and ending to $(1,1,\dots,1)$. The minimum $E[C_{S,\vec{x}}^T]$ among all possible inspection schemes returns the optimal inspection scheme $\overrightarrow{x^*}$.

It should be noted that x_{n+1}^+ is always 1, because even when we have the inspection scheme $\vec{x} = (0,0,\dots,0)$ as the input of the simulation model, we still perform one inspection at the end of lifecycle T. Thus, always $\mathrm{y}_1 > 0$.

The number of possible inspection schemes is increased significantly by increasing n (i.e. 2^n schemes). In the case of large n, checking all possible schemes to find the optimal one is not efficient. To find the optimal inspection scheme more efficiently, we embed the simulation model described above in a Genetic Algorithm (GA).

3.2. Coupling the simulation model and the genetic algorithm (GA)

The genetic algorithm is a heuristic search method, which is applied to the problems with large solutions space to find the optimal solution [73, 74]. The algorithm starts with a population of candidate solutions (called a generation), and iteratively evaluates the fitness of each solution in the generation. The next generation is constructed based on the solutions with the best fitness in the previous generation [75, 76].

For the fitness value of the GA, we use the simulation model described previously. The GA randomly selects the initial generation consisting of multiple inspection schemes and evaluates each solution's fitness using the simulation model. Since we need to have multiple simulation replications (the replication number in our numerical example is 5,000) to obtain $E\left[C_{S,\vec{x}}^{T}\right]$ for each solution (i.e. $\vec{x}$), we save the value of $E\left[C_{S,\vec{x}}^{T}\right]$ at the first time it is calculated by the simulation model. Thus, if a solution is repeated in the next generations, the simulation model just returns its corresponding $E\left[C_{S,\vec{x}}^{T}\right]$, which had been previously stored. This can save a significant amount of time.

For our problem, to construct a relationship between n (the number of possible scheduled inspections) and the population size to be used in the GA, we first obtain the optimal inspection scheme for different values of $n = 1, 2, \ldots, 13$, using only the simulation model (i.e. all possible inspection schemes were examined for each n). We then investigate the performance of the GA, to observe for which population size it results in the same optimal scheme which is obtained solely by the simulation model. After this investigation, we determined that the population size should be approximately $0.4193(n)^{1.9289}$, so the GA can

obtain the global optimal solution (with the coefficient of determination $R^2 = 0.98$). The results of this investigation are shown in Figure 3.5.

To stop the GA, we consider the two following criteria:

***1) Fitness tolerance limit* and *stalls generation number*:** Fitness tolerance limit should be as small as possible to be able to distinguish between two possible solutions. If the fitness tolerance limit appears a certain number of times (equal to the stalls generation number), then the GA stops automatically.

2) Maximum number of generations: It should be adequately large, but not larger than the maximum number of possible solutions, which is 2^n.

By coupling the genetic algorithm and the simulation model, we can find the optimal inspection scheme much faster while requiring significantly fewer simulation runs.

3.3. Flowcharts of the Simulation model and the GA code

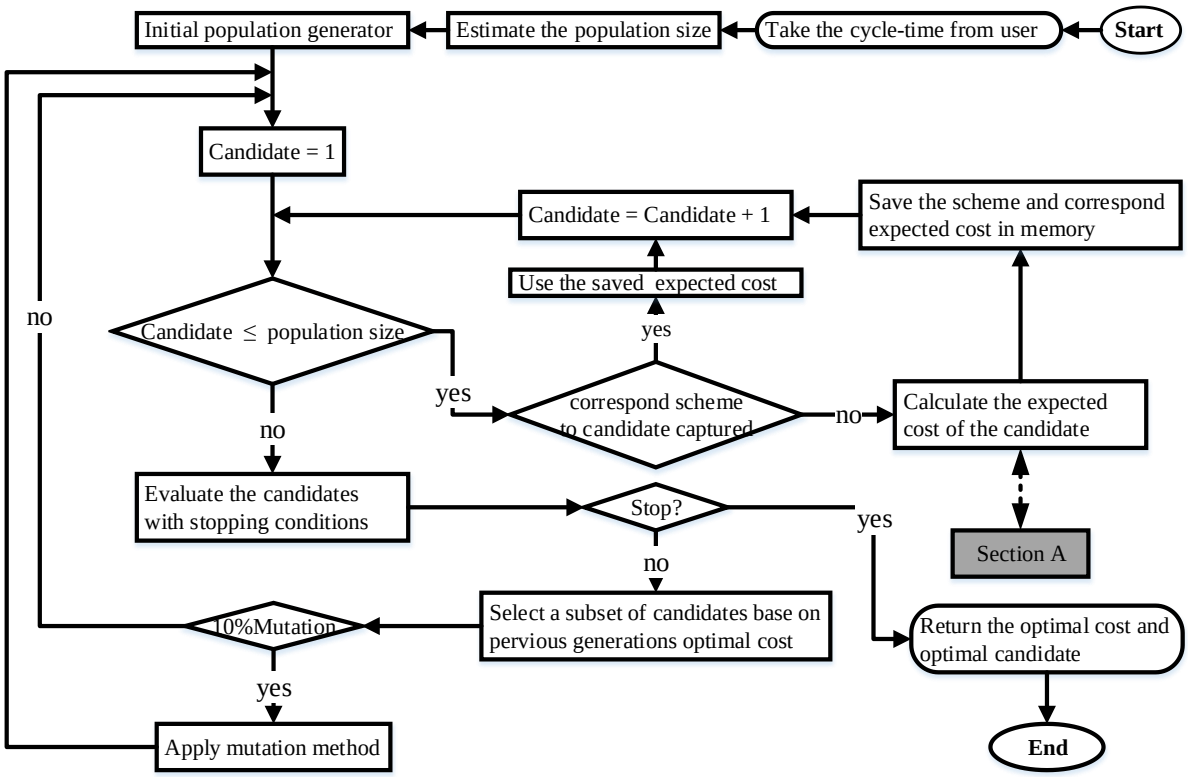

Figure 3.1. Genetic algorithm.

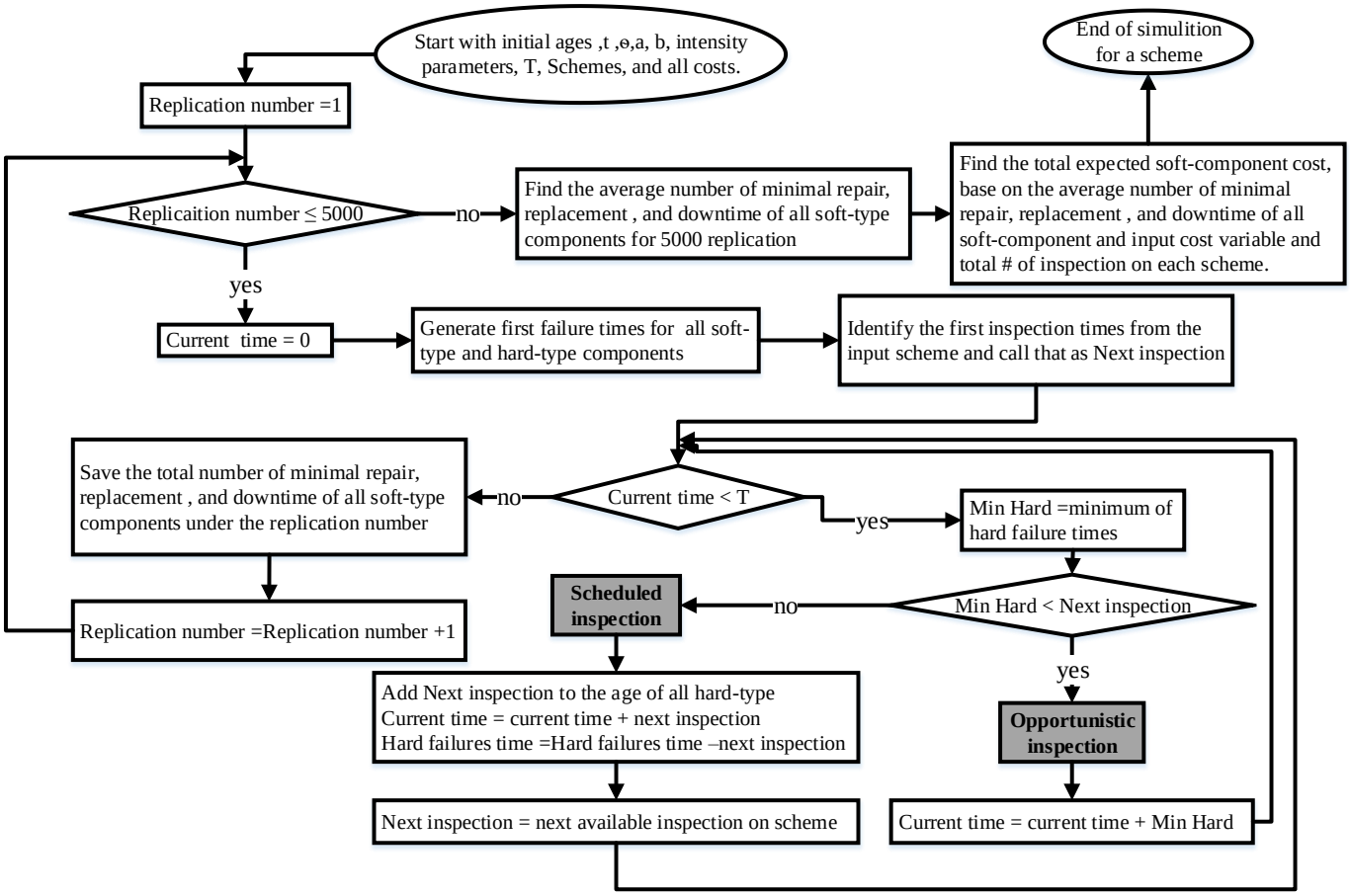

Figure 3.2. Simulation algorithm for the system with hard-type and soft-type components.

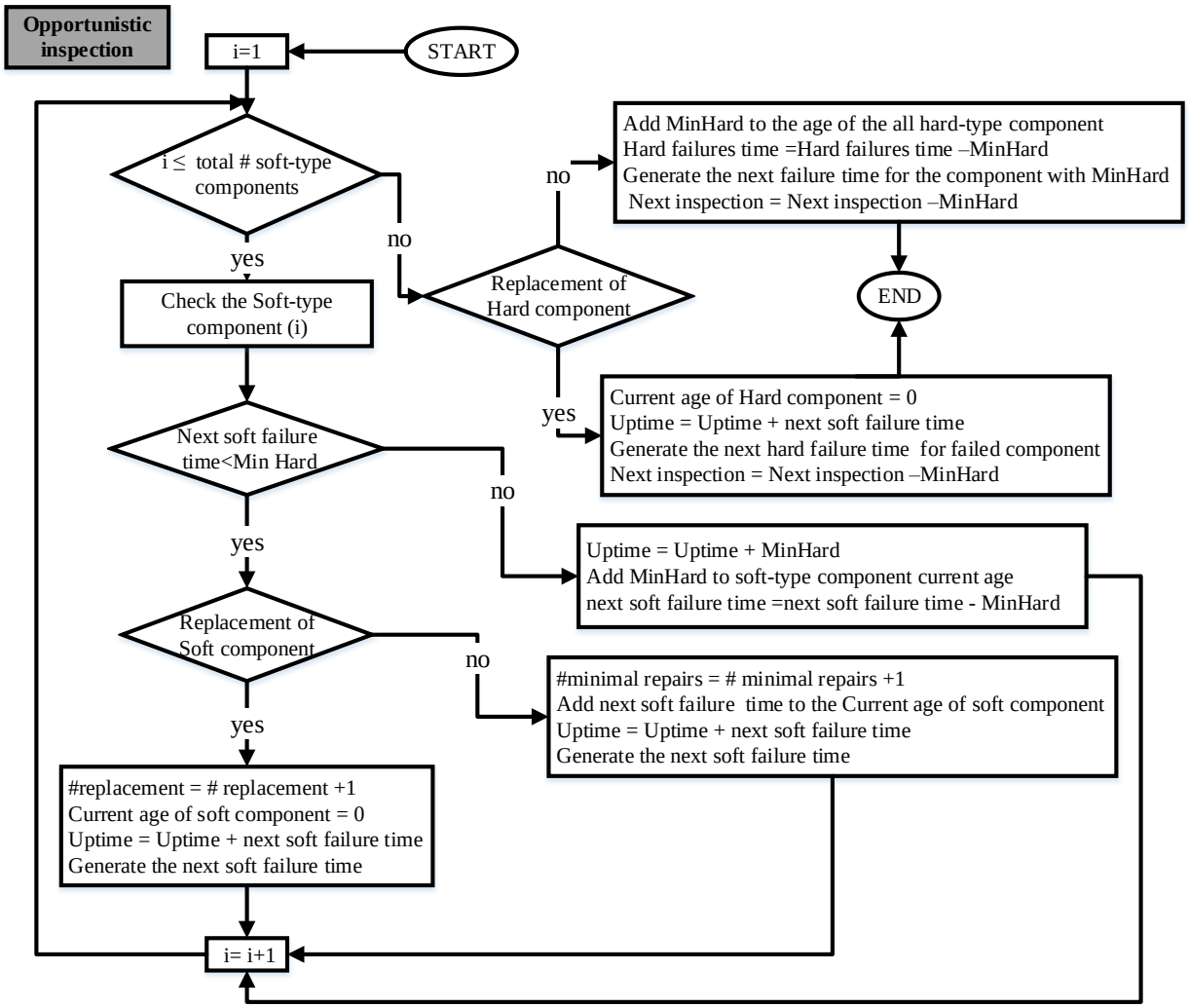

Figure 3.3. Algorithm for an opportunistic inspection of the system with hard-type and soft-type components.

