ANALYSIS OF REPRESENTATION ALTERNATIVES FOR A MULTIPLE-OBJECTIVE FLOATING BOUND SCHEDULING PROBLEM OF A NUCLEAR POWER PLANT SAFETY SYSTEM

Daniel E. Salazar *, Sebastián S. Martorell † and Blas J. Galván *

* Institute of Intelligent Systems and Numerical Applications in Engineering (IUSIANI)
University of Las Palmas de Gran Canaria (ULPGC)
danielsalazaraponte@gmail.com / bgalvan@step.es

† Department of Chemical and Nuclear Engineering
Polytechnic University of Valencia
P.O. Box 22012, 46071, Spain
smartore@iqn.upv.es

Key words: Genetic Algorithms, Evolutionary Multiple-Objective Optimization (EMO), Multiple-Objective Evolutionary Algorithm (MOEA), Safety System design, Nuclear Power Plant.

Abstract. The simultaneous optimization of Test Intervals and Test Strategies considered in the Surveillance Requirements for Nuclear Power Plants is a multiple-objective scheduling problem which has been tackled only recently [7-8]. The formulation of this class of problems presents dependence between the current values of some variables and the allowed upper values of others, defining a domain composed by a set of disjoint subsets. In this paper we propose and study different alternatives for representing and solve the problem using multiple-objective genetic algorithms. Additionally, the concepts of $\varepsilon$-dominance and $\varepsilon$-approximate Pareto set were employed to post-process and compare the outcomes. As a result, the introduced percentage representation showed the best performance.
1 INTRODUCTION

As a part of the Technical Specifications of Nuclear Power Plants, the Surveillance Requirements establish the Test Intervals (TI) and the Test Strategy or Planning (TP) of the Safety Systems of such nuclear plants. Given a Safety System, its TI consists in a period between tests, whereas the TP consists in the schedule for testing each component of the system. For example, if one system of two components (A, B) has the TI = 120 days and the TP = (10, 70), it means that component A will be tested on the 10th day, and again on the 130th day, and on the 250th day and so on. Analogously, component B will be tested on the (70+i*120)th day, for \( i \in \mathbb{N} \).

When a contingency occurs, the Safety Systems are supposed to respond. However, normally the Safety Systems are not under constant demand, but in standby; thus, the system is expected to be available any time it is needed, in other words, there is a need to minimise the unavailability of the system. Such unavailability can be reduced in average, decreasing the TI, but it involves increasing the testing costs.

Additionally, the TP has an impact on both unavailability and costs. Notice that during the tests, the tested components are not available for responding; in consequence, inappropriate scheduling can increase the peak unavailability, leading to risky scenarios for the plant.

Considering the above exposition, it is clear that establishing both the TI and the TP is a multiple-objective scheduling problem. Nevertheless, most of the previous studies have been focused the optimization of TI or TP separately, based on single or multiple criteria \([6, 1, 2, 4]\). Only recently the simultaneous multiple-objective optimization of the TI and the TP has been accomplished \([7-8]\).

The practical importance of the above problem is evident, but additionally there are two features that justify a deeper analysis of solving alternatives. On one hand, the objective functions evaluation is time consuming, therefore a complete enumeration is unacceptable; on the other hand, there is a dependency between the TI and the TP that makes the enumerative search space a set of disjoint subspaces.

In this paper we study the particular characteristics of the problem and propose different alternatives for solving it efficiently using genetics algorithms. The remainder of the paper is organised as follows: In section 2 a complete formulation of the problem is presented; section 3 and 4 are devoted to analyse the characteristics of the search space and provide some alternatives for representing and implementing the problem with multiple-objective genetic algorithms. In section 5 and 6 some results and conclusions are offered.

