ABSTRACT

In this paper, a multi-objective preventive maintenance scheduling model for repairable multi-component systems developed by (Moghaddam and Usher, 2011) is adopted to evaluate computational performance of five multi-objective genetic algorithms. After computational analyses, the algorithms are ranked based on different performance metrics in order to identify the most efficient algorithm(s) to tackle similar problems encountered in research and industrial applications. It is found that the NSGA, MOGA, and NSGA-II outperform VEGA and NPGA in obtaining more and high-quality non-dominated solutions along with maintaining high level of diversity among generated solutions.

Keywords: Preventive Maintenance Scheduling, Multi-Objective Optimization, Multi-Objective Genetic Algorithms, Comparison of Algorithms

INTRODUCTION

Reliability engineering attempts to characterize, measure, and analyze the failure and repair behavior of systems in order to improve upon their operational use by increasing their design life, eliminating or reducing the likelihood of failures and safety risks, and reducing downtime thereby increasing available operating time. In this context, the maintenance is defined as those activities performed at certain intervals over a planning horizon to extend useful life of a system and to keep the system in a good condition to be productive and responsive. Preventive maintenance is a broad area of operations that encompasses a set of activities in order to improving the overall reliability and availability of systems (Elsayed, 2012). All types of complex and multi-component systems, ranging from manufacturing systems to computer network systems, have prescribed maintenance schedules set forth by the manufacturer aimed at reducing the risk of unexpected system failure during its useful life. Preventive maintenance activities generally consist of inspection, cleaning, lubrication, adjustment, alignment, and/or even replacement of components in a system subject to wear-out or degradation. These activities reduce the “effective age” of the system and thus the rate of occurrence of failure. Regardless of the specific type of system, preventive maintenance activities can be categorized in one of two ways, component maintenance or component replacement. Note that these activities change the aging characteristics of the subsystems, and if done correctly, ultimately decrease the rate of occurrence of unexpected failure of the system. In addition, preventive maintenance strategies involve a basic trade-off between the minimization of the total costs of maintenance and replacement activities and maximization of the overall reliability of the system. Designers of preventive maintenance policies must balance these individual costs along with the cost of system failure in order to minimize the overall cost of system operation.
Most real-life problems are multiple-objective problems in which objectives under consideration conflict with each other. Multi-objective models are mathematical formulations to model many complex engineering optimization problems. The population search structure of genetic algorithms (GAs) makes them suitable search procedures to find the Pareto-optimal solutions of a multi-objective problem in a single simulations run (Deb, 2001). This feature provides an opportunity for decision makers who may choose a solution, from a set of non-dominated solutions. Earlier multi-objective genetic algorithms (MOGAs) demonstrated their performance by comparing the generated non-dominated solutions with the true Pareto-optimal front in the objective functions space. But for many complex problems there is no way to perform such a test since the true Pareto-optimal front is unknown partially or completely. However, with the existence of different types of MOGAs, it is possible to evaluate their computational performance on different test problems. A MOGA will be an effective algorithm, if it can successfully perform three following conflicting features (Zitzler et al., 2000):

1. The obtained non-dominated solutions should be close enough to the true Pareto front. Ideally, the solutions should be a subset of the Pareto-optimal set.
2. The obtained non-dominated solutions should be uniformly distributed and diverse over of the Pareto front in order to provide the decision-maker a true insight of trade-offs.
3. The obtained non-dominated solutions should capture the whole spectrum of the Pareto front. This requires investigating non-dominated solutions at the extreme ends of the objective functions space.

This research adopts a multi-objective preventive maintenance and replacement scheduling model developed by (Moghaddam and Usher, 2011) and evaluates the best solution approach from available multi-objective genetic algorithms to solve the mathematical model. In order to find the non-dominated solutions, five MOGAs are employed and their computational effectiveness is evaluated and compared using different performance metrics. The main contribution of this paper resides in a performance comparison among these five algorithms. These performance metrics provide the critical benchmarks to pick the best solution method for a developed model.

The paper is organized as follows: Section 2 reviews the related work in the area of multi-objective maintenance optimization as well as MOGAs evaluations and comparison studies. Section 3 describes the multi-objective scheduling model of preventive maintenance and replacement problem. In Section 4, implementation aspects and computational results to solve the proposed scheduling model are demonstrated. In addition, an extensive evaluation and comparison to determine the most efficient algorithm are provided in Section 4. Finally, Section 5 concludes the paper and gives venues for future research and extension.