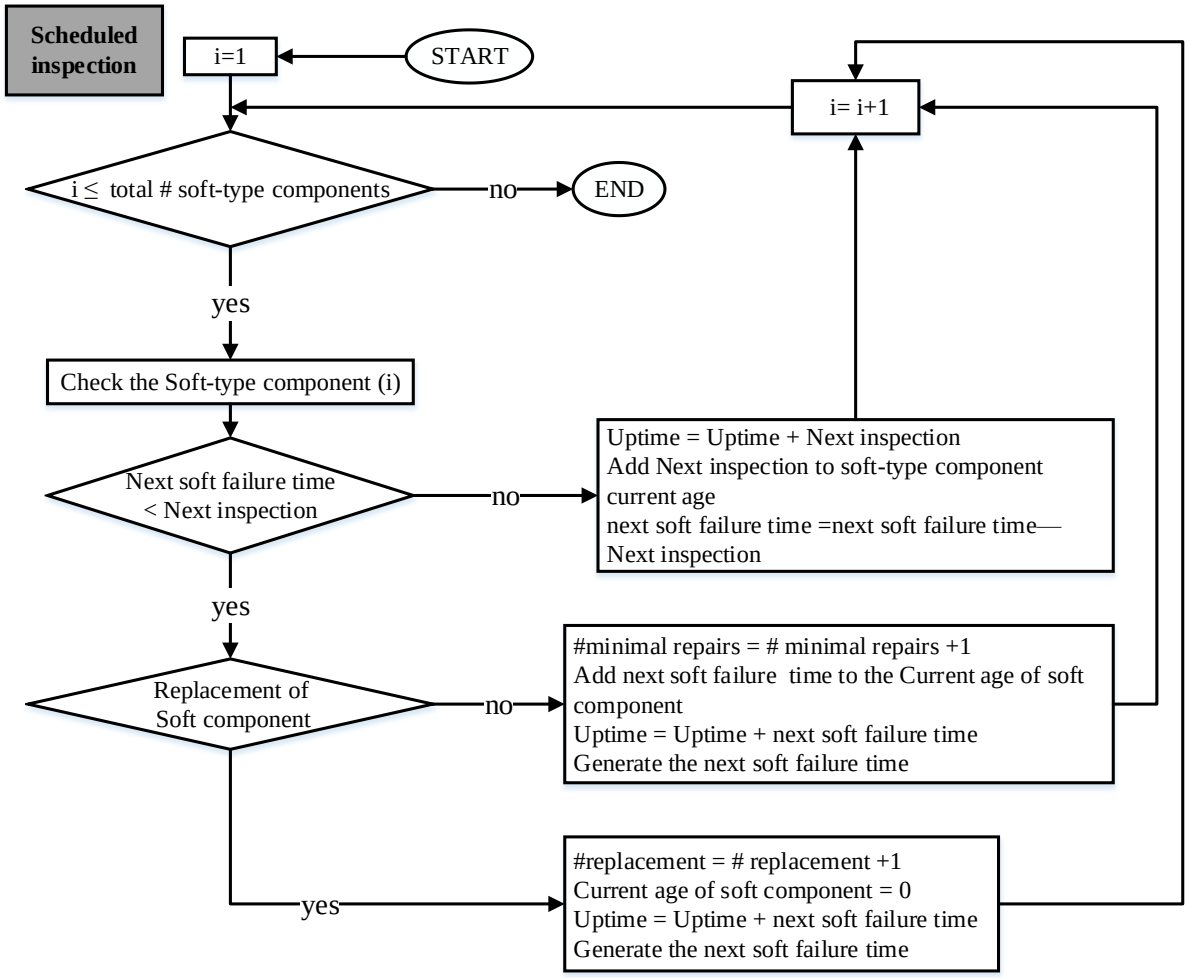

Figure 3.4. Algorithm for a scheduled inspection of the system with hard-type and soft-type components.

3.4. Numerical Example

We consider a multi-component repairable system with the total of eight components. The system consists of three hard-type components, and five soft-type components with the failures following a NHPP with power law intensity function.

The probability of minimal repairs for a component of age $\boldsymbol{t_j}$ is obtained from $\boldsymbol{r_j(t_j) = a_j e^{-b_j t}}$. Minimal repair, replacement, and downtime costs for the soft-type and hard-type components are given in Table 3.1.

Table 3.1. Costs of maintenance and downtime for different components.

Cost	**Soft type components**					**Hard type components**		
	1	2	3	4	5	1	2	3
Minimal repair $(\mathbf{c_j^M})$	\$70	\$45	\$100	\$75	\$150	\$100	\$200	\$150
Replacement $(\mathbf{c_j^R})$	\$200	\$150	\$300	\$450	\$280	\$240	\$450	\$600
Downtime penalty (c_j^D)	\$300	\$350	\$400	\$250	\$300	-	-	-

The cost of inspection is \$200 for each time we inspect the system. The parameters of the power law intensity function, and a_j and b_j are given in Table 3.2. The data for the numerical example is taken from Taghipour et al. [1].

Table 3.2. Parameters of the power law intensity function, and the probability function of minimal repair for all components.

	Soft type components					Hard type components		
	1	2	3	4	5	1	2	3
$\boldsymbol{\beta_j}$	1.3	1.1	2.1	1.8	1.7	1.5	1.2	1.7
$\boldsymbol{\eta_j}$ **(month)**	3.5	4.6	6	10	3.6	11	7.2	2.8
$\boldsymbol{a_j}$	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9
$\boldsymbol{b_j}$	0.2317	0.1763	0.1352	0.0811	0.2253	0.06	0.02	0.28

We are interested in finding the optimal non-periodic inspection scheme resulting in the minimum total cost incurred over life cycle T. We assume that all components are as-good-as new at the beginning of the cycle, so we set the initial ages for all components to zero. We also assume that $\tau = 1$ month is the minimum time unit at which a scheduled inspection can take place.

To obtain the optimal inspection scheme ($\overrightarrow{x^*}$) for different cycle lengths T, we calculate $E\left[C^T_{S,\vec{x}}\right]$ from the simulation model for different values of n. In the simulation model, we consider 5,000 replications. In Table 3.3, the results of the simulation model are presented, by which all the possible inspection schemes for different cycle lengths T are examined. In addition, the minimum expected cost, the optimal inspection scheme for each value of T and n are presented.

Table 3.3. Simulation results for different cycle lengths T.

Cycle length T (month)	**# of possible scheduled inspections (n)**	**# of possible inspection scheme/s (2^n)**	**Time to examine all the schemes (seconds)**	$E\left[C^T_{S,\overrightarrow{x^*}}\right]$ **($)**	**Optimal scheme $\overrightarrow{x^*}$**
5	4	16	379	2,027.43	01001
6	5	32	846	2,553.35	001001
7	6	64	2,017	3,092.53	0010101
8	7	128	4,831	3,648.99	00010101
9	8	256	10,378	4,193.31	001010101
10	9	512	22,222	4,756.09	0010010101
11	10	1024	50,388	5,287.61	00100010101
12	11	2048	95,481	5,854.56	001010100101
13	12	4096	198,774	6,385.02	0001010010101

For example, the optimal scheme for $T = 12$ is $\overrightarrow{x^*} = (001010100101)$, which implies that we should inspect the system at the third, fifth, seventh and tenth months. This is a reasonable result, since at the beginning of the cycle, the components still are new and do not fail often, so we need fewer inspections. Similarly, by the middle of the lifecycle, more components have been replaced rather than minimally repaired. Therefore, we have less frequent inspections when the components are still new, and have more inspections as the components get older or they are subject to more minimal repairs.

For different values of n and cycle lengths T provided in Table 3.3, we use the genetic algorithm to obtain the optimal schemes in shorter time. As the first step, we find the population size for the GA which results in the same optimal solutions given in Table 3.3. In other words, we determine what population size makes the integration of the GA and

simulation model valid (i.e. provides the global optimal inspection policy). The conditions and criteria for running the genetic algorithm are presented in Table 3.4.

Table 3.4. The conditions and criteria for running the genetic algorithm.

Conditions and criteria	Numerical value	Description
Simulation replication for each scheme	5000	The simulation runs 5,000 times for each scheme
Population size	$0.4193(n)^{1.9289}$	Depends on n
Maximum generation number	50	The GA stops after 50 generations
Stall generation limit	20	If the tolerance fitness function appears 20 times in a row, the GA stops
Tolerance of fitness function	e^{-5}	The difference between two consecutive fitness values

The results of the GA are shown in Table 3.5.

Table 3.5. Minimum required population size for different cycle lengths T.

Cycle length T (month)	# of possible Scheduled inspection (n)	Population size
5	4	6
6	5	10
7	6	12
8	7	20
9	8	24
10	9	25
11	10	32
12	11	50

We determine a power relationship between n and the population size in the GA (Figure. 3.5).

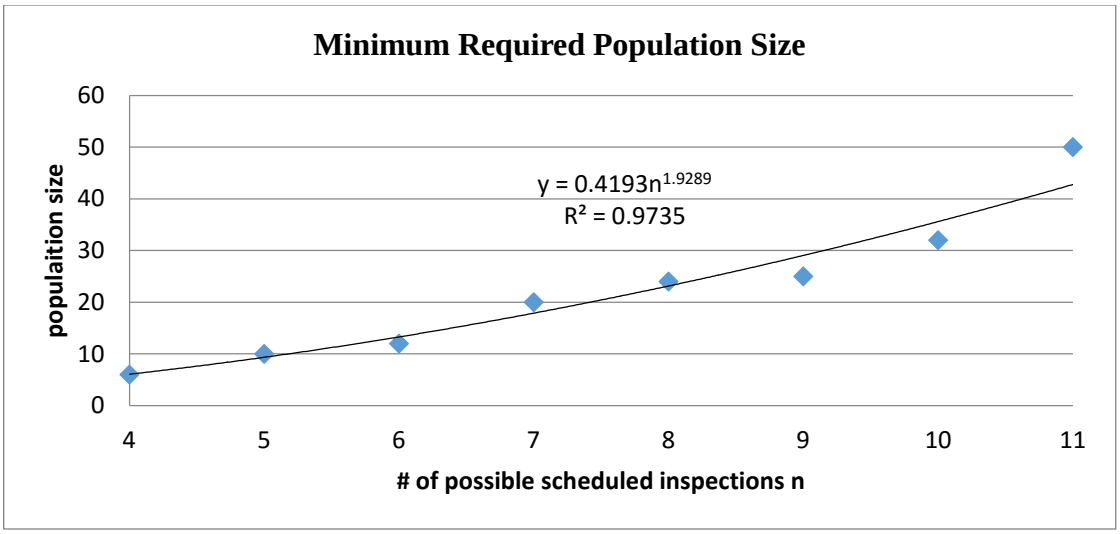

Figure 3.5. Minimum number of required population size vs. the number of possible scheduled inspections n.

The GA starts with $0.4193(n)^{1.9289}$ as the initial population size (which depends on n), and generates random inspection schemes in the population and calculates the total expected cost for each of them using (5) and the simulation model. Then, the GA generates the second generation of population and obtains the total expected costs for them. Some of the individuals (schemes) may appear at different generations. In this case, the simulation algorithm uses the existent results for those individuals instead of obtaining their expected total costs again, which helps significantly decrease the execution time.

In Table 3.6, the results of the GA for different values of n and T are presented. For cycle length $T = 13$, the population size is derived from the power equation that we obtained in Figure. 3.5. For other cycle lengths, the population sizes are taken from Table 3.5.

Table 3.6. The GA's results for different cycle lengths T.

Cycle length T (month)	Population size	Actual # schemes checked by the GA	Total time of running the GA (second)	$E\left[C^T_{S,\overrightarrow{x^*}}\right]$ ($)	Optimal scheme $\overrightarrow{x^*}$
5	6	8	11	2,027.43	01001
6	10	16	244	2,553.35	001001
7	12	24	642	3,092.53	0010101
8	20	35	1359	3,648.99	00010101
9	24	69	2953	4,193.31	001010101
10	25	79	3737	4,756.09	0010010101
11	32	107	4348	5,287.61	00100010101
12	50	121	5828	5,854.56	001010100101
13	51	214	12943	6,385.02	0001010010101

To compare the performance of the GA coupled with the simulation model with the case of using only the simulation model (i.e. examining all possible inspection schemes), we consider the case of $T = 12$ and $n = 11$, and compare the optimal results as well as the total running time from the simulation model solely with the integration of the GA and simulation model. The results are presented in Table 3.7.

Table 3.7. The results from the integration of the GA and the simulation model, and the simulation model solely.