2 PROBLEM FORMULATION

Figure 1 depicts a simplified High Pressure Injection System (HPIS), which is used to remove heat from the reactor in some accidents. The system consists of 3 pumps and several valves. For this system, it is necessary to establish the TI, which comprises 3 intervals \( \{T_1, T_2, T_3\} \), and the TP composed by 6 times to first test \( \{TA, TB, TC, TD, TE, TF\} \). E.g., the decision variables associated to pump PA correspond to the set \( \{T_1, TA\} \).
To simplify the problem, the following relationships were considered (for more details see [8]):

- Regarding TI:
  \[ T_2 = 3 \times T_1 \]
  \[ T_3 = 3 \times T_1 \]

- Regarding TP:
  \[ T_D = T_B \]
  \[ T_E = T_B + T_1 \]
  \[ T_F = T_B + 2 \times T_1 \]

Thus, the search space is reduced to four variables \{T_1, T_A, T_B, T_C\}, the first one concerning TI and the remainder concerning TP. The problem resultant can be formulated as:

Minimize the multiple-objective vectors function of the form:

\[ y = (U_m(x), U_{tm}(x), C(x)) \]

s.t. the objectives constraints:

\[ U_m(x) \leq 2 \times 10^{-4} \]
\[ U_{tm}(x) \leq 2 \times 10^{-2} \]
\[ C(x) \leq 5 \times 10^{4} \]

where \( x \) is the decision vector \{T_1, T_A, T_B, T_C\} and \( y \) the objectives vector, composed by:

\( U_m(x) = \text{Mean system unavailability for the decision vector, } x. \)

\( U_{tm}(x) = \text{Maximum time-dependent unavailability of the system for the decision vector } x. \)

\( C(x) = \text{Yearly cost associated for the decision vector, } x [\text{€/Yr}]. \)

For details of the calculation of the objectives see [8]. It is worth to mention that each evaluation of vector \( y \) takes approximately one second in moderns CPU.
The minimum and maximum values for the decision variables can be seen in table 1. Notice that the lower bound are fixed for all the variables, but the maximum bound is fixed for only T1, whereas for TA, TB, and TC such bound is variable, i.e. depends on T1. Both the TI and the TP are expressed in terms of hours, but in practice expressing the schedules in terms of days is enough. Besides, notice also that the maximal value for T1 is 58, because no feasible individual could be found for larger T1 [8]. Nonetheless, initially this quantity was fixed in 365.

<table>
<thead>
<tr>
<th>x</th>
<th>Minimum (days)</th>
<th>Maximum (days)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>30</td>
<td>58</td>
</tr>
<tr>
<td>TA</td>
<td>0</td>
<td>Variable (equal to current T1-1)</td>
</tr>
<tr>
<td>TB</td>
<td>0</td>
<td>Variable (equal to current T1-1)</td>
</tr>
<tr>
<td>TC</td>
<td>0</td>
<td>Variable (equal to current T1-1)</td>
</tr>
</tbody>
</table>

Table 1: Bounds of decision vector x components.

3 SEARCH SPACE ANALYSES

Given that the maximal period between consecutive tests is T1, the times for first test, i.e. \{TA, TB, TC\}, cannot be higher than T1. For this reason, once the T1 value is chosen, the search space for this particular T1, denoted by S(T1), is defined as the combinatorial space of \{TA, TB, TC\} which has size \(T1^3\). Notice that all subspaces S(T1) are disjoint regarding each others. Hence, in general, the complete search space S is a collection of disjoint sets, which has size:

\[
|S| = |S(\min T1)| + \cdots + |S(\max T1 - 1)| = \sum_{n=\min T1}^{\max T1 - 1} n^3
\]  

(1)

According to the above equation, the cardinality of S for the problem considered (T1 from 30 to 58) is 2738296, which means about 760 hours of CPU for a complete enumeration. Obviously, the employment of heuristics to approximate the solution as fast as possible should be considered. In this work we study three alternatives, the first one previously developed in [8], and the others two proposed for this work.

The first alternative is derived in a natural way from the structure of the problem. Due to the search space is composed by disjoint subspaces, it is necessary to explore each one of these subspaces to approximate the non dominated frontier of the problem. This can be accomplished by means of a double-loop configuration, which means that an external loop fixes the values of T1, and then an internal nested loop performs a multiple-objective optimization. This alternative was first studied in [8], showing the ability to approximate the Pareto frontier as expected.

The main limitation of the double-loop configuration is that, precisely because it works on disjoint subspaces, any solution found in one space is not defined in the others; in consequence the good solutions cannot be extrapolated or mixed directly between subspaces.
This last seems to contradict the foundations of metaheuristics, which try to take advantage of any good feature discovered so far during the process.