**LITERATURE REVIEW**

**Multi-Objective Preventive Maintenance Optimization**

(Kralj and Petrovic, 1995) presented a novel approach in preventive maintenance scheduling of thermal generating systems. The authors developed a large-scale multi-objective combinatorial optimization model with three objective functions and a set of constraints. They considered minimization of total fuel costs, maximization of reliability, and minimization of technological
concerns as the objective functions. They developed a multi-objective branch-and-bound algorithm and applied their methodology to a real system of 8 power plants with 21 thermal units with 11 maintenance classes over 31-week planning horizon. (Chareonsuk et al., 1997) developed a multi-criteria approach to find optimal preventive maintenance intervals of workstation in a paper factory production line with total expected costs and reliability as the objective functions. The authors then utilized a Preference Ranking Organization Method for Enrichment Evaluations (PROMETHEE) as the solution approach. (Leng et al., 2006) presented an integrated preventive maintenance scheduling and production planning as a multi-objective optimization model for a single machine. They used a chaotic particle swarm optimization algorithm to solve the model and showed its application and effectiveness via numerical examples. (Verma and Ramesh, 2007) integrated systems and subsystems of a large engineering plant into higher modular assemblies and applied a multi-objective preventive maintenance scheduling approach. They modeled this problem as a constrained nonlinear multi-objective mathematical program with reliability, cost, and non-concurrence of maintenance periods and maintenance start time into the objective functions and use a genetic algorithm to optimize the model. (Quan et al., 2007) developed a novel multi-objective genetic algorithm in order to optimize preventive maintenance scheduling problems. They defined the problem as a multi-objective optimization model by considering minimization of workforce idle time and the minimization of maintenance time and mention that there is an existing trade-off between the objective functions. As the solution procedure, they employed utility theory instead of dominance-based Pareto search to determine the non-dominated solutions. (Yulan et al., 2008) considered five objectives as minimizing maintenance cost, makespan, total weighted completion time of jobs, total weighted tardiness, and maximizing machine availability in a multi-objective integrated production and maintenance planning problem and solved the mathematical model by MOGA.

(Berrichi et al., 2009) developed a multi-objective integrated production and maintenance scheduling model to determine the Pareto-optimal front of the assignment of production tasks to machines along with preventive maintenance activities in a production system. The authors solved the model by Weighted Sum Genetic Algorithm (WSGA) and Elitist Non-dominated Sorting Genetic Algorithm (NSGA-II) and compared the computational performance of each algorithm with respect to closeness, diversity, and number of non-dominated solutions generated by each algorithm. (Berrichi et al., 2010) later proposed a heuristic algorithm which is based on multi-objective ant colony optimization (MOACO) and compared it with two multi-objective genetic algorithms, Improved the Strength Pareto Evolutionary Algorithm (SPEA2) and NSGA-II. (Moradi et al., 2011) presented a multi-objective optimization model integrating flexible job shop problem with preventive maintenance scheduling by considering the minimization of the makespan for the production part and the minimization of the system unavailability for the maintenance part. In their study, the proposed model is solved by four algorithms and it is reported that composite dispatching rule (CDR) integrated with NSGA-II and Non-Dominated Ranked Genetic Algorithm (NRGA) outperform the standard versions of NSGA-II and NRGA respectively with respect to three performance metrics. (Wang and Pham, 2011) recently presented a new multi-objective maintenance optimization model embedded within the imperfect preventive maintenance for one single-unit system. The authors employed elitist non-dominated sorting genetic algorithm (NSGA-II) to simultaneously maximize the system asymptotic availability and to minimize the system total costs. (Certa et al., 2011) developed a multi-
objective maintenance optimization model to determine the Pareto-optimal front while minimizing the total maintenance and downtime costs and also minimizing the total maintenance time with a constraint on the required reliability level.

**Comparison of Multi-Objective Genetic Algorithms**

Many studies have been conducted with the aim of comparing different types of multi-objective genetic algorithms. (Zitzler et al., 2000) evaluated and compared eight multi-objective meta-heuristic algorithms on the six experimental problems. According to their computational study, all multi-objective meta-heuristic algorithms have a better performance than the random search method while an elitist algorithm, strength Pareto evolutionary algorithm (SPEA), shows superiority over all algorithms on all tested problems. The researchers also reported the observed difficulties due to non-convexity, discreteness, and multi-modality of the objective functions in solving the problems and obtaining converge and diverse set of solutions. Finally they concluded that elite-preserving mechanism in SPEA is a critical feature for converging to the Pareto-optimal front. (Van Veldhuizen, 1999) conducted a research to compare multi-objective messy genetic algorithm (MOMGA) with MOGA, NSGA, and NPGA on seven test problems by comparing performance measure metrics of convergence and diversity. The author reports that MOMGA outperforms to MOGA, NSGA, NPGA since it is the only one armed with elite-preserving feature. The research also reports the NSGA is dominated in performance by all other algorithms. Another comparative study has been done by (Knowles and Corne, 2000) in which the authors measured the computational effectiveness of 13 algorithms on six experimental problems. They reported the NSGA algorithm with an elite-preserving process outperforms all other algorithms. (Deb et al., 2000) compared three elite-preserving MOEAs on nine problems and made a conclusion on superiority of NSGA-II over other algorithms on seven problems based on the generational distance values as a convergence metric. Their developed NSGA-II also outperforms to all algorithms in terms of diversity of the non-dominated solutions measured by spread metric values. Several other comparisons can be found in (Zitzler and Thiele, 1998), (Zydalis et al., 2001), and (Van Veldhuizen and Lamont, 2000).