	Simulation model solely	**Integration of the GA and simulation model**
Total running time	95,481 sec	6,727 sec
# of inspection schemes examined	2048	355

As it is shown in Table 3.7, the optimal cost, and the optimal scheme obtained from the simulation model solely, and the integration of the GA and simulation model are identical. The optimal cost of $5,854.56 is obtained for both methods. It should be noted that identical result is obtained if the population size in the GA is selected correctly. A too small population size may not result in the global optimal policy. The population size for our model depends on the number of possible inspections, which has been previously discussed. The genetic algorithm uses only 7.05% of the time required to run all the possible schemes. Moreover, instead of running the simulation for all possible inspection schemes (2,048 possibilities), the genetic algorithm can obtain the result by only checking 355 different schemes, which is just 17.3 % of the total possible inspection schemes. It is clearly shown that the GA reduces the number of inspection schemes that must be evaluated for obtaining the optimal scheme. In both methods, we should inspect the system at month third, fifth, seventh, and tenth.

Figure 3.6, presents the average tolerance between the individuals. The GA stops when the average tolerance between the individuals is less than e^{-5}. The distance between the expected costs of the best and worst inspection schemes in each generations decreases over time. The GA obtains the optimal solution after 12 generations.

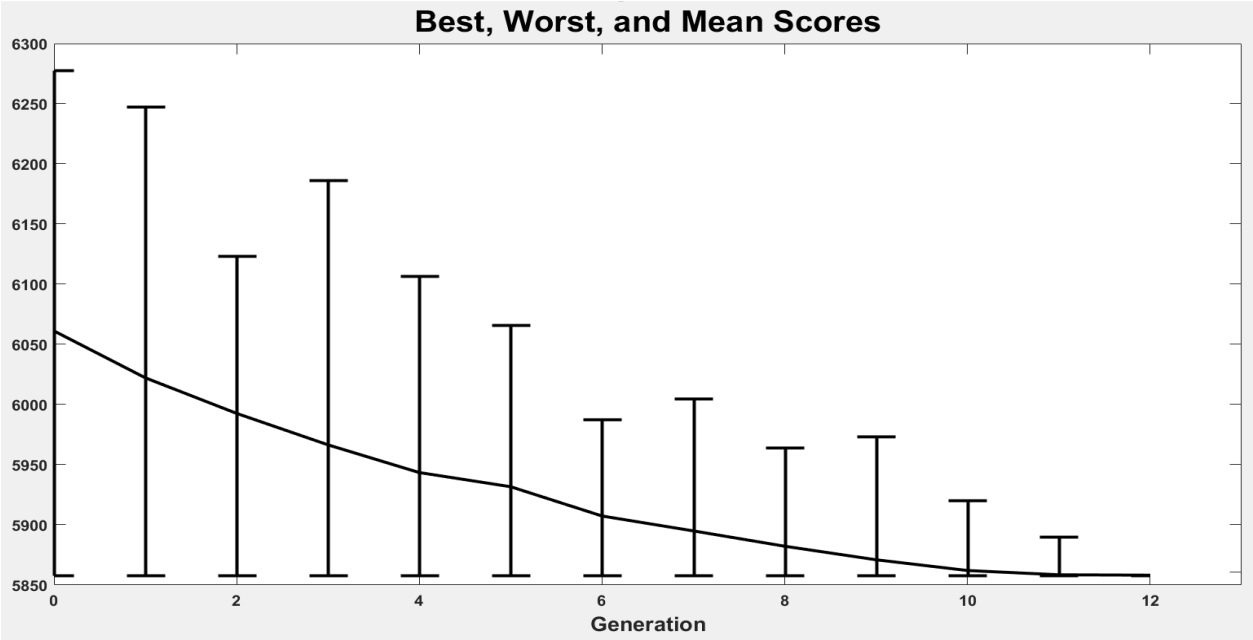

Figure 3.6. The fitness value for the best and the worst inspection schemes at each generation of the GA.

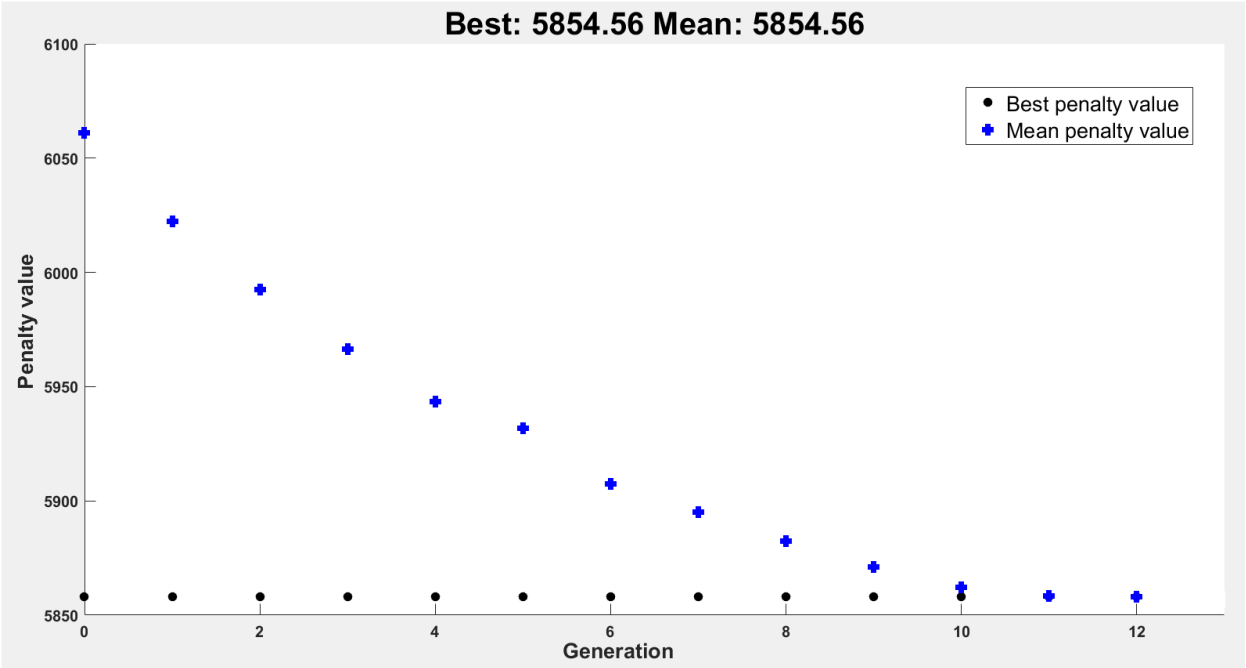

Figure 3.7. The best and average expected costs obtained by the GA at each generation.

Figure 3.7, shows the best and the average $E\left[C^{T}_{S,\overrightarrow{x^*}}\right]$ for each population at each generation. The GA is getting closer to the optimal solution, and the average expected costs of a generation is approaching the best fitness value in the generation, as the number of generation increases.

3.5. Sensitivity Analysis

We conduct the sensitivity analysis of our model to investigate how the optimal solution varies by changing the inputs parameters. For the system of hard-type and soft-type components, our different cases are considered, in which, some of the input parameters, i.e. the shape parameter, downtime penalty cost or inspection cost have been changed from the base parameters given in Table 3.1 and Table 3.2. The four cases (A to D) are presented in Table 3.8. In Case A, the downtime penalty cost of each component and the system inspection cost have been increased by $100. After these changes, the optimal inspection scheme (000010010101) with the expected cost of $7754.52 is obtained. Increasing the inspection cost results in fewer scheduled inspections as the optimal policy (3 inspections vs. 4 inspections in the numerical example given in 3.4). Moreover, since the components have now higher downtime penalty cost, they should be inspected every other month starting from month 8, when they are more likely to fail. In Case B, the shape parameter and the downtime penalty cost of each soft-type component have been increased by 1 and $100, respectively, and the inspection cost has been decreased by $50. This case results in the optimal scheme (001111111111), which implies that the system needs to be inspected every month starting from month 3, because the components are failing more frequently with a higher downtime penalty cost. In addition, the inspection cost is lower which justifies monthly inspection of the system. In Case C, there is no change in the downtime penalty cost of the components, but the inspection cost and the shape parameters are increased by 1 and $50, respectively. The optimal result of (000000100101) is obtained for this case. Comparing to the optimal scheme of the

numerical example in 3.4, i.e. (001010100101), the third and fifth inspections are eliminated in Case C due to higher inspection cost despite an increase in the shape parameters. Case D is identical to Case C, but the inspection cost is $30 higher in Case D. The optimal scheme (000000100011) is obtained for this case, which means the second inspection should be performed in month 11, although the total number of scheduled inspections is identical to Case C.

Table 3.8. Changes in the parameters of the system of hard-type and soft-type components.

Case	β_j	Downtime penalty (c_j^D)	Scheduled Inspection Cost $	$E\left[C_{S,\overrightarrow{x^*}}^T\right]$ ($)	Optimal scheme $\overrightarrow{x^*}$
A	---	+ 100	+ 100	7734.52	000010010101
B	+ 1	+ 100	+ 100	6613.85	001111111111
C	+ 1	---	---	6440.14	000000100101
D	+ 1	---	---	6480.46	000000100011

Chapter 4 – A k-out-of-m System

4.1. Simulation algorithm for a k-out-of-m system

Assume all m components of a k-out-of-m system follow a $NHPP$ with a power law intensity function, and the hazard rate of a component is $\lambda(x) = \frac{\beta}{\eta}\left(\frac{x}{\eta}\right)^{\beta-1}$, where x is the age of the component. The parameters of the power law β, η, scheduled inspection time, and T are all known. We assume the system starts with m new components whose initial ages are zero. Moreover, we define the age-dependent function $r(t) = ae^{-bt}$ for the simulation model which specifies the probability of minimal repair for a component with age t, and $1 - r(t)$ is the probability of replacement. We assume that a and b are given. Thus, the inputs of the simulation model are $\vec{x}, T, \tau, a, b, C^M, C^R, C^D, \beta$ and η, and $t_j, j = 1, \dots, m$ for all components, and the output of the simulation is $E\left[C^T_{S,\vec{x}}\right]$. To obtain the optimal inspection interval, we execute the simulation model for different inspection schemes and obtain their total expected costs. The optimal inspection scheme for the system is the one with the minimum total expected cost for the system.

The simulation model generates the first failure times for all components. We then find the minimum failure time of all components, and compare it with the next inspection interval. If the minimum failure time is less than the next inspection interval, the first failure occurs within the inspection interval. We bring forward the simulation clock to the first failure time, add 1 to the number of failures in the inspection interval, and deduct the minimum failure time from all

the survived components. Once more, the minimum failure time is found and compared with the next inspection interval. We follow the same steps as described above.

If the total number of failures in a scheduled inspection interval is less than $m - k + 1$, all the failed components are rectified at the scheduled inspection time, and their downtimes from the failure moment to the inspection time are measured and added to the total downtime. The total number of failures in the next inspection interval is set to zero, and the next failure times are generated for the rectified components. The period between the last failure time and the inspection time is then deducted from the failure times of all the survived components, and the simulation clock is brought forward to the inspection time.

If the total number of failures in a scheduled inspection interval reaches $m - k + 1$, the system fails and all the failures are detected and rectified. Thus, the total number of minimal repairs and replacements are updated accordingly, the current number of failures in the inspection interval is set to zero, and the next failure times are generated for the rectified components. The $(m - k + 1)^{th}$ failure time is then deducted from the survived components' failure times, and the simulation clock is brought forward to the $(m - k + 1)^{th}$ failure time.

Until the simulation clock reaches T , the same steps as described above are followed.

Similarly to the system discussed in Chapter 3, to find the optimal non-periodic inspection scheme for a k-out-of-m system more efficiently, we integrate the simulation model proposed here with the genetic algorithm.