For this reason, we propose a second alternative, based on the observation that the dependency between the TI and the TP can be expressed in relative terms (percentages) instead of absolute terms, like in double-loop configuration. In fact, a representation of the type \(\{T_1, T_{A\%} = T_1, T_{B\%} = T_1, T_{C\%} = T_1\}\) unifies the search space making possible to explore the whole domain of \(T_1\) at the same time. However, this unification implies an expansion of the search space, which can make the size of the resultant space significantly bigger than the initial one, depending on the type of representation adopted. For instance, considering the case when the percentages are represented with bytes, the search space becomes of size \((58-30+1)*255^3 = 480859875\) which is 175.6 times bigger than the original space. Additionally, if \(T_1\) is larger than e.g. 300, one byte is not enough to represent all possible states of \(T_A, T_B\) and \(T_C\).

Of course, the use of bigger binary variables is possible, as well as the use of real representation for the percentages, but with the consequent augment of the search space.

Hence, the minimal unified search space that contains any point of the original one can be defined making \(T_{A\%}, T_{B\%}\) and \(T_{C\%}\) integer variables between 0 and \((\text{maximal } T_1 - 1)\) and then decoding each individual by calculating the percentages in terms of its current \(T_1\) value as

\[
T = \left[ \frac{T_{\%}}{\text{max } T_1} \right]
\]  

For example, consider two individuals \(a = \{58, 45, 57, 30\}\) and \(b = \{40, 10, 50, 57\}\) of type \(\{T_1, T_{A\%}, T_{B\%}, T_{C\%}\}\). According to the above equation for \(\text{max } T_1 = 58\), the final (decoded) value of \(a\) is \(\{58, 45, 57, 30\}\), whereas for \(b\) the decoded value is \(\{40, 6, 34, 39\}\). In this way, one can apply crossover and mutation operators over any chromosome without restrictions, overcoming the limitations imposed by the disjoint structure of the original search space. Obviously, the expansion of the search space remains as a drawback, but the size of the space is smaller than in any other type of percentage representation. For the problem considered the search space is \((58-30+1)*58^3 = 5658248\). On the other hand, sometimes an expansion of the search space is useful to solve a problem; the simplex method is a good example of that.

Anyhow, there is another reason that makes the percentage representation attractive. Suppose that the effect of each component of the HPIS system over the unavailability and cost is not the same, due to the layout of the system; in that case, the scheduling should consider the impact or weight of each component. Therefore, it seems logical that some efficient percentages for certain \(T_1\) might have a big chance to be good when \(T_1\) changes. Moreover, exploring the space using percentage representation is a good way to approximate the limits of \(T_1\). Remember that the original limit of \(T_1\) was 365 days, but beyond 58 days, any feasible individuals could not be found. A fast exploration using percentages offers an estimate of where are the limits of feasibility.
Finally, it is possible to integrate the above ideas taking advantage of the good features of each one. This third alternative consists in keep the double-loop configuration and export some of the good individuals founded in one subspace to the following, scaling them. This alternative has the advantage that the search space remains equal to the original one; although if the smallest subspace is selected to start the process, the algorithm should consider that it is not possible to scale a smaller space into a bigger one perfectly.

Summarizing, the three alternatives considered are:

1. Exploring each disjoint subspace using a double-loop configuration where the external loop fixes $T_1$ and the inner nested loop is a multiple-objective evolutionary algorithm. (MOCA1)
2. Exploring each disjoint subspace using a double-loop configuration, exporting some good individuals from the one subspace to the next one, scaling them. (MOCA2)
3. Unify the search space adopting percentage representation and decoding the chromosomes according to eq. 2. (MOCA3)

4 IMPLEMENTATION

The alternatives described above were implemented in C++ using NSGA-II [3] as a multiple-objective evolutionary algorithm. The constraint handling technique is the constrained domination described also in [3]. For the sake of simplicity the alternatives are labeled MOCA1, MOCA2 and MOCA3 respectively.

For more details about the implementation of MOCA1 see [8].