**MULTI-OBJECTIVE MAINTENANCE SCHEDULING MODEL**

**Notation**

Parameters
- \( N \) : Number of subsystems
- \( L \) : Length of the planning horizon
- \( T \) : Number of maintenance intervals
- \( \lambda_i \) : Scale parameter (Characteristic life) of subsystem \( i \)
- \( \beta_i \) : Shape parameter of subsystem \( i \)
- \( \alpha_i \) : Improvement factor of subsystem \( i \)
- \( F_i \) : Unexpected failure cost of subsystem \( i \)
- \( M_i \) : Maintenance (repair) cost of subsystem \( i \)
- \( R_i \) : Replacement cost of subsystem \( i \)
- \( D \) : Downtime cost of the system
Decision Variables

$X_{i,t}$: Effective age of subsystem $i$ at the start of period $t$

$X'_{i,t}$: Effective age of subsystem $i$ at the end of period $t$

$m_{i,t} = \begin{cases} 
1 & \text{if subsystem $i$ at period $t$ is maintained (repaired)} \\
0 & \text{otherwise} 
\end{cases}$

$r_{i,t} = \begin{cases} 
1 & \text{if subsystem $i$ at period $t$ is replaced} \\
0 & \text{otherwise} 
\end{cases}$

Model Description

Suppose there is a new repairable and maintainable series system of $N$ subsystems. It is important to note that other system configurations (parallel, series-parallel, parallel-series, k-out-of-n, complex, etc) can be modeled just by modifying and adapting the reliability function to reflect the specific structure of the system. It is also assumed that each subsystem is subject to deterioration and has an increasing rate of occurrence of failure. Because of maintainability of the system under study, subsystem failure corresponds to the Non-Homogeneous Poisson Process (NHPP) identified by $\lambda_i$ and $\beta_i$ as the scale (characteristic life) and the shape parameters of subsystem $i$ respectively.

It is desirable to find a schedule of future maintenance and replacement actions for each subsystem over the planning horizon $[0, L]$. The interval $[0, L]$ is segmented into $T$ discrete intervals, each of length $L/T$. At the end of period $t$, the system is either, maintained, replaced, or no action is performed. In most systems maintenance or replacement activities in period $t$ reduce the “effective age” of the subsystems and subsequently “failure rate” of system. This kind of maintenance activities are known as minimal repairs in the literature. For simplicity it can be considered that these activities are instantaneous by which the time required to replace or maintain is negligible relative to the size of the interval, and thus can be assumed to be zero. However, a cost associated with any repair or maintenance action is considered. To account for the instantaneous changes in system age and system failure rate, first the initial age for each subsystem is also set to zero. Then let $X_{i,t}$ denote the effective age of subsystem $i$ at the start of period $t$, and $X'_{i,t}$ denotes the age of subsystem $i$ at the end of period $t$.

- Maintenance Actions

Consider the case where subsystem $i$ is maintained at the end of the period $t$. The maintenance action effectively reduces the age of subsystem $i$ for the start of the next period. The term $\alpha$ is an “improvement factor” which allows for a variable effect of maintenance on the aging of a system. When $\alpha = 0$, the effect of maintenance is to return the system to a state of “good-as-new” and when $\alpha = 1$, maintenance has no effect, and the system remains in a state of “bad-as-old”. Note that the maintenance action at the end of period $t$ results in an instantaneous drop in the rate of occurrence of failure of subsystem $i$ and also a maintenance cost $M_i$ is incurred at the end of that period.
• Replacement Actions
If subsystem $i$ is replaced at the end of period $t$ with a new identical one, then it is returned to a state of “good-as-new”. In addition, the system is charged with a replacement cost equals to the initial purchase price of the subsystem $i$, denoted as $R_i$.

• Do Nothing
If no action is planned to be taken in period $t$, the rate of occurrence of failure of subsystem $i$ will be not reduced and the subsystem continues its aging with an increasing rate of occurrence of failure. When a future schedule of maintenance operations for a system is planned, the inevitable costs caused by unexpected subsystem failures must be taken into account. In order to take into account these unplanned costs, the calculation of expected number of failures in each period for each subsystem in the system is proposed. The cost of each failure is defined as $F_i$ (in units of $/failure event) and $F_{i,t}$ the cost of failures attributable to subsystem $i$ in period $t$ can be found as follows. For more information on development of the failure cost see (Moghaddam and Usher, 2011).

• System Downtime Cost
There should be some overall system penalty cost when an action is performed on any subsystem. It would seem that there should be some logical advantage to combining maintenance and replacement actions at the same time. For instance, while the system is shut down to replace one subsystem it may make sense to perform maintenance or replacement for some other subsystems, even if they are not at their individual optimum point where maintenance or replacement would ordinarily be performed. Under this scenario, the optimal time to perform maintenance or replacement actions on individual subsystem is completely dependent upon the decision made for other subsystems. As such, it is proposed that a fixed cost of “downtime”, $D$, is charged in period $t$ if any subsystem (one or more) is maintained or replaced in that period.