4.2. Flowcharts of the Simulation model for a k-out-of-m system

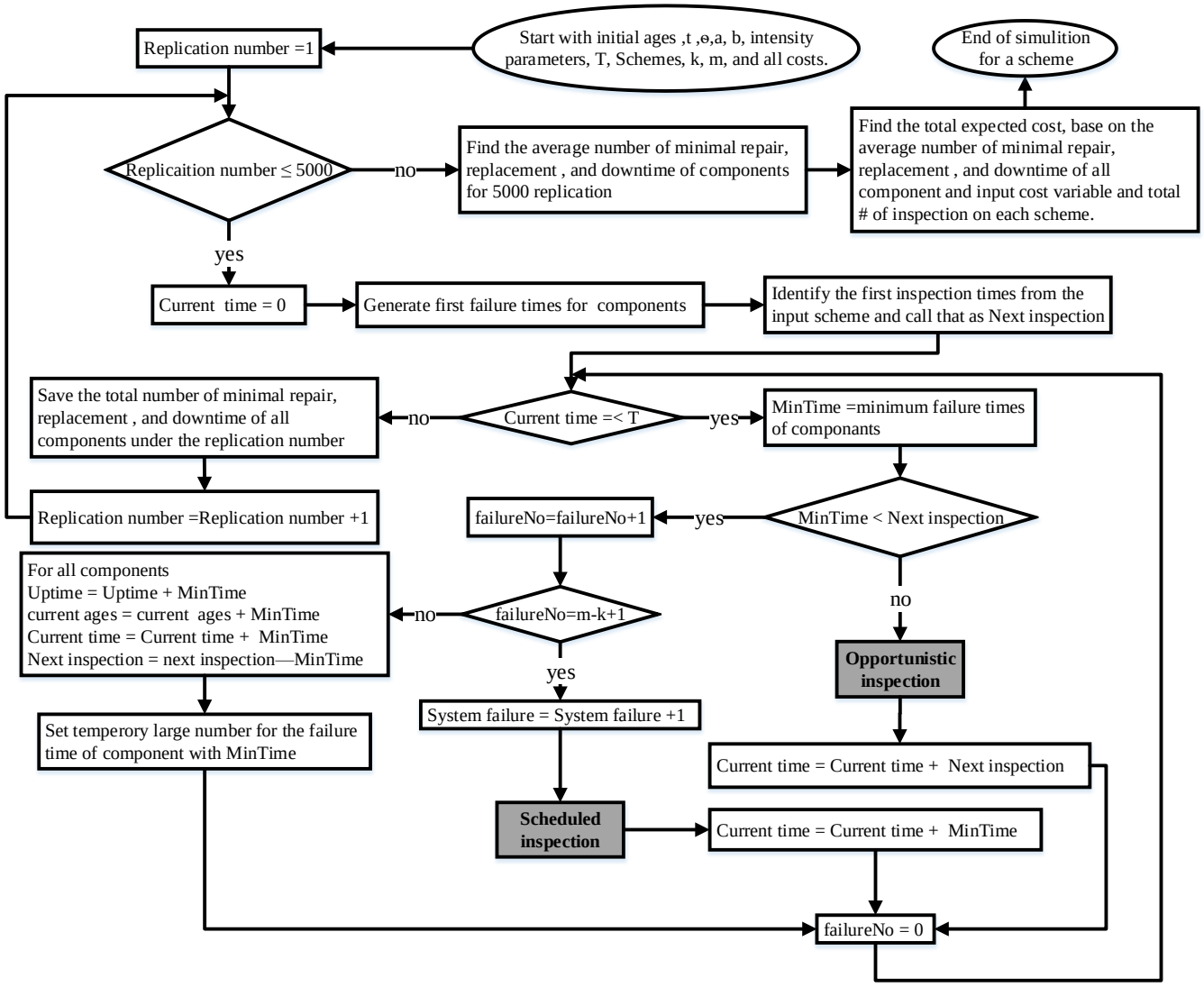

Figure 4.1. Simulation algorithm for a k-out-of-m system.

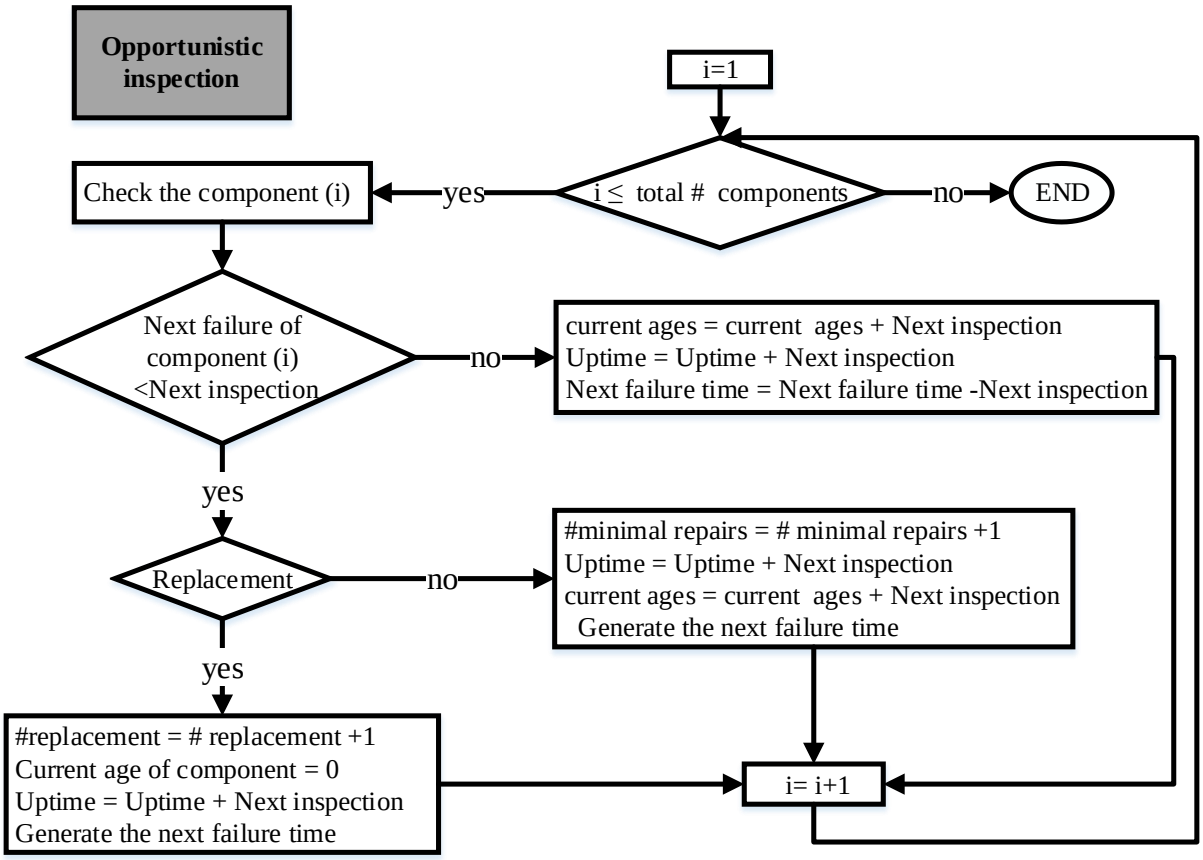

Figure 4.2. Algorithm for the inspection at a system failure for the k-out-of-m system.

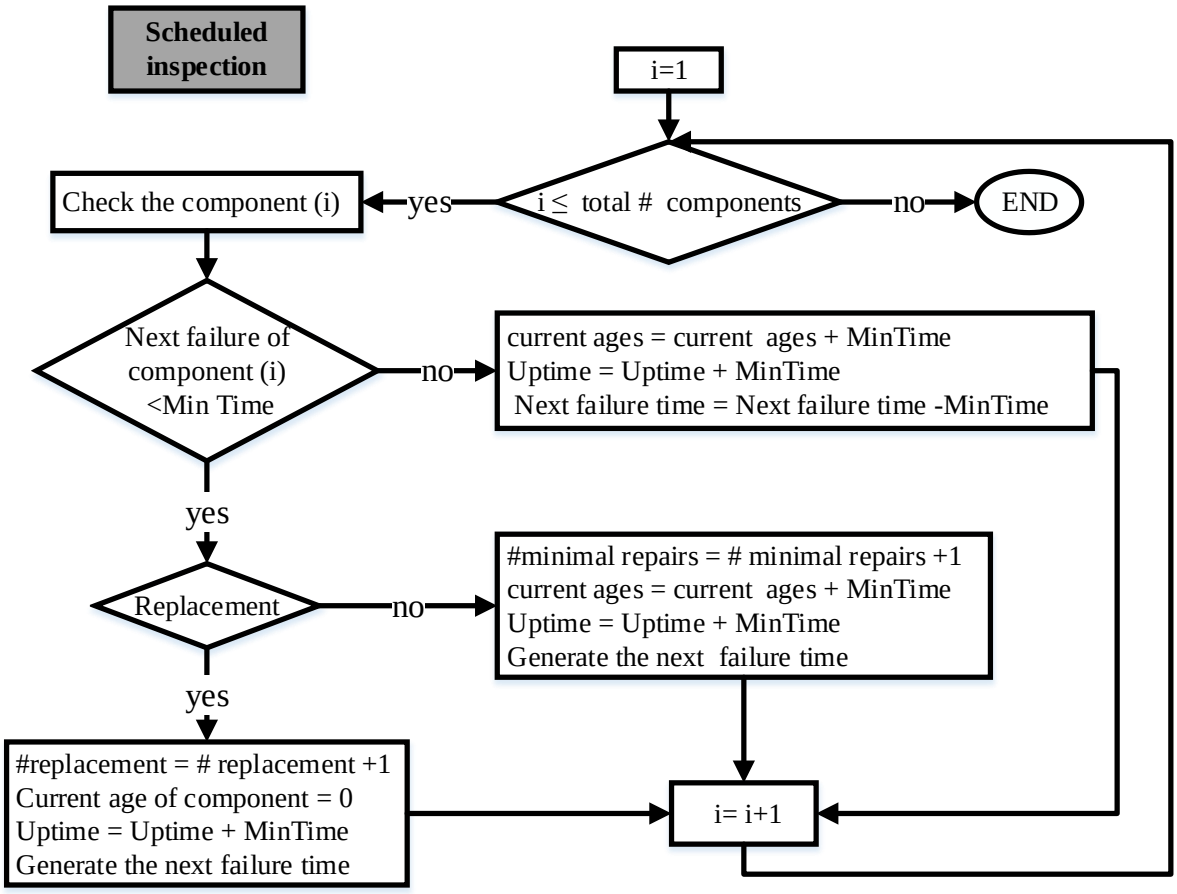

Figure 4.3. Algorithm for a scheduled inspection of the k-out-of-m system.

4.3. Numerical Example

We consider a k-out-of-m repairable system with the total of five identical components. We consider the failures of all components on this system also follow a $NHPP$ with the following power law intensity function.

The probability of minimal repairs for all components of age t is calculated from $(t) = ae^{-bt}$. The parameters of the power law process and parameters a and b, minimal repair, replacement, and downtime costs for the components are given in Table 4.1.

The cost of each scheduled inspection is $300, and the cost of each system failure including the inspection is $800.

Table 4.1. Parameters of the power law intensity function, and the probability of minimal repair, and required costs.

Minimal repair ($\mathbf{C^M}$)	Replacement ($\mathbf{C^R}$)	Downtime penalty ($\mathbf{C^D}$)	$\boldsymbol{\beta}$	$\boldsymbol{\eta}$	$\boldsymbol{a}$	$\boldsymbol{b}$
$70	$210	$200	1.3	3.5	0.9	0.2317

We are interested in finding the optimal non-periodic inspection scheme that results in the minimum total cost incurred over the lifecycle T for the k-out-of-m system. Similarly to the pervious system, we assume that at the beginning of the cycle all components are as-good-as new, so we set their initial ages to zero. We also assume that the minimum time unit at which a scheduled inspection can take place is one month. We assume the k-out-of-m system is 3-out-of-5, which implies the system is up while at least 3 components are up. The system will fail when three or more components are failed.

To obtain the optimal inspection scheme ($\overrightarrow{x^*}$) for cycle lengths $T = 12$, we calculate $E\left[C_{S,\vec{x}}^{T}\right]$ from the simulation model with 5,000 replications. We examine all possible inspection schemes for cycle length $T = 12$ with simulation model. The minimum expected cost calculated is \$8,539.17, and its corresponding optimal inspection scheme is $\overrightarrow{x^*} = (01010100101)$, which implies we should inspect the system at the second, forth, sixth, ninth, and eleventh months.

The expected number of minimal repairs, replacements, the expected total uptime of all 5 components, and the expected number of the system failures are given in Table 4.2.

Table 4.2. The results of the simulation model for the k-out-of-m system with cycle length T=12.

Optimal scheme	010101001011	Expected number of minimal repairs	7.5
Optimal cost	\$8,539.17	Expected number of replacements	8.4
System failure	3.1	Expected total uptime	50.0

Over lifecycle length $= 12$, we expect to have approximately 3 system failures, and five scheduled inspections. The system failure more likely will happen around month seven, and this is the reason why we do not have a scheduled inspection at this time, and instead we will have an opportunistic inspection. We also expect to have 7.5 minimal repairs, and 8.4 replacements to be performed for the failed components. This implies that in total we should have 16 component failures. The total expected uptime for this example is 50 months, which shows the components were up about 83.33% of the time. The optimal scheme (010101001011**)** results in the minimum total expected cost of the system.

To obtain the results mentioned above, we have to run the simulation model for all possible schemes for $T = 12$, which is 2,048 different schemes. Similarly to the previous model,

we use the genetic algorithm to obtain the optimal inspection scheme in shorter time. First, we derive the population size for the GA from the power equation obtained in Figure 3.5. In the case of $T = 12$ and $n = 11$, the population size is 50.

The optimal cost and the optimal inspection scheme obtained from the simulation model solely, and the integration of the GA and simulation model are exactly the same. Instead of running the simulation model for all possible inspection schemes (i.e. 2,048 possibilities), the genetic algorithm obtains the result by checking only 280 different schemes, which is just 13.7 % of the total possible inspection schemes. It is clearly shown that the GA reduces the number of inspection schemes that must be evaluated for obtaining the optimal scheme.

Figure 4.4 shows the average tolerance between the individuals. The GA stops when the average tolerance between the individuals is less than e^{-5}. The distance between the expected costs of the best and worst inspection schemes in each generations decreases over time. The GA obtains the optimal solution after 18 generations.

