A Clustering Method Based on Dynamic Self Organizing Trees for Post-Pareto Optimality Analysis

Oswaldo Aguirre MS\textsuperscript{a,*}, Heidi Taboada PhD\textsuperscript{b}

\textsuperscript{a} Computational Science Program. The University of Texas at El Paso, 500 West University, El Paso, TX 79968-0521, United States of America.

\textsuperscript{b} Department of Industrial, Manufacturing and Systems Engineering. The University of Texas at El Paso, 500 West University, El Paso, TX 79968-0521, United States of America.

Abstract

Multiple objective optimization involves the simultaneous optimization of several objective functions. Solving this type of problem involves two stages; the optimization stage and the post-Pareto analysis stage. The first stage focuses in obtaining a set of nondominated solutions while the second one involves the selection of one solution from the Pareto set. Most of the work found in the literature focuses in the first stage. However, the decision making stage is as important as obtaining the set of nondominated solutions. Selecting one solution over others, or reducing the number of alternatives to choose from is not a simple task since the Pareto-optimal set can potentially contain a very large number of solutions. This paper introduces the dynamic self organizing tree algorithm as a method to perform post-Pareto analysis. This algorithm offers two main advantages: there is no need to provide an initial number of clusters, and at each hierarchical level, the algorithm optimizes the number of clusters, and can reassign data from previous hierarchical levels in order to rearrange misclustered data. The proposed method is tested in a well-known multiple objective optimization problem in order to show the performance of the algorithm.

© 2011 Published by Elsevier B.V.

Keywords: Multiple objective optimization, post-Pareto analysis.

1. Introduction

Solving a multiple objective optimization problem consists in obtaining a set of solutions called nondominated solutions or Pareto-optimal solutions. Several methods to obtain nondominated solutions have been proposed in the
literature, however, little prior work has been done on the post-Pareto analysis stage. To choose one solution over others from the Pareto set can be a very challenging task since this set often contains a large number of solutions. At this stage, the decision-maker selects feasible solutions according to criteria which depend of explicit objective function preferences. In order to make the section of a feasible solution a manageable task, the set of Pareto solutions has to be filtered or reduced to a small number of representative solutions. The selection of solutions from the Pareto set is called post-Pareto optimality analysis. To perform post-Pareto optimality, this work proposes to use a data mining approach called dynamically growing self-organizing tree (DGSOT) to classify Pareto-optimal solutions into clusters in order to intelligently reduce the size of the set and obtain representative solutions. The DGSOT algorithm constructs a hierarchical tree from top to bottom by division. At each hierarchical level, the algorithm optimizes the number of clusters, and can reassign data from previous hierarchical levels in order to rearrange misclustered data. Each leaf of the tree represents a cluster, each cluster is a subset of nondominated solutions from the original set of solutions. Therefore, each leaf in the final tree is a nondominated solution.

To demonstrate the performance of this algorithm, the all-terminal network reliability problem [1], will be analyzed. The remainder of the paper is as follows. Section 2 presents some of the previous works that have been published in multiple objective decision making. Section 3 is completely dedicated to explain the DGSOT algorithm. In Section 4 the numerical results are presented. This work presents some conclusions at the end.

2. Background & Previous work.

Many problems in engineering involve optimization. This work refers to the selection of the optimal solutions among a set of possible alternatives. A natural scenario in optimization is to have more than one objective to optimize simultaneously, usually those objectives are in conflict with each other. The difference between single objective and multiple objective optimization can be explained with the following example. Consider a single objective optimization problem in which the objective to be optimized is the total system cost. The solution to this problem will be the solution that achieves the lowest cost. In this case, just one solution can be selected (assuming that two different solutions cannot have the same cost). In contrast, consider a multiple objective optimization problem with two objectives to be optimized simultaneously, the system failure rate and the system cost. An optimal solution to this problem is the one that achieves the lowest failure rate and the minimum cost. However, this solution usually does not exist because optimizing one objective involves decreasing the value of the other objective. Figure 1 shows two optimal or nondominated solutions for this example. One solution has a better cost than the other solution, but worse failure rate than the other solution. The natural question here is to determine which solution is better. The answer will be both. Both solutions are considered to be optimal and are called nondominated solutions. Therefore, the solution of a multiple objective optimization problem is set of nondominated solutions. This set is also known as the Pareto front or Pareto set of solutions.