The adopted multi-objective nonlinear mixed-integer optimization model with the total operational costs and system reliability objective functions (Moghaddam and Usher, 2011) is presented as:
Min $\text{Total Operational Costs} = \sum_{i=1}^{N} \sum_{t=1}^{T} \left[ F_i \cdot \lambda_i \left( (X'_{i,t})^\beta - (X_{i,t})^\beta \right) + M_i \cdot m_{i,t} + R_i \cdot r_{i,t} \right]$

$$+ \sum_{t=1}^{T} \left[ D \left( 1 - \prod_{i=1}^{N} (1 - (m_{i,t} + r_{i,t})) \right) \right]$$

Max $\text{System Reliability} = \prod_{i=1}^{N} \prod_{t=1}^{T} \exp \left[ -\lambda_i \left( (X'_{i,t})^\beta - (X_{i,t})^\beta \right) \right]$

subject to:

$$X_{i,t} = 0 \quad i = 1, \ldots, N$$

$$X_{i,t} = (1 - m_{i,t-1})(1 - r_{i,t-1})X'_{i,t-1} + m_{i,t-1}(\alpha_i \cdot X'_{i,t-1}) \quad i = 1, \ldots, N; t = 2, \ldots, T$$

$$X'_{i,t} = X_{i,t} + (L / T) \quad i = 1, \ldots, N; t = 1, \ldots, T$$

$$m_{i,t} + r_{i,t} \leq 1 \quad i = 1, \ldots, N; t = 1, \ldots, T$$

$$m_{i,t}, r_{i,t} = 0 \text{ or } 1 \quad i = 1, \ldots, N; t = 1, \ldots, T$$

$$X_{i,t}, X'_{i,t} \geq 0 \quad i = 1, \ldots, N; t = 1, \ldots, T$$

(1)

In the above optimization model, $m_{i,t}$ and $r_{i,t}$ are binary variables of maintenance and replacement actions for subsystem $i$ in period $t$. The first set of constraints indicates that the initial age for each subsystem is equal to zero. The second set mentions that if a subsystem replacement occurs in the previous period then $r_{i,t-1} = 1$, $m_{i,t-1} = 0$, so $X_{i,t} = 0$. If a subsystem was maintained in the previous period then $r_{i,t-1} = 0$, $m_{i,t-1} = 1$, so $X_{i,t} = (\alpha_i \cdot X'_{i,t-1})$ and finally if no action was taken, $r_{i,t-1} = 0$, $m_{i,t-1} = 0$, and $X_{i,t} = X'_{i,t-1}$.

**COMPUTATIONAL RESULTS**

The traditional approach to solve multi-objective optimization problems is based on preference-based approach with predetermined parameters. All the classic methods employ a point-by-point deterministic optimization approach by finding single optimal solutions. Since multi-objective optimization problems have equally important Pareto-optimal solutions (also known as “trade-off optimal solutions”), the ideal approach would be finding multiple Pareto-optimal solutions in a single run and let the decision maker choose the desired solution based on other higher-level information. Genetic Algorithms (GAs) mimic nature’s evolutionary principles to derive an intelligent search towards an optimal solution using a population of solutions in each generation (i.e. iteration). Since a population of solutions is tracked and evaluated in each generation, the outcome of such an algorithm will be a population of solutions. This feature of genetic algorithms to generate, track, and evaluate multiple solutions in one single simulation run enables this family of algorithms to find multiple optimal solutions and makes the algorithms unique solution methodologies in solving multi-objective optimization problems (Zitzler and Thiele, 1998). In addition, genetic algorithms have been proved to be able to search simultaneously different regions of a solution space which makes it possible to find a diverse set of solutions for complex problems with nonlinear and non-convex functions, discrete variables,
and multi-modal solution spaces characteristics (Konak et al., 2006). Since any genetic algorithm needs one fitness function to evaluate a candidate solution, the main task in modifying a single objective genetic algorithm to tackle multi-objective optimization problems is to find a single metric from a number of objective functions to identify non-dominated solutions as close as possible to true Pareto-optimal front (Deb, 2001). Furthermore, the obtained non-dominated solutions must be as diverse as possible to be uniformly and sparsely distributed in the Pareto-optimal region showing trade-off among different objective functions (Deb, 2001). In this research we consider the following algorithms to be evaluated for the adopted model (1):

- Vector Evaluated Genetic Algorithm (VEGA)
- Multi-Objective Genetic Algorithm (MOGA)
- Non-dominated Sorting Genetic Algorithm (NSGA)
- Niched-Pareto Genetic Algorithm (NPGA)
- Elitist Non-dominated Sorting Genetic Algorithm (NSGA-II)

In order to illustrate the optimization model numerically and evaluate the algorithms, a representative data set shown in Table 1 along with $1000 as the downtime cost and 12 months as the planning horizon is used. Under this setting, the multi-objective problem has 470 variables, 240 of which are binary and 360 functional constraints, 110 of which are nonlinear along with two nonlinear objective functions, total operational cost and reliability of the system.