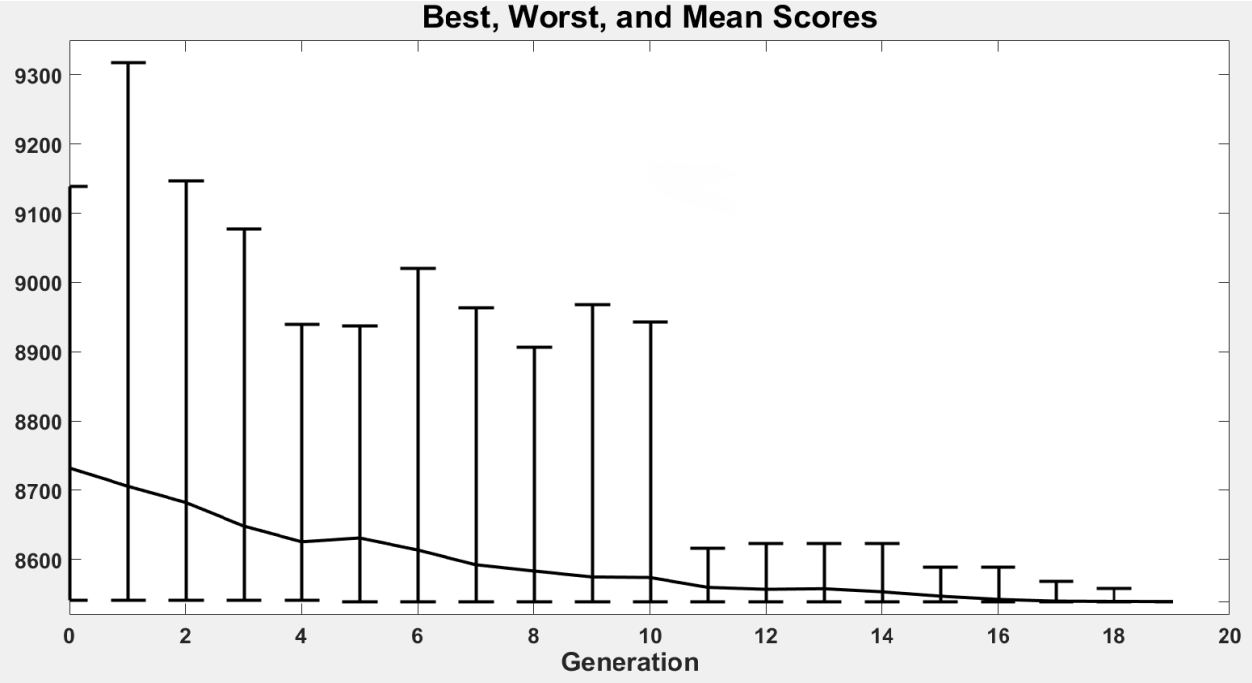

Figure 4.4. The fitness value for the best and the worst inspection schemes at each generation of the GA for the 3-out-of-5 system.

Figure 4.5 presents the best and the average $E\left[C^{T}_{S,\overrightarrow{x^*}}\right]$ for each population at each generation. The GA is getting closer to the optimal solution, and the average expected costs of a generation is approaching the best fitness value in the generation, as the number of generation increases.

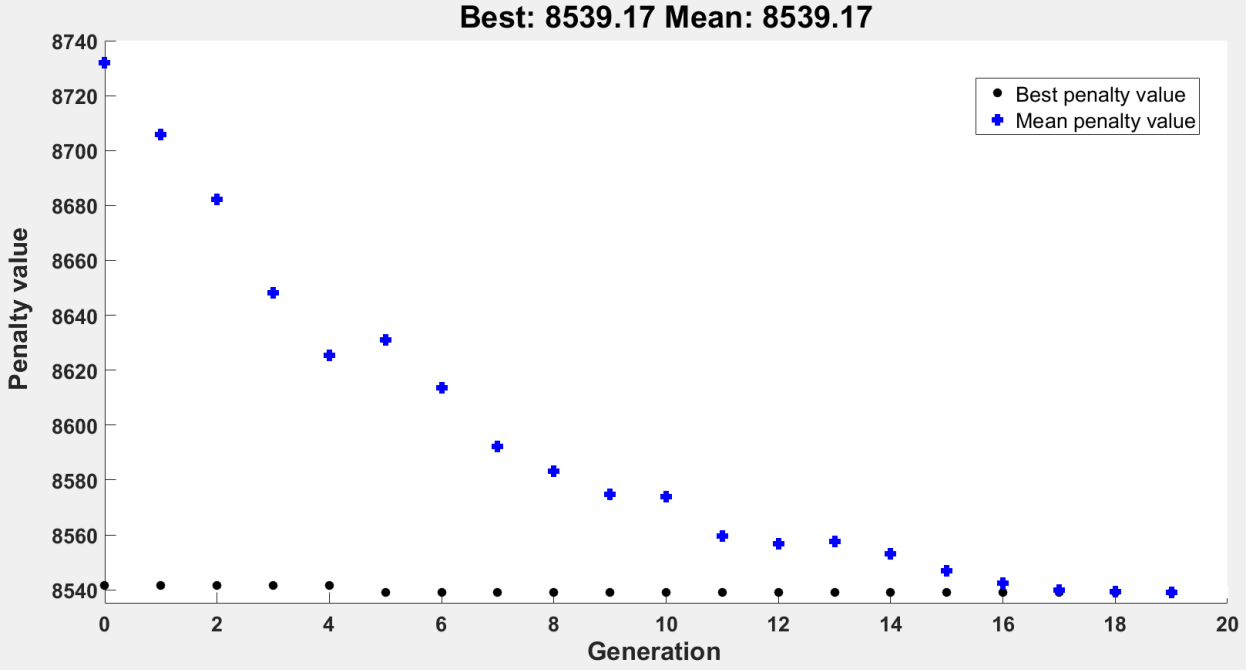

Figure 4.5. The best and average expected costs obtained by the GA at each generation for the 3-out-of-5 system.

4.4. Sensitivity analysis

For the k-out-of-m system, we also conduct sensitivity analyses on β, downtime and inspection costs to investigate the changes in the optimal inspection scheme. Similarly, four different cases are considered here (Table 4.3) in Case E, β and downtime cost have been increased by 1 and \$100, respectively, which results in (001111111111) as the optimal scheme due to higher number of failures and the components downtime. In Case F, β and downtime penalty cost have been increased by 1 and \$200, respectively, but the inspection cost has been decreased by \$80, which suggests more inspections to be done compared to Case E. In Case G, an increase of β by 2.2 and decrease of downtime penalty cost by \$200 are considered. The optimal inspection scheme for this case is (000110110011), which suggests the inspections should be done less frequently around the mid-lifecycle of the system. In Case H, an increase of β and inspection cost by 2.2 and \$50 are considered, respectively. The optimal scheme is (000101101011) in this case, which suggests no inspection in the first 3 months when the system is still new and inspection is more costly (compared to the inspection cost in the numerical example given in 4.2), but more inspections in the mid-life of the system when the components are more likely to be found failed.

Table 4.3. Changes in the parameters of the k-out-of-m system.

Case	β_j	Downtime penalty (c_j^D)	Scheduled Inspection Cost $	$E\left[C^T_{S,\overrightarrow{x^*}}\right]$ ($)	Optimal scheme $\overrightarrow{x^*}$
E	+ 1	+ 100	---	10400.2	001111111111
F	+ 1	+ 200	- 80	10407.7	011111111111
G	+ 2.2	- 200	---	10520.9	000110110011
H	+ 2.2	---	+ 50	10557.5	000101101011

Chapter 5 – Conclusions and Future Work

In this research, we first consider a multi-component system with soft-type and hard-type components and introduce a model to determine the optimal non-periodic inspection scheme for the system over a finite planning horizon. We then develop a model to find the optimal non-periodic inspection interval for a k-out-of-m system. We assume that the failures of the components for both systems follow a non-homogenous Poisson process (NHPP).

In the first system, the failures of hard-type components are detected and fixed instantaneously; however, the failures of soft-type components are only detected at opportunistic or at scheduled inspections. An opportunistic inspection occurs at a hard failure's time, at which the failed hard-type component and all other soft-type components are inspected. Soft failures do not stop the functioning of the system, but they may affect the performance of the system, so for the downtime of the soft-type components a penalty cost is incurred. Hard-type components do not have downtime, because they are detected and fixed immediately after a failure.

The proposed model considers minimal repair and replacement of a soft-type and a hard-type component as repair actions. Moreover, it assumes that at a scheduled inspection only soft-type components are inspected for possible soft failures. We develop a simulation model to calculate the total expected cost of each possible inspection scheme, and then combine the simulation model with the GA to efficiently find the optimal scheme.

For a k-out-of-m system, we assume that the system remains operational if the number of failed components is less than m − k + 1 in an inspection interval, and it fails, otherwise. The

system is inspected according to a non-periodic inspection scheme to detect the failed components. At a system failure, all the failed components are detected and rectified. We develop and use a simulation model to find the optimal non-periodic inspection scheme for the system to overcome the mathematical complexities of an analytical solution for the problem.

In summary, less frequent inspections are required when the components are still new, and the system should be inspected more often as the components get older. Non-periodic inspection is more efficient and practical when the inspection of a system is challenging or costly. For examples, underground mining equipment such as scooptrams, scissor lifts, and low profile trucks are not easily assessable for inspection. For these systems, non-periodic inspection policy is more practical and cost-effective since it eliminates unnecessary inspections when the system is relatively new. On the other hand, periodic inspection is more practical for organizations, such as hospitals, which are dealing with many different devices.

Another conclusion of this research is the integration of the simulation and the GA which significantly reduces the execution time for this kind of problem. The models proposed in this thesis can be extended to a more interesting and complicated case, where the combination of soft failures can be converted to a hard failure, if they left undetected. The optimal maintenance decisions can also be considered, in which we should also decide which components should be replaced or repaired if they fail. Our proposed models can also be extended for a load-sharing system [14, 16]. Another extension of the work is to compare the performance of other heuristic search algorithms, such as the simulated annealing with the GA. The joint optimization of a non-periodic inspection and inventory [69, 70] could be also an extension of this study.