In general, implementation of alternative MOCA2 requires scaling, nevertheless, due to $T_1$ is incremented in one unit from one subspace to the following, i.e. the search space is explored in strict order, the scaling was not necessary for MOCA3.

The chromosomes for the three alternatives were represented using integer variables. The crossover operator is a one-point crossover. For mutation, two strategies were implemented: with probability 0.4 a uniform mutation is applied, that is, one value from the list of possible states is uniformly selected. Otherwise, i.e. with probability 0.6 a triangular operator is applied (see Fig. 2). Note that the length of the triangle base is equal to the number of valid states that the integer variable can take. The total area is one, so a random $U(0,1)$ area number equivalent to an cumulative probability is generated. If that area is lower than the cumulative probability of the current value, the new state that replaces the current one is the state that generates the maximal triangle which area is lower than the random cumulative area generated early (see fig 2, shaded area). Likewise, if the random number is bigger than the cumulative probability of the current state, the new state is that with the minimal cumulative area larger than the random one (fig 2, dotted area).
5 RESULTS

In order to assess the quality of each approach, a reference set was obtained with a long run of MOCA1 (18000 evaluations of objective function).

5.1 $\varepsilon$-Dominance

According to [9], given two vectors $f, g \in \mathbb{R}_+^m$ and $\varepsilon > 0$, $f$ is said to $\varepsilon$-dominates $g$, denoted as $f \preceq \varepsilon - g$, iff $f_i < g_i \cdot (1+\varepsilon)^{-1}, \forall i \in \{1,2,...,m\}$ in a minimisation case (or iff $f_i < g_i - \varepsilon_i$ for an additive $\varepsilon$-dominance). Thus, the scalar $\varepsilon$ represents a proportion of how far is one vector from the other. In other words, $\varepsilon$ can be used to measure if the difference between two vectors is representative.

In [5], the authors proposed the generation of a $\varepsilon$-approximate Pareto set, by dividing the objective space into $\varepsilon$-boxes and comparing boxes instead of solutions, as a way of assure convergence to the real Pareto frontier. Therefore, the idea was employed to build and archiving strategy. In the present work, the methodology described in [5] is utilized to filter the final output in terms of a decision maker’s accuracy criteria.

Analyzing the differences between solutions, one can conclude that some of them have no practical meaning. For example, the first two vectors obtained with MOCA1 in 18000 evaluations have a mean unavailability of 0.0000988879 and 0.0000983942 respectively. Clearly, a difference of nearly 5.0e-7 is not significant in terms of unavailability. Thus, these two quantities are indifferent for the Decision Maker (DM).

With this idea in mind, some kind of clustering or filtering technique is necessary to reduce the outcomes to only significant solutions. Hence, the $\varepsilon$-approximate Pareto set concept is used here in a different way, i.e. not to assure convergence as in [5], but for filtering the outcomes in a simple way: the objective space is divided into $\varepsilon$-boxes, which represent a region of indifference for the DM, and then the outcome is reduced discarding those solutions that belong to dominated $\varepsilon$-boxes. Note that this kind of filtering is faster than simple clustering.

---

1 The concept of $\varepsilon$-dominance is expressed like $f_i < \varepsilon \cdot g_i$ in [9]. For the sake of convenient, we use the above formulation, which is mathematically equivalent.
For doing so, one type of $\varepsilon$–dominance and one value of $\varepsilon$ were selected for each objective according to custom DM practices, i.e. differences under the second significant digit are irrelevant for unavailability, whereas an amount of 10€ is considered the lower significant difference for comparisons effects. Consequently, proper values of $\varepsilon$ were selected to assure the described accuracy (see table 2), then each vector was related to its corresponding $\varepsilon$-box, calculating the coordinates like:

$$coordinate_i = \left[ \frac{\log(\text{max}_i)}{\log(1 + \varepsilon)} \right]$$

if multiplicative $\varepsilon$-dominance is considered, and

$$coordinate_i = \frac{\text{max}_i}{\varepsilon_i}$$

if additive $\varepsilon$-dominance is employed.

Observe that the values of unavailability were scaled to avoid problems with logarithmic. Finally, only the vectors belonging to non dominated $\varepsilon$-boxes were conserved. Figure 3 shows the original outcomes, whereas figure 4 shows the filtered outcomes in terms of $\varepsilon$-box coordinates.