### Table 1: Parameters of the optimization model

<table>
<thead>
<tr>
<th>Subsystem</th>
<th>Scale parameter</th>
<th>Shape parameter</th>
<th>Improvement factor</th>
<th>Failure cost ($)</th>
<th>Maintenance cost ($)</th>
<th>Replacement cost ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0022</td>
<td>2.20</td>
<td>0.62</td>
<td>250</td>
<td>35</td>
<td>200</td>
</tr>
<tr>
<td>2</td>
<td>0.0035</td>
<td>2.00</td>
<td>0.58</td>
<td>240</td>
<td>32</td>
<td>210</td>
</tr>
<tr>
<td>3</td>
<td>0.0038</td>
<td>2.05</td>
<td>0.55</td>
<td>270</td>
<td>65</td>
<td>245</td>
</tr>
<tr>
<td>4</td>
<td>0.0034</td>
<td>1.90</td>
<td>0.50</td>
<td>210</td>
<td>42</td>
<td>180</td>
</tr>
<tr>
<td>5</td>
<td>0.0032</td>
<td>1.75</td>
<td>0.48</td>
<td>220</td>
<td>50</td>
<td>205</td>
</tr>
<tr>
<td>6</td>
<td>0.0028</td>
<td>2.10</td>
<td>0.65</td>
<td>280</td>
<td>38</td>
<td>235</td>
</tr>
<tr>
<td>7</td>
<td>0.0015</td>
<td>2.25</td>
<td>0.75</td>
<td>200</td>
<td>45</td>
<td>175</td>
</tr>
<tr>
<td>8</td>
<td>0.0012</td>
<td>1.80</td>
<td>0.68</td>
<td>225</td>
<td>30</td>
<td>215</td>
</tr>
<tr>
<td>9</td>
<td>0.0025</td>
<td>1.85</td>
<td>0.52</td>
<td>215</td>
<td>48</td>
<td>210</td>
</tr>
<tr>
<td>10</td>
<td>0.0020</td>
<td>2.15</td>
<td>0.67</td>
<td>255</td>
<td>55</td>
<td>250</td>
</tr>
</tbody>
</table>

### Implementation of the Genetic Algorithms

MATLAB programming environment is utilized to construct the model and the multi-objective metaheuristic algorithms to be all executed on a laptop computer (Intel/Core 2, 1.7 GHz and 2 GB RAM). As known, the quality of solutions is largely affected by the parameter values of a heuristic or metaheuristic algorithm thus in order to find suitable values for the parameters of the genetic algorithms, a set of candidate values for each parameter is selected and tested in intensive computational experiments by varying one parameter at each step using a trial and error approach. The selected parameter setting for all multi-objective genetic algorithms is presented in Table 2.
The first step in any genetic algorithm implementation is to develop an encoding of the solution to be represented as unique chromosome. In order to represent the solution of preventive maintenance and replacement scheduling problem with do nothing, maintenance and replacement actions; an array with length of $N \times T$ for $N$ subsystems and $T$ periods is defined. Each cell in this array contains 0, 1 or 2 corresponds to three different actions.

Crossover Procedures
The crossover procedures create a new solution as an offspring of pair of selected solutions (parent solutions). The offspring should inherit some useful properties of both parents in order to facilitate its propagation throughout the population. In this research, several common crossover procedures (i.e. one-point crossover, two-point crossover, inverse crossover, etc) were employed and tested in terms of generated solutions, but it is found that they produce poor solutions resulting to premature convergence of solutions over the generations of the genetic algorithms. Therefore, based on the special structure of the problem, two new crossover procedures to overcome this ineffectiveness of the tested crossovers are designed as follow:

Two-Point Inverse Crossover: This type of crossover first generates two random numbers between 1 and $N \times T$, then makes an offspring from selected parents in which all elements outside the position of those random numbers are copied from the first parent but in an inverse order and inner elements are copied from the second parents. If the chosen parents are identical, this type of crossover makes a different offspring, which is not the same to its parents.

$NT$-Point Crossover: In this type of crossover, the even genes are copied from the first parent and odd genes are copied from the second parent.

It is designed that if two selected solutions are equal to each other, then the algorithm uses Two-Point Inverse Crossover, and if selected solutions are not same, the algorithm uses $NT$-Point Crossover to produce new solutions.

Mutation Procedure
The mutation procedure is applied to the generated offspring solution. It makes changes into the solution encoding string by modifying some of the string elements. Again, based on the special structure of the optimization model in which if even one maintenance or replacement action takes place in a period, the whole system encounters large downtime cost, special type of mutation procedure is designed. In this type of mutation, first a random number is generated between 1 and $N \times T$ rather than between 1 and $T$ to have more randomness and variability over the entire solution. Then the corresponding gene is changed to 1 or 2 if it equals to 0, and it is changed to 0 if it equals to 1 or 2 and the same procedure in the same period for other
subsystems is performed. This generally produces scheduled maintenance and replacement activities in same periods across many subsystems.

Table 3 depicts a typical Pareto-optimal schedule obtained by NSGA-II with the total operational cost of $13298.49 and the overall systems reliability of 0.4488. Note that in presented Pareto-optimal schedule all maintenance and replacement actions tend to occur in the same period, which reflects the effect of the downtime cost.