Appendix A

A.1 Genetic algorithm code for the hard-soft system

```
function [Out1,Out2,Out3]=RunGA()
%*****************
function TotalCost =
SimulationNonPeriodic(InspectionScheme,TotalTime,Tau,SoftParams,HardParams,InspectionCo
st)

  function [Out1,Out2]=OneSimulationRun()

    NextFailuresSoft=zeros(1,SoftCompNo);
    CurrentAgesSoft=zeros(1,SoftCompNo);
    NextFailuresHard=zeros(1,HardCompNo);
    CurrentAgesHard=zeros(1,HardCompNo);
    OutSoft=zeros(SoftCompNo,3);
    OutHard=zeros(HardCompNo,2);

    for i=1:SoftCompNo;

      z= random('uniform',0,1);

NextFailuresSoft(i)=SoftParams(i,2)*power(power(CurrentAgesSoft(i)/SoftParams(i,2),SoftPara
ms(i,1))-log(z),1/SoftParams(i,1))-CurrentAgesSoft(i);
      i=i+1;

    end;

    for i=1:HardCompNo;
```

```
        z= random('uniform',0,1);

NextFailuresHard(i)=HardParams(i,2)*power(power(CurrentAgesHard(i)/HardParams(i,2),HardP
arams(i,1))-log(z),1/HardParams(i,1))-CurrentAgesHard(i);
        i=i+1;

    end;

    CurrentTime=0;
    InspectSchemeTemp=InspectionScheme;
    [NextInspect,NextInspectIdx]=max(InspectSchemeTemp);
    NextInspection=NextInspectIdx*Tau*NextInspect;

    while (CurrentTime < TotalTime) && (NextInspection>0)
      [MinHard,MinIdx]=min(NextFailuresHard);

      while MinHard < NextInspection

        for i=1:SoftCompNo;

          if NextFailuresSoft(i) < MinHard

            OutSoft(i,3)=OutSoft(i,3)+NextFailuresSoft(i);
            CurrentAgesSoft(i)=CurrentAgesSoft(i)+NextFailuresSoft(i);
            rf=SoftParams(i,3)*exp(-SoftParams(i,4)*CurrentAgesSoft(i));
            repairz= random('uniform',0,1);

            if repairz <= rf  % minimal repair
```

```
            OutSoft(i,1)=OutSoft(i,1)+1;

        else  % replacement

            OutSoft(i,2)=OutSoft(i,2)+1;
            CurrentAgesSoft(i)=0;

        end


NextFailuresSoft(i)=GenerateNextFailure(SoftParams(i,1),SoftParams(i,2),CurrentAgesSoft(i));

    else

        OutSoft(i,3)=OutSoft(i,3)+MinHard;
        CurrentAgesSoft(i)=CurrentAgesSoft(i)+MinHard;
        NextFailuresSoft(i)=NextFailuresSoft(i)-MinHard;

    end

end

CurrentTime=CurrentTime+MinHard;
CurrentAgesHard=CurrentAgesHard+MinHard;
rf=HardParams(MinIdx,3)*exp(-HardParams(MinIdx,4)*CurrentAgesHard(MinIdx));
repairz= random('uniform',0,1);

if repairz <= rf  % minimal repair
```

```
            OutHard(MinIdx,1)=OutHard(MinIdx,1)+1;
            CurrentAgesHard(MinIdx)=CurrentAgesHard(MinIdx)+MinHard;

        else  % replacement

            OutHard(MinIdx,2)=OutHard(MinIdx,2)+1;
            CurrentAgesHard(MinIdx)=0;

        end


NextfailureofHardfailed=GenerateNextFailure(HardParams(MinIdx,1),HardParams(MinIdx,2),Cu
rrentAgesHard(MinIdx));
            NextFailuresHard=NextFailuresHard-MinHard;
            NextFailuresHard(MinIdx)=NextfailureofHardfailed;
            NextInspection=NextInspection-MinHard;
            [MinHard,MinIdx]=min(NextFailuresHard);

        end


        for i=1:SoftCompNo;

            if NextFailuresSoft(i) < NextInspection

                CurrentAgesSoft(i)=CurrentAgesSoft(i)+NextFailuresSoft(i);
                rf=SoftParams(i,3)*exp(-SoftParams(i,4)*CurrentAgesSoft(i));
                repairz= random('uniform',0,1);
```

```
            OutSoft(i,3)=OutSoft(i,3)+NextFailuresSoft(i);

            if repairz <= rf  % minimal repair

               OutSoft(i,1)=OutSoft(i,1)+1;
               CurrentAgesSoft(i)=CurrentAgesSoft(i)+NextFailuresSoft(i);

            else  % replacement

               OutSoft(i,2)=OutSoft(i,2)+1;
               CurrentAgesSoft(i)=0;

            end


NextFailuresSoft(i)=GenerateNextFailure(SoftParams(i,1),SoftParams(i,2),CurrentAgesSoft(i));

         else

            OutSoft(i,3)=OutSoft(i,3)+NextInspection;
            CurrentAgesSoft(i)=CurrentAgesSoft(i)+NextInspection;
            NextFailuresSoft(i)=NextFailuresSoft(i)-NextInspection;

         end

      end

      CurrentAgesHard=CurrentAgesHard+NextInspection;
      CurrentTime=CurrentTime+NextInspection;
```

```
        NextFailuresHard=NextFailuresHard-NextInspection;
        InspectSchemeTemp(NextInspectIdx)=0;
        PrevInspection=NextInspectIdx*Tau;
        [NextInspect,NextInspectIdx]=max(InspectSchemeTemp);

        if NextInspectIdx*Tau*NextInspect > 0

           NextInspection=NextInspectIdx*Tau-PrevInspection;
           ss=length(InspectionScheme);

           if NextInspectIdx==length(InspectionScheme)

              NextInspection=TotalTime-PrevInspection;

           end;

        else

           NextInspection=0;

        end;

     end

     Out1=OutSoft;
     Out2=OutHard;

end
```

```
function nextfailure=GenerateNextFailure(beta,eta,CurrentAge)

   z= random('uniform',0,1);
   nextfailure=eta*power(power(CurrentAge/eta,beta)-log(z),1/beta)-CurrentAge;

end

InspectionScheme(length(InspectionScheme)+1)=1;
InspectionSchemeString='';

for i=1:length(InspectionScheme);

   InspectionSchemeString=strcat(InspectionSchemeString,num2str(InspectionScheme(i)));

end;

FoundInspection=0;

if ~isempty(SavedCosts > 0)

   FoundInspection=find(ismember(SavedPolicies,InspectionSchemeString));

end;

if FoundInspection > 0

   disp(SavedCosts(FoundInspection));
   disp(SavedPolicies(FoundInspection));
   OptPolicy=InspectionScheme;
```

```
    TotalCost=SavedCosts(FoundInspection);

else

    SoftCompNo=size(SoftParams,1);
    HardCompNo=size(HardParams,1);
    SimOutSoft=zeros(SoftCompNo,3);
    SimOutHard=zeros(HardCompNo,2);
    SimulationRuns=10;
    MinimalMatrix=zeros(SimulationRuns,SoftCompNo);
    ReplacementMatrix=zeros(SimulationRuns,SoftCompNo);
    UpMatrix=zeros(SimulationRuns,SoftCompNo);
    MinimalMatrixHard=zeros(SimulationRuns,HardCompNo);
    ReplacementMatrixHard=zeros(SimulationRuns,HardCompNo);
    s = RandStream('mt19937ar','Seed',0);
    RandStream.setGlobalStream(s);

    for iteration=1:SimulationRuns;

        [out1,out2]=OneSimulationRun();
        MinimalMatrix(iteration,:)=out1(:,1)';
        ReplacementMatrix(iteration,:)=out1(:,2)';
        UpMatrix(iteration,:)=out1(:,3)';
        SimOutSoft=SimOutSoft+out1;
        MinimalMatrixHard(iteration,:)=out2(:,1)';
        ReplacementMatrixHard(iteration,:)=out2(:,2)';
        SimOutHard=SimOutHard+out2;

    end;
```

```
for i=1:SoftCompNo;

   AvgSTDMinimals(i,1)=mean(MinimalMatrix(:,i));
   AvgSTDMinimals(i,2)=std(MinimalMatrix(:,i));
   AvgSTDMinimals(i,3)=std(MinimalMatrix(:,i))/sqrt(SimulationRuns);
   AvgSTDReplacement(i,1)=mean(ReplacementMatrix(:,i));
   AvgSTDReplacement(i,2)=std(ReplacementMatrix(:,i));
   AvgSTDReplacement(i,3)=std(ReplacementMatrix(:,i))/sqrt(SimulationRuns);
   AvgSTDUp(i,1)=mean(UpMatrix(:,i));
   AvgSTDUp(i,2)=std(UpMatrix(:,i));
   AvgSTDUp(i,3)=std(UpMatrix(:,i))/sqrt(SimulationRuns);

end;

for i=1:HardCompNo;

   AvgSTDMinimalsHard(i,1)=mean(MinimalMatrixHard(:,i));
   AvgSTDMinimalsHard(i,2)=std(MinimalMatrixHard(:,i));
   AvgSTDMinimalsHard(i,3)=std(MinimalMatrixHard(:,i))/sqrt(SimulationRuns);
   AvgSTDReplacementHard(i,1)=mean(ReplacementMatrixHard(:,i));
   AvgSTDReplacementHard(i,2)=std(ReplacementMatrixHard(:,i));
   AvgSTDReplacementHard(i,3)=std(ReplacementMatrixHard(:,i))/sqrt(SimulationRuns);

end;

SimOutSoft=SimOutSoft/SimulationRuns;
Out1=AvgSTDMinimals;
Out2=AvgSTDReplacement;
```

```
    Out3=AvgSTDUp;
    Out4=AvgSTDMinimalsHard;
    Out5=AvgSTDReplacementHard;
    CountInspections = sum(sum(InspectionScheme));
    TotalCost=0;

    for i=1:SoftCompNo;

TotalCost=TotalCost+AvgSTDMinimals(i,1)*SoftParams(i,5)+AvgSTDReplacement(i,1)*SoftPara
ms(i,6)+SoftParams(i,7)*(TotalTime-AvgSTDUp(i,1));

    end;

    TotalCost=TotalCost+InspectionCost*CountInspections+2850;
    OptPolicy=InspectionScheme;
    SavedPoliciesInx=SavedPoliciesInx+1;
    InspectionSchemeString='';

    for i=1:length(InspectionScheme);

      InspectionSchemeString=strcat(InspectionSchemeString,num2str(InspectionScheme(i)));

    end;

    SavedPolicies{SavedPoliciesInx}=InspectionSchemeString;
    SavedCosts(SavedPoliciesInx)=TotalCost;

  end
```

```
end

%*****************

TotalTime=12;
Tau=1;
CI=280;
Data = xlsread('c:\Data\Yassin\excel\SoftComponents.xls');
RecordsNo=numel(Data(:,2));
rowindx=0;

while (rowindx < RecordsNo)

   rowindx=rowindx+1;
   SoftParams(rowindx,1)=Data(rowindx,1); % beta %
   SoftParams(rowindx,2)=Data(rowindx,2); % eta %
   SoftParams(rowindx,3)=Data(rowindx,3);
   SoftParams(rowindx,4)=Data(rowindx,4);
   SoftParams(rowindx,5)=Data(rowindx,5);  % cost of minimal repair%
   SoftParams(rowindx,6)=Data(rowindx,6); % cost of replacement%
   SoftParams(rowindx,7)=Data(rowindx,7);% cost of downtime%

end

SoftCompNo=rowindx;
Data = xlsread('c:\Data\Yassin\excel\HardComponents.xls');
RecordsNo=numel(Data(:,2));
rowindx=0;
```

```
while (rowindx < RecordsNo)

   rowindx=rowindx+1;
   HardParams(rowindx,1)=Data(rowindx,1);
   HardParams(rowindx,2)=Data(rowindx,2);
   HardParams(rowindx,3)=Data(rowindx,3);
   HardParams(rowindx,4)=Data(rowindx,4);
   HardParams(rowindx,5)=Data(rowindx,5); % cost of minimal repair%
   HardParams(rowindx,6)=Data(rowindx,6); % cost of replacement%

end

HardCompNo=rowindx;

if rem(TotalTime,Tau)==0

   MaxInspections=TotalTime/Tau-1;

else

   MaxInspections=floor(TotalTime/Tau);

end;

global SavedPolicies;
global SavedCosts;
global SavedPoliciesInx;
SavedPoliciesInx=0;
```

```
Policies=zeros(1,MaxInspections);
LowerBounds=zeros(1,MaxInspections);
UpperBounds=ones(1,MaxInspections);
IntegerVars=cumsum(ones(1,MaxInspections));
FitnessFunction = @(Policies)
SimulationNonPeriodic(Policies,TotalTime,Tau,SoftParams,HardParams,CI);
opts = gaoptimset('PlotFcns',{@gaplotbestf,@gaplotrange});
opts = gaoptimset(opts,'PopulationSize',150);
opts = gaoptimset(opts,'Generations',50,'StallGenLimit', 15);
opts = gaoptimset (opts, 'EliteCount' , 1 );
opts = gaoptimset(opts,'TolFun', 1e-5);
global OptPolicy;
[OptimalPolicy,MinCost] =
ga(FitnessFunction,MaxInspections,[],[],[],[],LowerBounds,UpperBounds,[],IntegerVars,opts);
disp(OptPolicy);
disp(OptimalPolicy);
disp(MinCost);
disp(SavedPoliciesInx);

end
```