Notice that mixed criteria (multiplicative and additive $\varepsilon$-dominance) were employed to define the $\varepsilon$-boxes. To the best knowledge, this is the first time when the $\varepsilon$-dominance is used in this way.

### 5.2 Outcomes comparison

In order to assess what outcome is better, comparisons of the filtered outcomes with the reference set were performed using the $\varepsilon$-Indicator ($I_\varepsilon$) [9]. The $\varepsilon$-Indicator is defined in [9] as

$$I_\varepsilon (A, B) = \inf_{\varepsilon \in \mathbb{R}} \{ \forall z^2 \in B \exists z^1 \in A : z^1 \succeq_\varepsilon z^2 \}$$

thus the smaller the indicator the better.
Figure 3: Original outcomes obtained with MOCA1, MOCA2 and MOCA3

Figure 4: Filtered outcomes using ε-dominance and ε-boxes (axes labels are coordinates)
Table 3 presents the values of each indicator, showing that MOCA3 performed better than the others did, even when the search space for this approach is the bigger. MOCA2 performed better than MOCA1 in terms of convergence, but MOCA2 could not find non-dominated solutions for \( T_1 = 58 \) days.

Obviously, the hierarchy obtained depends on the level of indifference chosen by the DM. While the needed accuracy is greater, the approaches will show more difference in their performance. Conversely, when the needed accuracy is smaller, the performances will be more similar.

<table>
<thead>
<tr>
<th>Approach</th>
<th>( I_\varepsilon ) (MOCA, Reference)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOCA1</td>
<td>1.0467</td>
</tr>
<tr>
<td>MOCA2</td>
<td>1.0353</td>
</tr>
<tr>
<td>MOCA3</td>
<td>1.0238</td>
</tr>
</tbody>
</table>

Table 3: \( \varepsilon \)-Indicator measures for the 3 approaches

The results obtained encourage the use of percentage representation in more complex problems. However, it is convenient to assess the balance between the size of the search space and the quality of the outcome in other scenarios. In particular, for the problem considered in this work, some constraints can be relaxed. For instance, the proportion between \( T_1 \) and \( T_2 \) and \( T_3 \) is fixed (\( T_2 = T_3 = 3 \times T_1 \)), but this proportion could be variant, with the consequent expansion of the search space. For this kind of scenarios, the strengths and weaknesses of each approach should be considered.

Notice also that, despite of the search space size, the percentage representation has the advantage that the limits of feasibility can be unknown before the search -for the type of constraints considered in this problem- without modifying the way of performing the search, whereas for MOCA1 and MOCA2 the feasibility is an important issue. The results presented were obtained searching only over the portion of the search space that produces feasible solutions in objective space, but if this information were not be available before the search, MOCA1 and MOCA2 would be searching solutions without success.

Finally, it is important to mention that the use of \( \varepsilon \)-dominance as a tool for filtering and comparing outcomes seems to be of great utility, especially due to the incorporation of decision-making criteria.

6 CONCLUSIONS

In the present article three alternatives for representing a multiple-objective floating-bound scheduling problems were proposed. Two of them was implemented and compared with another alternative developed in a previous work.

To simplify the comparisons of outcomes, the concepts of \( \varepsilon \)-dominance and \( \varepsilon \)-approximate Pareto set were employed as a filtering technique, using both additive and multiplicative \( \varepsilon \)-dominance in the same outcome. The values of \( \varepsilon \) for each objective were selected according to DM criteria. As a result, the original outcomes were reduced significantly, showing that the \( \varepsilon \)-dominance, as used in this work, is a useful alternative for filtering real-problems outcomes.
The $\varepsilon$-Indicator was employed to assess the convergence of outcomes obtained with each studied approach. The percentage representation shows the better performance in approximating the Pareto frontier in a short run, pointing out that the unification of the search space by means of percentages was useful for this particular problem.

Further studies must be developed to assess the balance between search space size and quality of the approximation in bigger real problems. Besides, the integration of $\varepsilon$-dominance with DM criteria should be considered in these future works.

REFERENCES