<table>
<thead>
<tr>
<th>Subsystem/Month</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>M</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>R</td>
<td>M</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>R</td>
<td>R</td>
<td>M</td>
<td>R</td>
<td>R</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>R</td>
<td>M</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>R</td>
<td>M</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>R</td>
<td>R</td>
<td>M</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>R</td>
<td>R</td>
<td>M</td>
<td>R</td>
<td>R</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>R</td>
<td>M</td>
<td>M</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>R</td>
<td>M</td>
<td>R</td>
<td>R</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Computational Performance of the MOEAs

In this study, 10 independent runs of each algorithm are performed to solve the model. The computational efficiency of the algorithms in terms of average number of obtained non-dominated solutions and average CPU time is examined as shown in Table 4 and Figures 1-2. As can be observed, VEGA is the fastest algorithm with 2 minutes and 30 seconds of average running time among the tested algorithms however comparing to other algorithms it generates fewer non-dominated solutions at the end. It is useful to reiterate that VEGA is the only algorithm in our study without any mechanism of maintaining diversity among non-dominated solutions. On the other side of the spectrum, NSGA-II takes 6 minutes and 41 seconds to generate 128 non-dominated solutions. Again remember that NSGA-II is the only algorithm in our study with elite-preserving mechanism to maintain and pass good non-dominated solutions from one generation to another which eventually results to the longest computational time. In large-scale optimization problems with more than two objective functions and with thousands of decision variables and functional constraints the computational time will definitely be an issue that needs to be considered before choosing the most suitable algorithm to tackle the problem. As in most situations, the trade-off between desired number of non-dominated solutions and computational time must be taken into account for every problem.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Average number of non-dominated solutions</th>
<th>Average computational time (second)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VEGA</td>
<td>32</td>
<td>150</td>
</tr>
<tr>
<td>MOGA</td>
<td>78</td>
<td>205</td>
</tr>
<tr>
<td>NSGA</td>
<td>120</td>
<td>387</td>
</tr>
<tr>
<td>NPGA</td>
<td>70</td>
<td>354</td>
</tr>
<tr>
<td>NSGA-II</td>
<td>128</td>
<td>401</td>
</tr>
</tbody>
</table>
Figures 3-8 illustrate the obtained non-dominated solutions in objective functions space by employing the multi-objective genetic algorithms. The most extreme non-dominated solutions obtained by the algorithms are identified to be (Cost = 37574.8260, Reliability = 0.7311037) generated by VEGA and (Cost = 961.8265, Reliability = 0.0189880) generated by all other algorithms. By examining the Figures 3-8 it is subjectively clear that non-dominated solutions generated by NSGA and NSGA-II might be more likely to be closer to the true Pareto-optimal front and also they cover more area in the objective functions space to reflect the existing trade-off between the total operational cost and the overall reliability of the system. As stated before the first task in solving multi-objective optimization problems is to identify non-dominated solutions as close as possible to the un-known Pareto-optimal front. Another necessary feature to be carried is that the obtained non-dominated solutions must be uniformly distributed in the Pareto-optimal region reflecting the existing trade-off among different objective functions. Finally in order to capture whole spectrum of the Pareto front, extreme solutions at the objective functions space should be identified. In order to quantitatively compare capabilities of the algorithms in solving the model, four types of performance metrics to determine the relative domination of obtained non-dominated solutions by each algorithm and also to measure the diversity of those solutions in the objective functions space are considered.
Figure 3: Non-dominated solutions obtained by VEGA

Figure 4: Non-dominated solutions obtained by MOGA

Figure 5: Non-dominated solutions obtained by NSGA
Figure 6: Non-dominated solutions obtained by NPGA

Figure 7: Non-dominated solutions obtained by NSGA-II

Figure 8: All non-dominated solutions obtained by MOEAs
Metric Evaluating Closeness to the True Pareto-Optimal Front

Since the true Pareto-optimal solutions $P^*$ of the multi-objective scheduling model are unknown, comparing obtained non-dominated solutions with the members of $P^*$ is impossible. Therefore the set coverage metric is used to evaluate the relative domination of non-dominated solutions obtained by each algorithm.

- **Set coverage metric**

  The set coverage metric $C(A, B)$ suggested by (Zitzler, 1999) calculates the proportion of solutions generated by algorithm $B$ which are dominated by solutions generated by algorithm $A$ as follows:

  $$C(A, B) = \frac{\left| \{ b \in B \mid \exists a \in A : a \prec b \} \right|}{|B|}$$

  (2)

  Where $a \prec b$ means that solution $b$ is dominated by solution $a$. Hence, if the metric value $C(A, B) = 1$ it reveals that all members of $B$ are dominated by $A$ and if $C(A, B) = 0$ it shows that no solutions of $B$ is dominated by $A$. Therefore the smaller value of the metric is preferred with respect to algorithm $B$. It is also necessary to clarify that domination operator is not a symmetric logical operator ($C(A, B) \neq 1 - C(B, A)$) so both $C(A, B)$ and $C(B, A)$ should be calculated. Table 5 provides the value of set coverage metric from a pair-wise comparison of the algorithms.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>VEGA</th>
<th>MOGA</th>
<th>NSGA</th>
<th>NPGA</th>
<th>NSGA-II</th>
<th>Average percentage of dominate solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>VEGA</td>
<td>-</td>
<td>0.012821</td>
<td>0.100000</td>
<td>0.114286</td>
<td>0.171875</td>
<td>0.099745</td>
</tr>
<tr>
<td>MOGA</td>
<td>0.531250</td>
<td>-</td>
<td>0.175000</td>
<td>0.500000</td>
<td>0.398438</td>
<td>0.401172</td>
</tr>
<tr>
<td>NSGA</td>
<td>0.625000</td>
<td>0.500000</td>
<td>-</td>
<td>0.871429</td>
<td>0.640625</td>
<td>0.659263</td>
</tr>
<tr>
<td>NPGA</td>
<td>0.375000</td>
<td>0.102564</td>
<td>0.008333</td>
<td>-</td>
<td>0.296875</td>
<td>0.195693</td>
</tr>
<tr>
<td>NSGA-II</td>
<td>0.375000</td>
<td>0.205128</td>
<td>0.100000</td>
<td>0.428571</td>
<td>-</td>
<td>0.277175</td>
</tr>
</tbody>
</table>