A.2 Genetic algorithm code for the k-out-of-m system

```
function [Out1 ,Out2,Out3,out4]=RunknGA()
%*****************
function TotalCost =
SimulationNonPeriodic(k,n,InspectionScheme,TotalTime,Tau,CompParams,InspectionCost)

  function [Out1]=OneSimulationRun()

    NextFailuresSoft=zeros(1,CompNo);
    CurrentAgesSoft=zeros(1,CompNo);
    OutSoft=zeros(CompNo,3);

    for i=1:CompNo;

      z= random('uniform',0,1);

NextFailures(i)=CompParams(i,2)*power(power(CurrentAgesSoft(i)/CompParams(i,2),CompPar
ams(i,1))-log(z),1/CompParams(i,1))-CurrentAgesSoft(i);
      i=i+1;

    end;

    CurrentTime=0;
    InspectSchemeTemp=InspectionScheme;
    [NextInspect,NextInspectIdx]=max(InspectSchemeTemp);
    NextInspection=NextInspectIdx*Tau*NextInspect;
    OpportunisticInspectionNo=0;
    failureNo=0;
```

```
while (CurrentTime < TotalTime) && (NextInspection>0)

   [MinCompFailure,MinIdx]=min(NextFailures);

   while (MinCompFailure ~= 10000) && (MinCompFailure < NextInspection)
      failureNo=failureNo+1;

      if failureNo >= n-k+1

         failureNo=0;
         OpportunisticInspectionNo=OpportunisticInspectionNo+1;

         for i=1:CompNo;

            if i== MinIdx

               OutComp(i,3)=OutComp(i,3)+MinCompFailure;
               CurrentAgesSoft(i)=CurrentAgesSoft(i)+MinCompFailure;
               rf=CompParams(i,3)*exp(-CompParams(i,4)*CurrentAgesSoft(i));
               repairz= random('uniform',0,1);

               if repairz <= rf  % minimal repair

                  OutComp(i,1)=OutComp(i,1)+1;

               else  % replacement

                  OutComp(i,2)=OutComp(i,2)+1;
                  CurrentAgesComp(i)=0;
```

```
            end

NextFailures(i)=GenerateNextFailure(CompParams(i,1),CompParams(i,2),CurrentAgesComp(i));

        else

            if NextFailures(i) ~= 10000

                OutComp(i,3)=OutComp(i,3)+MinCompFailure;
                CurrentAgesComp(i)=CurrentAgesComp(i)+MinCompFailure;
                NextFailures(i)=NextFailures(i)-MinCompFailure;

            else

                rf=CompParams(i,3)*exp(-CompParams(i,4)*CurrentAgesComp(i));
                repairz= random('uniform',0,1);

                if repairz <= rf  % minimal repair

                    OutComp(i,1)=OutComp(i,1)+1;

                else  % replacement

                    OutComp(i,2)=OutComp(i,2)+1;
                    CurrentAgesComp(i)=0;

                end
```

```
NextFailures(i)=GenerateNextFailure(CompParams(i,1),CompParams(i,2),CurrentAgesComp(i));

            end

        end;

    end;

else

    for i=1:CompNo;

        if i== MinIdx

            OutComp(i,3)=OutComp(i,3)+MinCompFailure;
            CurrentAgesComp(i)=CurrentAgesComp(i)+MinCompFailure;
            NextFailures(i)=10000;

        else

            if NextFailures(i) ~= 10000

                OutComp(i,3)=OutComp(i,3)+MinCompFailure;
                CurrentAgesComp(i)=CurrentAgesComp(i)+MinCompFailure;
                NextFailures(i)=NextFailures(i)-MinCompFailure;

            end
```

```
        end;

    end;

  end;

  CurrentTime=CurrentTime+MinCompFailure;
  NextInspection=NextInspection-MinCompFailure;
  [MinCompFailure,MinIdx]=min(NextFailures);

end;

for i=1:CompNo;

  if NextFailures(i) ~= 10000

    OutComp(i,3)=OutComp(i,3)+NextInspection;
    CurrentAgesComp(i)=CurrentAgesComp(i)+NextInspection;
    NextFailures(i)=NextFailures(i)-NextInspection;

  else

    rf=CompParams(i,3)*exp(-CompParams(i,4)*CurrentAgesComp(i));
    repairz= random('uniform',0,1);

    if repairz <= rf  % minimal repair

      OutComp(i,1)=OutComp(i,1)+1;
```

```
            else  % replacement

               OutComp(i,2)=OutComp(i,2)+1;
               CurrentAgesComp(i)=0;

            end


NextFailures(i)=GenerateNextFailure(CompParams(i,1),CompParams(i,2),CurrentAgesComp(i));

         end

      end;

      failureNo=0;
      CurrentTime=CurrentTime+NextInspection;
      InspectSchemeTemp(NextInspectIdx)=0;
      PrevInspection=NextInspectIdx*Tau;
      [NextInspect,NextInspectIdx]=max(InspectSchemeTemp);

      if NextInspectIdx*Tau*NextInspect > 0

         NextInspection=NextInspectIdx*Tau-PrevInspection;

         if NextInspectIdx==length(InspectionScheme)

            NextInspection=TotalTime-PrevInspection;

         end;
```

```
        else

            NextInspection=0;

        end;

    end;

    Out1=OutComp;

end

function nextfailure=GenerateNextFailure(beta,eta,CurrentAge)

    z= random('uniform',0,1);
    nextfailure=eta*power(power(CurrentAge/eta,beta)-log(z),1/beta)-CurrentAge;

end

InspectionScheme(length(InspectionScheme)+1)=1;
InspectionSchemeString='';

for i=1:length(InspectionScheme);

    InspectionSchemeString=strcat(InspectionSchemeString,num2str(InspectionScheme(i)));

end;
```

```
FoundInspection=0;

if ~isempty(SavedCosts > 0)

   FoundInspection=find(ismember(SavedPolicies,InspectionSchemeString));

end;

if FoundInspection > 0

   disp(SavedCosts(FoundInspection));
   disp(SavedPolicies(FoundInspection));
   OptPolicy=InspectionScheme;
   TotalCost=SavedCosts(FoundInspection);

else

   CompNo=size(CompParams,1);
   SimOut=zeros(CompNo,3);
   TOpportunisticInspectionNo=0;
   SimulationRuns=10000;
   MinimalMatrix=zeros(SimulationRuns,CompNo);
   ReplacementMatrix=zeros(SimulationRuns,CompNo);
   UpMatrix=zeros(SimulationRuns,CompNo);
   s = RandStream('mt19937ar','Seed',0);
   RandStream.setGlobalStream(s);

   for iteration=1:SimulationRuns;
```

```
    [out1]=OneSimulationRun();
    MinimalMatrix(iteration,:)=out1(:,1)';
    ReplacementMatrix(iteration,:)=out1(:,2)';
    UpMatrix(iteration,:)=out1(:,3)';
    SimOut=SimOut+out1;
    TOpportunisticInspectionNo=OpportunisticInspectionNo+TOpportunisticInspectionNo;

end;

for i=1:CompNo;

    AvgSTDMinimals(i,1)=mean(MinimalMatrix(:,i));
    AvgSTDMinimals(i,2)=std(MinimalMatrix(:,i));
    AvgSTDMinimals(i,3)=std(MinimalMatrix(:,i))/sqrt(SimulationRuns);
    AvgSTDReplacement(i,1)=mean(ReplacementMatrix(:,i));
    AvgSTDReplacement(i,2)=std(ReplacementMatrix(:,i));
    AvgSTDReplacement(i,3)=std(ReplacementMatrix(:,i))/sqrt(SimulationRuns);
    AvgSTDUp(i,1)=mean(UpMatrix(:,i));
    AvgSTDUp(i,2)=std(UpMatrix(:,i));
    AvgSTDUp(i,3)=std(UpMatrix(:,i))/sqrt(SimulationRuns);

end;

SimOut=SimOut/SimulationRuns;
Out1=AvgSTDMinimals;
Out2=AvgSTDReplacement;
Out3=AvgSTDUp;
Minimals=sum(AvgSTDMinimals(:,1));
Replacement=sum(AvgSTDReplacement(:,1));
```

```
    Up=sum(AvgSTDUp(:,1));
    TOpportunisticInspectionNo=TOpportunisticInspectionNo/SimulationRuns;
    CountInspections = sum(sum(InspectionScheme));
    TotalCost=0;

    for i=1:CompNo;

TotalCost=TotalCost+AvgSTDMinimals(i,1)*CompParams(i,5)+AvgSTDReplacement(i,1)*CompPa
rams(i,6)+CompParams(i,7)*(TotalTime-AvgSTDUp(i,1));

    end;

TotalCost=TotalCost+InspectionCost*CountInspections+TOpportunisticInspectionNo*800+1000
;
    OptPolicy=InspectionScheme;
    SavedPoliciesInx=SavedPoliciesInx+1;
    InspectionSchemeString='';

    for i=1:length(InspectionScheme);

       InspectionSchemeString=strcat(InspectionSchemeString,num2str(InspectionScheme(i)));

    end;

    SavedPolicies{SavedPoliciesInx}=InspectionSchemeString;
    SavedCosts(SavedPoliciesInx)=TotalCost;
```

```
    end

end

%*****************

TotalTime=12;
Tau=1;
k=3;
InspectionCost=300;

if rem(TotalTime,Tau)==0

    MaxInspections=TotalTime/Tau-1;

else

    MaxInspections=floor(TotalTime/Tau);

end

Data = xlsread('c:\Data\Yassin\excel\data.xls');
RecordsNo=numel(Data(:,2));
rowindx=0;

while (rowindx < RecordsNo)

    rowindx=rowindx+1;
    CompParams(rowindx,1)=Data(rowindx,1); % beta %
```

```
    CompParams(rowindx,2)=Data(rowindx,2); % eta %
    CompParams(rowindx,3)=Data(rowindx,3);
    CompParams(rowindx,4)=Data(rowindx,4);
    CompParams(rowindx,5)=Data(rowindx,5);  % cost of minimal repair%
    CompParams(rowindx,6)=Data(rowindx,6); % cost of replacement%
    CompParams(rowindx,7)=Data(rowindx,7);% cost of downtime%

end

CompNo=rowindx;
n=CompNo;
global SavedPolicies;
global SavedCosts;
global SavedPoliciesInx;
SavedPoliciesInx=0;
Policies=zeros(1,MaxInspections);
LowerBounds=zeros(1,MaxInspections);
UpperBounds=ones(1,MaxInspections);
IntegerVars=cumsum(ones(1,MaxInspections));
FitnessFunction = @(Policies)
SimulationNonPeriodic(k,n,Policies,TotalTime,Tau,CompParams,InspectionCost);
opts = gaoptimset('PlotFcns',{@gaplotbestf,@gaplotrange});
opts = gaoptimset(opts,'PopulationSize',60);
opts = gaoptimset(opts,'Generations',50,'StallGenLimit', 20);
opts = gaoptimset (opts, 'EliteCount' , 1 );
opts = gaoptimset(opts,'TolFun', 1e-5);
global OptPolicy;
[OptimalPolicy,MinCost] =
ga(FitnessFunction,MaxInspections,[],[],[],[],LowerBounds,UpperBounds,[],IntegerVars,opts);
```

```
disp(OptPolicy);
disp(OptimalPolicy);
disp(MinCost);
disp(SavedPoliciesInx);
 end
```

References

[1] Taghipour S, Banjevic D, Jardine AK. Periodic inspection optimization model for a complex repairable system. Reliability Engineering & System Safety. 2010 Sep 30;95(9):944-52.

[2] Moubray J. Reliability centered maintenance. Industrial Press;1997.

[3] Taghipour S. Reliability and maintenance of medical devices (Doctoral dissertation, University of Toronto);2011.

[4] Taghipour S, Banjevic D, Jardine AK. Risk-based inspection and maintenance for medical equipment. In IIE Annual Conference. Proceedings 2008 Jan 1: 104-9. Institute of Industrial Engineers-Publisher.

[5] Meeker WQ, Escobar LA. Statistical methods for reliability data. John Wiley & Sons; 2014 Aug 21.

[6] Taghipour S, Banjevic D, Jardine AK. Reliability analysis of maintenance data for complex medical devices. Quality and Reliability Engineering International. 2011 Feb 1;27(1):71-84.

[7] Taghipour S, Banjevic D. Periodic inspection optimization models for a repairable system subject to hidden failures. Reliability, IEEE Transactions on. 2011 Mar;60(1):275-85.

[8] Wang H. A survey of maintenance policies of deteriorating systems. European journal of operational research. 2002 Jun 16;139(3):469-89.

[9] Wang H, Pham H. Reliability and optimal maintenance. Springer Science & Business Media; 2006 Sep 27.

[10] Dekker R, Wildeman RE, Van der Duyn Schouten FA. A review of multi-component maintenance models with economic dependence. Mathematical Methods of Operations Research. 1997 Oct 1;45(3):411-35.

[11] Özekici S. Optimal periodic replacement of multicomponent reliability systems. Operations Research. 1988 Aug;36(4):542-52.

[12] Wang KH, Kuo CC. Cost and probabilistic analysis of series systems with mixed standby components. Applied Mathematical Modelling. 2000 Oct 31;24(12):957-67.

[13] Zhang YL, Wu S. Reliability analysis for ak/n (F) system with repairable repair-equipment. Applied Mathematical Modelling. 2009 Jul 31;33(7):3052-67.

[14] Kassaei ML, Taghipour S. Inspection optimization model for a k-out-of-n load-sharing system with dependent components. In IIE Annual Conference. Proceedings 2013 Jan 1: 3282-90. Institute of Industrial Engineers-Publisher.

[15] Taghipour S, Kassaei ML. Periodic Inspection Optimization of a k-Out-of-n Load-Sharing System. Reliability, IEEE Transactions on. 2015 Sep;64(3):1116-27.

[16] Taghipour S. Optimal inspection model for a load-sharing redundant system. In Reliability and Maintainability Symposium (RAMS), 2014 Annual. 2014 Jan 27:1-5. IEEE.

[17] Cho DI, Parlar M. A survey of maintenance models for multi-unit systems. European Journal of Operational Research. 1991 Mar 6;51(1):1-23.

[18] S Sheu SH, Griffith WS. Optimal age-replacement policy with age-dependent minimal-repair and random-leadtime. Reliability, IEEE Transactions on. 2001 Sep;50(3):302-9.

[19] Chien YH, Sheu SH. Extended optimal age-replacement policy with minimal repair of a system subject to shocks. European journal of operational research. 2006 Oct 1;174(1):169-81.

[20] Makis V, Jardine AK. Optimal replacement policy for a general model with imperfect repair. Journal of the Operational Research Society. 1992 Feb 1;122(1):111-20.

[21] Wang GJ, Zhang YL. A bivariate mixed policy for a simple repairable system based on preventive repair and failure repair. Applied Mathematical Modelling. 2009 Aug 31;33(8):3354-9.

[22] Zhang YL, Wang GJ. A geometric process repair model for a series repairable system with k dissimilar components. Applied Mathematical Modelling. 2007 Sep 30;31(9):1997-2007.

[23] Wu Q, Zhang J. A bivariate replacement policy for a cold standby system under poisson shocks. American Journal of Mathematical and Management Sciences. 2013 Jul 3;32(3):145-77.

[24] Coria VH, Maximov S, Rivas-Dávalos F, Melchor CL, Guardado JL. Analytical method for optimization of maintenance policy based on available system failure data. Reliability Engineering & System Safety. 2015 Mar 31;135(1):55-63.

[25] Pan ES, Liao WZ, Zhuo ML. Periodic preventive maintenance policy with infinite time and limit of reliability based on health index. Journal of Shanghai Jiaotong University (Science). 2010 Apr 1;15(1):231-5.

[26] Lienhardt B, Hugues E, Bes C, Noll D. Failure-finding frequency for a repairable system subject to hidden failures. Journal of Aircraft. 2008 Sep 1;45(5):1804-9.

[27] Taghipour S, Banjevic D, Jardine AK. Periodic inspection optimization model for a complex repairable system. Reliability Engineering & System Safety. 2010 Sep 30;95(9):944-52.

[28] Sheu SH, Li SH, Chang CC. A generalised maintenance policy with age-dependent minimal repair cost for a system subject to shocks under periodic overhaul. International Journal of Systems Science. 2012 Jun 1;43(6):1007-13.

[29] Baohe S. An optimal inspection and diagnosis policy for a multi-mode system. Reliability Engineering & System Safety. 2002 May 31;76(2):181-8.

[30] Zhou XJ, Lu ZQ, Xi LF, Lee J. Opportunistic preventive maintenance optimization for multi-unit series systems with combing multi-preventive maintenance techniques. Journal of Shanghai Jiaotong University (Science). 2010 Oct 30;15(1):513-8.

[31] Dagpunar JS. A maintenance model with opportunities and interrupt replacement options. Journal of the Operational Research Society. 1996 Nov 30;1(1):1406-9.