Each column in the above table shows the percentage of solutions obtained by each algorithm which are dominated by solutions of the other algorithms. Last row also demonstrates the average percentage of dominated solutions of each algorithm. By carefully reviewing the metric value it can be stated that only 9.5% of the solutions generated by NSGA are dominated by the solutions from other algorithms and NSGA performs excellent in producing high-quality of non-dominated solutions over all other tested algorithms. On the other hand about 48% of the NPGA and VEGA solutions are dominated by the solutions of other algorithms. Beside NSGA, MOGA also performs well in generating non-dominated solutions in which only 20% of its solutions are being dominated by other algorithms. Similar conclusion can be drawn be comparing the average percentage of dominate solution calculated in the last column. Therefore NSGA and MOGA satisfactorily demonstrate the capability of solving the multi-objective scheduling problem by identifying non-dominated solutions as close as possible to the un-known Pareto-optimal front. It was initially expected that NSGA-II would outperform the other algorithms because it is the only
one armed with elite-preserving mechanism to maintain and pass good non-dominated solutions from one generation to another. However NSGA-II computational performance in generating non-dominated solutions is still in an acceptable level since only, on average, 37% of its solutions are dominated by others’ solutions.

**Metrics Evaluating Diversity among Non-Dominated Solutions**

- **Spacing**
  The spacing metric developed by (Schott, 1995) measures the standard deviation of distance between any solution and all other solutions and it reveals how well the non-dominated solutions are distributed in the objective functions space by the following equation:

  \[ S = \sqrt{\frac{1}{|Q|} \sum_{i=1}^{|Q|} (d_i - \bar{d})^2} \]  

  \[ S = \sqrt{\frac{1}{|Q|} \sum_{i=1}^{|Q|} (d_i - \bar{d})^2} \]  

  Where \( d_i \) is the minimum rectilinear distance between each non-dominated solution and all other obtained non-dominated solutions in set \( Q \) in objective functions space whereas \( \bar{d} \) is the average of minimum rectilinear distances of all solutions in that space.

  \[ d_i = \min_{k \in Q, k \neq i} \sum_{m=1}^{M} \left| f_m^{(i)} - f_m^{(k)} \right| \]  

  \[ d_i = \min_{k \in Q, k \neq i} \sum_{m=1}^{M} \left| f_m^{(i)} - f_m^{(k)} \right| \]  

  \[ \bar{d} = \frac{\sum_{i=1}^{|Q|} d_i}{|Q|} \]  

  \[ \bar{d} = \frac{\sum_{i=1}^{|Q|} d_i}{|Q|} \]  

- **Maximum Spread**
  The maximum spread metric first used by (Zitzler, 1999) measures the length of the diagonal of a hypercube formed by the extreme obtained non-dominated solutions in the normalized objective functions space by the following formulation:

  \[ \overline{D} = \sqrt{\frac{1}{M} \sum_{m=1}^{M} \left( \frac{|c|}{\max_i f_m^i - \min_i f_m^i} \right)^2} \]  

  \[ \overline{D} = \sqrt{\frac{1}{M} \sum_{m=1}^{M} \left( \frac{|c|}{\max_i f_m^i - \min_i f_m^i} \right)^2} \]  

  Where, \( F_m^{\max} \) and \( F_m^{\min} \) are the maximum and minimum value of the m-th objective functions. The metric provides a measurement about how far the extreme non-dominated solutions are located. Therefore \( \overline{D} = 1 \) represents the fact that a widely spread set of non-dominated solutions is generated in the last generation so basically values close to one are preferred.

Table 6 demonstrates the spacing and maximum spread metrics values calculated based on non-dominated solutions obtained by the tested MOGAs. It is seen that maximum spread of VEGA to be found as 0.999989 and the algorithm generates a near perfect extreme non-dominated
solutions. However these non-dominated solutions are not uniformly distributed in the objective functions space as the spacing metric of VEGA is about 3.4%. On the other hand, NSGA-II generates excellent distributed solutions in the objective functions space however the algorithm cannot compete with other algorithms in generating most extreme non-dominated solutions that results to a moderate maximum spread metric value of 0.895663. In general, MOGA, NSGA, NPGA, and NSGA-II perform very well in generating spread and well distributed non-dominated solutions making these algorithms trustable in achieving the secondary task of solving the multi-objective scheduling model.

<table>
<thead>
<tr>
<th>Table 6: Evaluating diversity among non-dominated solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Algorithm</strong></td>
</tr>
<tr>
<td>VEGA</td>
</tr>
<tr>
<td>MOGA</td>
</tr>
<tr>
<td>NSGA</td>
</tr>
<tr>
<td>NPGA</td>
</tr>
<tr>
<td>NSGA-II</td>
</tr>
</tbody>
</table>

**Metric Evaluating both Closeness and Diversity among Non-Dominated Solutions**

- **Hyper volume**
  To better evaluate the quality of the obtained non-dominated solutions another metric is adopted. The hyper volume metric developed by (Van Veldhuizen, 1999) calculates an approximation of the volume of the hypercube formed by the non-dominated solutions in the objective functions space. This metric can measure both main and secondary tasks of solving a multi-objective optimization problem; closeness to the true Prato-optimal front and diversity of non-dominated solutions. Literally, a volume, $v_i$, is constructed by each non-dominated solution and a reference point $w$ as a diagonal corners of the hypercube.

$$HV = \text{volume} \left( \bigcup_{i=1}^{n} v_i \right)$$  \hspace{1cm} (7)

In our analysis, the nadir solution, $w_{nadir}$ (Cost = 37574.8260, Reliability = 0.0189880) is selected as reference point (the worst obtained value of each objective function). Because of different order of magnitude of total operational cost and system reliability, the normalized values of non-dominated solutions along with normalized nadir point (Cost = 1, Reliability = 0) are used to calculate the $HV_{normalized}$ (with maximum possible value of one) by the following equation:

$$HV_{normalized} = \left( \left( Cost_{i, normalized} - Cost_{nadir, normalized} \right) \times \left( R_{i, normalized} - R_{nadir, normalized} \right) \right) + \sum_{i=2}^{k} \left( \left( Cost_{i, normalized} - Cost_{i-1, normalized} \right) \times \left( R_{i, normalized} - R_{nadir, normalized} \right) \right)$$  \hspace{1cm} (8)
Table 7 provides the normalized hyper volumes formed by non-dominated solutions generated by the algorithms. It is observed that NSGA’s non-dominated solutions with HV = 0.675579 forms a larger hypercube than the other algorithms and it outperforms in generating diverse and well-distributed solutions close enough to the true Pareto-optimal front by this metric making NSGA satisfies both main and secondary tasks expected from an ideal MOEA.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Hyper volume (HV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VEGA</td>
<td>0.645600</td>
</tr>
<tr>
<td>MOGA</td>
<td>0.666090</td>
</tr>
<tr>
<td>NSGA</td>
<td>0.675579</td>
</tr>
<tr>
<td>NPGA</td>
<td>0.652809</td>
</tr>
<tr>
<td>NSGA-II</td>
<td>0.663407</td>
</tr>
</tbody>
</table>

Table 8 illustrates the rank of each algorithm according to the different computational performance metrics in solving the multi-objective preventive maintenance and replacement scheduling model developed in this study. It can be seen that NSGA, MOGA, and NSGA-II outperform VEGA and NPGA in most performance measures. However selection of the most suited algorithm depends on the decision maker’s preferences in a specific task such as generating large number of non-dominated solutions regardless of how long it takes to identify as many candidate solutions as possible (i.e. NSGA and NSGA-II) or just obtaining a few but extreme non-dominated solutions to provide informative insight about the existing trade-off among conflicting objective functions (i.e. VEGA).

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Number of non-dominated solutions</th>
<th>Computational time</th>
<th>Set coverage (C)</th>
<th>Spacing (S)</th>
<th>Maximum spread</th>
<th>Hyper volume (HV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VEGA</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>MOGA</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>NSGA</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>NPGA</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>NSGA-II</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>3</td>
</tr>
</tbody>
</table>

CONCLUSIONS AND FUTURE RESEARCH

This research adopted a multi-objective nonlinear mixed-integer optimization model to determine the Pareto-optimal preventive maintenance and replacement schedules in a multi-subsystem repairable system with total operational cost and overall system reliability as the objective functions. In order to tackle this NP-hard problem, five multi-objective genetic algorithms are evaluated using different performance metrics. In addition to this evaluation and based on the special structure of the problem, two new crossover procedures and a mutation mechanism are considered. After computational analyses, the algorithms are ranked based on different performance measures. It is found that the NSGA, MOGA, and NSGA-II outperform VEGA and NPGA in obtaining more and high-quality non-dominated solutions along with maintaining high level of diversity among generated solutions. However, the results in Table 8 indicate that the selection of the best algorithm(s) is itself a multi-criteria decision making
problem (MCDM) with 6 criteria and 5 alternatives. Therefore, in order to determine the best algorithm, MCDM methods such as Analytic Hierarchy Process (AHP) are recommended to be employed in practical applications.

The adopted model in this paper can be directly applied or indirectly integrated in designing maintenance plans in wide variety of applications such as productions systems, transportation fleet operations, material handling systems, and power plant management. The model and the tested MOGAs can be used to quickly generate new preventive maintenance and replacement plans in complex systems even after occurring unexpected failures. In such a situation, the original schedule should be updated and the new optimal schedule can be used over the remaining of the planning horizon. This multi-objective model can also be used in condition-based simulation models as a real-time optimization procedure to refine and update maintenance plans during the simulation run. Future work in this area is needed to investigate the effects of other constraints such as limited maintenance resources on system availability and the expected demand by considering other strategies of maintenance activities.

References