[32] Zhu W, Fouladirad M, Berenguer C. A reactive multi-component maintenance policy for offshore wind turbines. In European Safety and Reliability Conference-ESREL 2013 2014:811-7). Taylor & Francis (CRC Press/Balkema).

[33] Cui L, Li H. Opportunistic maintenance for multi-component shock models. Mathematical Methods of Operations Research. 2006 Jul 1;63(3):493-511.

[34] Aven T, Dekker R. A useful framework for optimal replacement models. Reliability Engineering & System Safety. 1997 Oct 31;58(1):61-7.

[35] S Taghipour S, Banjevic D. Optimum inspection interval for a system under periodic and opportunistic inspections. Iie Transactions. 2012 Nov 1;44(11):932-48.

[36] Taghipour S, Banjevic D. Optimal inspection of a complex system subject to periodic and opportunistic inspections and preventive replacements. European Journal of Operational Research. 2012 Aug 1;220(3):649-60.

[37] Huynh KT, Barros A, Berenguer C, Castro IT. A periodic inspection and replacement policy for systems subject to competing failure modes due to degradation and traumatic events. Reliability Engineering & System Safety. 2011 Apr 30;96(4):497-508.

[38] Courtois PJ, Delsarte P. On the optimal scheduling of periodic tests and maintenance for reliable redundant components. Reliability Engineering & System Safety. 2006 Jan 31;91(1):66-72.

[39] Mohandas K, Chaudhuri D, Rao BV. Optimal periodic replacement for a deteriorating production system with inspection and minimal repair. Reliability Engineering & System Safety. 1992 Dec 31;37(1):73-7.

[40] Scarf PA, Wang W, Laycock PJ. A stochastic model of crack growth under periodic inspections. Reliability Engineering & System Safety. 1996 Mar 31;51(3):331-9.

[41] Kapur PK, Butani NL. Optimum inspection policies for a computer system with hidden failure. International journal of systems science. 1987 Jan 1;18(4):601-9.

[42] Zequeira RI, Berenguer C. An inspection & imperfect maintenance model for a system with two competing failure modes. In Fault Detection, Supervision and Safety of Technical Processes 2006 Aug 29: 6(1):932-7.

[43] Zhao X, Al-Khalifa KN, Nakagawa T. Approximate methods for optimal replacement, maintenance, and inspection policies. Reliability Engineering & System Safety. 2015 Dec 31;144(1):68-73.

[44] J Vaurio JK. Optimization of test and maintenance intervals based on risk and cost. Reliability Engineering & System Safety. 1995 Dec 31;49(1):23-36.

[45] Rezaei E, Imani DM. Maintenance Risk Based Inspection Optimization Model in Multi-Component Repairable System with Economic Failure Interaction. In Current Trends in Reliability, Availability, Maintainability and Safety 2016 Jan 1:611-20. Springer International Publishing.

[46] Wang W, Banjevic D, Pecht M. A multi-component and multi-failure mode inspection model based on the delay time concept. Reliability Engineering & System Safety. 2010 Aug 31;95(8):912-20.

[47] Wang W. A joint spare part and maintenance inspection optimisation model using the Delay-Time concept. Reliability Engineering & System Safety. 2011 Nov 30;96(11):1535-41.

[48] Golmakani HR, Moakedi H. Periodic inspection optimization model for a multi-component repairable system with failure interaction. The International Journal of Advanced Manufacturing Technology. 2012 Jul 1;61(1-4):295-302.

[49] Wang W. An inspection model for a process with two types of inspections and repairs. Reliability Engineering & System Safety. 2009 Feb 28;94(2):526-33.

[50] Wang W, Zhao F, Peng R. A preventive maintenance model with a two-level inspection policy based on a three-stage failure process. Reliability Engineering & System Safety. 2014 Jan 31;121(1):207-20.

[51] Mendes AA, Coit DW, Ribeiro JL. Establishment of the optimal time interval between periodic inspections for redundant systems. Reliability Engineering & System Safety. 2014 Nov 30;131(1):148-65.

[52] Lin ZL, Huang YS, Fang CC. Non-periodic preventive maintenance with reliability thresholds for complex repairable systems. Reliability Engineering & System Safety. 2015 Apr 30;136(1):145-56.

[53] Lapa CM, Pereira CM, e Melo PF. Surveillance test policy optimization through genetic algorithms using non-periodic intervention frequencies and considering seasonal constraints. Reliability Engineering & System Safety. 2003 Jul 31;81(1):103-9.

[54] Castanier B, Grall A, Bérenguer C. A condition-based maintenance policy with non-periodic inspections for a two-unit series system. Reliability Engineering & System Safety. 2005 Jan 31;87(1):109-20.

[55] Wang W, Christer AH. Solution algorithms for a nonhomogeneous multi-component inspection model. Computers & Operations Research. 2003 Jan 31;30(1):19-34.

[56] Golmakani HR, Moakedi H. Optimal non-periodic inspection scheme for a multi-component repairable system using A∗ search algorithm. Computers & Industrial Engineering. 2012 Dec 31;63(4):1038-47.

[57] Lam JY, Banjevic D. A myopic policy for optimal inspection scheduling for condition based maintenance. Reliability Engineering & System Safety. 2015 Dec 31;144(1):1-1.

[58] Barker CT, Newby MJ. Optimal non-periodic inspection for a multivariate degradation model. Reliability Engineering & System Safety. 2009 Jan 31;94(1):33-43.

[59] Zhao X, Fouladirad M, Berenguer C, Bordes L. Nonperiodic Inspection/Replacement Policy for Monotone Deteriorating System with Covariates. In Fault Detection, Supervision and Safety of Technical Processes 2009 Jun 30:1617-22.

[60] Ben-Dov Y. Optimal reliability design of k-out-of-n systems subject to two kinds of failure. Journal of the Operational Research Society. 1980 Aug 1:192(1):743-8.

[61] Tian Z, Zuo MJ, Yam RC. Multi-state k-out-of-n systems and their performance evaluation. IIE Transactions. 2008 Nov 7;41(1):32-44.

[62] Chen Z. Component reliability analysis of k-out-of-n systems with censored data. Journal of statistical planning and inference. 2003 Sep 1;116(1):305-15.

[63] Coit DW, Chatwattanasiri N, Wattanapongsakorn N, Konak A. Dynamic k-out-of-n system reliability with component partnership. Reliability Engineering & System Safety. 2015 Jun 30;138(1):82-92.

[64] Eryilmaz S. Capacity loss and residual capacity in weighted k-out-of-n: G systems. Reliability Engineering & System Safety. 2015 Apr 30;136(1):140-4.

[65] Faghih-Roohi S, Xie M, Ng KM, Yam RC. Dynamic availability assessment and optimal component design of multi-state weighted k-out-of-n systems. Reliability Engineering & System Safety. 2014 Mar 31;123(1):57-62.

[66] Aboalkhair AM, Coolen FP, MacPhee IM. Nonparametric predictive inference for reliability of a k-out-of-m: G system with multiple component types. Reliability Engineering & System Safety. 2014 Nov 30;131(1):298-304.

[67] Moghaddass R, Zuo MJ, Wang W. Availability of a general k-out-of-n: G system with non-identical components considering shut-off rules using quasi-birth–death process. Reliability Engineering & System Safety. 2011 Apr 30;96(4):489-96.

[68] Bjarnason ET, Taghipour S, Banjevic D. Joint optimal inspection and inventory for a k-out-of-n system. Reliability Engineering & System Safety. 2014 Nov 30;131(2):203-15.

[69] Bjarnason ET, Taghipour S. Periodic Inspection Frequency and Inventory Policies for a k-out-of-n System. IIE Transactions. 2015 Dec 1(just-accepted):00-.

[70] Bjarnason ET, Taghipour S, Banjevic D, Jardine AK. Joint Optimization of Periodic Inspection and Inventory for a k-out-of-n System. In IIE Annual Conference. Proceedings 2013 Jan 1: 3198-207. Institute of Industrial Engineers-Publisher.

[71] Babishin V, Taghipour S. Joint optimal maintenance and inspection for a k-out-of-n system. The International Journal of Advanced Manufacturing Technology. 2016 Mar 10:1-11.

[72] Bjarnason ET, Taghipour S. Optimizing simultaneously inspection interval and inventory levels (s, S) for a k-out-of-n system. In Reliability and Maintainability Symposium (RAMS), 2014 Annual. 2014 Jan 27: 1-6. IEEE.

[73] Das Chagas Moura M, Lins ID, Droguett EL, Soares RF, Pascual R. A multi-objective genetic algorithm for determining efficient risk-based inspection programs. Reliability Engineering & System Safety. 2015 Jan 31;133(1):253-65.

[74] Gen M, Cheng R. Genetic algorithms and engineering optimization. John Wiley & Sons; 2000.

[75] Smith AE. A Review of “Genetic Algorithms and Engineering Optimization “Mitsuo Gen and Runwei Cheng Wiley, 2000. IIE Transactions. 2001 Jun 1;33(6):531-2.

[76] Tavakkoli-Moghaddam R, Safari J, Sassani F. Reliability optimization of series-parallel systems with a choice of redundancy strategies using a genetic algorithm. Reliability Engineering & System Safety. 2008 Apr 30;93(4):550-6.

Glossary

NHPP	Non-Homogeneous Poisson Process
SF	Failure of Soft-Type Component
HF	Failure of Hard-Type Component
CF	Failure of Component (k-out-of-m)
SF	System Failure (k-out-of-m)
OP	Opportunistic Inspection
PI	Potential Scheduled Inspection
$\boldsymbol{C_I}$	Inspection Cost
$\boldsymbol{C_j^M}$	Cost of Minimal Repairs
$\boldsymbol{C_j^R}$	Cost of Replacement
$\boldsymbol{C_j^D}$	Downtime Penalty Cost of a Failed Component
$\boldsymbol{C^{SD}}$	Penalty Cost for System Failure
$\boldsymbol{M_T^j}$	Expected Number of Minimal Repair of Component j over Cycle $[0, T]$
$\boldsymbol{R_T^j}$	Expected Number of Replacement of Component j over Cycle $[0, T]$
$\boldsymbol{D_T^j}$	Expected Down Time of Component j over Cycle $[0, T]$
$\boldsymbol{E\left[C_{S,\vec{x}}^T\right]}$	Total Expected Cost Incurred over Cycle [0, T] for Scheme $\vec{x}$
$\boldsymbol{E\left[C_{S,\vec{x}^*}^T\right]}$	Total Expected Cost Incurred over Cycle [0, T] for Optimal Scheme
$\boldsymbol{t_j}$	Age of Component (Soft-Type, k-out-of-m)
$\boldsymbol{\vec{\theta}}$	Age Vector of Hard-Type Component

$\boldsymbol{T}$	System's Life-Cycle
$\boldsymbol{\tau}$	Minimum Possible Unit Time
$\boldsymbol{m_1}$	Maximum Number of Hard-Type Components in the System
$\boldsymbol{m_2}$	Maximum Number of Soft-Type Components in the System
$\boldsymbol{m}$	Maximum Number of Components in the System (k-out-of-m)
$\boldsymbol{k}$	Minimum Number of Components in the System Must be Up (k-out-of-m)
$\boldsymbol{n}$	Maximum Number of Potential Scheduled Inspection in Scheme
$\boldsymbol{x_n}$	Value of Potential Scheduled Inspection (Binary 0 or 1)
$\boldsymbol{x_k^+}$	Number of Scheduled Inspection in Scheme at any Time
$\boldsymbol{x_{n+1}^+}$	Number of Scheduled Inspection at End of Life-Cycle
$\boldsymbol{y_{x_k^+}}$	Interval between Two Consecutive Scheduled Inspections
$\boldsymbol{y_{x_{n+1}^+}}$	Interval of End of the Cycle Inspection
$\boldsymbol{\vec{x}}$	Scheme, Binary Vector of Potential Scheduled Inspection
$\boldsymbol{\overrightarrow{x^*}}$	Optimal Scheme
$\boldsymbol{a_j, b_j}$	Component Parameters
$\boldsymbol{\beta_j, \eta_j}$	Parameters of Power Law Intensity Function
$\boldsymbol{r(x)}$	Probability of Minimal Repair (Soft-Type Components)
$\boldsymbol{\bar{r}(x)}$	Probability of Replacement (Soft-Type Components)
$\boldsymbol{r_j^z}$	Probability of Minimal Repair (Hard-Type Components)
$\boldsymbol{\bar{r}_j^z}$	Probability of Replacement (Hard-Type Components)
$\boldsymbol{I^X}$	Possible Events in One Inspection Interval
$\boldsymbol{\lambda_j(x)}$	Intensity Function of Soft-Type Components

$\boldsymbol{\lambda_j(z|\theta_j)}$	Intensity Function of Hard-Type Components
$\boldsymbol{\lambda(t)}$	Intensity Function of the NHPP (k-out-of-m)
$\boldsymbol{\lambda_H(z|\theta)}$	Intensity Function of Hard-Type Subsystem
$\boldsymbol{f^x(x|t)}$	Probability Density Failure Function of the Soft-Type Component
$\boldsymbol{f^z(z|\theta)}$	Probability Density Failure Function of the Hard-Type Subsystem
$\boldsymbol{R^x(x|t)}$	Reliability Function of the Soft-Type Component
$\boldsymbol{R^z(z|\theta)}$	Reliability Function of the Hard-Type Subsystem
$\boldsymbol{q^j(z)}$	Probability of the Failure of the Hard-Type Component