![Fig 1: Nondominated Solutions](image)

Figure 2 shows three different examples for other three different MOOP [1,3,4]. Each point in each plot represents one Pareto solution of the problem.
Due to the importance to solve multiple objective optimization problems (MOOP), there are several models in the literature that address different MOOP in different fields. For instance, Marler and Arora [2] presented a survey of different methods used to solve this type of problems. The majority of these methods are designed to generate a set of nondominated solutions.

Venkat et al [5] addressed the post-optimality analysis stage by introducing the Greedy Reduction (GR). This method works by obtaining subsets of Pareto solutions based on maximizing a scalarizing function. Venkat used a ranking approach to define their functions. Another approach was presented by Padhye et al [6]. This work proposed a mutation driven hill climbing local search using achievement scalarizing functions to refine the solutions from the Pareto set. Kacem et al [7] developed a hybrid approach using Fuzzy Logic and evolutionary algorithms (EAs) to obtain a satisfactory set of solutions.

Clustering is another approach to reduce the number of solutions to analyze. Once the clusters are formed, a reference solution (closest solution to the cluster centroid) from each cluster is selected. Using this approach the number of solutions is reduced to the number of clusters. In other words, the reduced Pareto set contains as many solutions as clusters were formed. There are several clustering methods that have been developed to classify data. Some of the most common clustering methods will be discussed in next section. Taboada and Coit [8] used the \(k\)-means method to cluster the Pareto set and obtain a smaller set of solutions to analyse. Clustering based methods have the main disadvantage that the number of clusters or groups has to be defined at the beginning of the procedure. Taboada and Coit [8] suggested the use of silhouettes values [9] to define what number of clusters is appropriate for a specific set of data. However, a method that automatically defines the number of clusters is desired. Figure 3 shows the main idea of the use of clustering to reduce the Pareto set and analyze a smaller number of representative solutions.

The post-Pareto optimality stage can be defined as a classification problem in which a specific amount of solutions have to be analyzed. The main objective of this work is to use a well-known hierarchical clustering approach to group the Pareto set. Each group or sub-set will have a representative solution (closest point to the centroid) reducing the number of solutions to choose from. The clustering method used in this work is the dynamically growing self organizing tree (DGSOT) algorithm. DGSOT is a hierarchical clustering method than has some advantages over other well-known clustering methods.

2.1 Clustering
Cluster analysis can be defined as the grouping of a set of data into subsets (called clusters) so that data (solutions) in the same cluster are similar in some sense. The two main branches of clustering algorithms are partitional and hierarchical methods.

The k-means algorithm is probably the most widely known partitional clustering technique [10]. The grouping is done by calculating the centroid for each group, and assigning each observation to the group with the closest centroid.

Hierarchical algorithms find successive clusters using previously established clusters. These algorithms usually are either agglomerative ("bottom-up") or divisive ("top-down"). Agglomerative algorithms begin with each element as a separate cluster and merge the element into successively larger clusters. Divisive algorithms begin with the whole set and proceed to divide it into successively smaller clusters (like DGSOT). A more detailed description of these methods is presented by Fung (2001) [10]. Some of the most important disadvantages of both methods are presented in Table 1.

<table>
<thead>
<tr>
<th>K-means</th>
<th>Hierarchical clustering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of k clusters have to be pre-defined</td>
<td>It doesn’t provide a discrete number of clusters. Clusters</td>
</tr>
<tr>
<td>Fixed number of clusters can make it difficult to predict what k should be.</td>
<td>need to defined with cut-offs</td>
</tr>
<tr>
<td>Different initial partitions can result in different final clusters. It is helpful to rerun the program using different values of k to compare results.</td>
<td>Cannot return to previous hierarchical level to reassign</td>
</tr>
<tr>
<td></td>
<td>misclustered data</td>
</tr>
<tr>
<td></td>
<td>Selection of split points is critical.</td>
</tr>
</tbody>
</table>

In order to overcome these drawbacks, the DGSOT algorithm was selected to perform post-Pareto optimality analysis. The algorithm will be described in next section.

3. **Dynamically growing self organizing tree (DGSOT)**

DGSOT is a hierarchical clustering method originally developed by Luo *et al* [11] based in the work by Dopazo and Carazo [12]. The objective of the method is to organize data without the necessity to introduce parameters to the method such as the number of clusters. This type of clustering can be classified as a form of self-organization. Self-organization is the process in which a pattern appears in a system without a central authority or external element imposing it through planning. Some of the main theory of self organization is presented by Heylighen (2001) [13], Heylighen (2009) [14]. The DGSOT algorithm has some characteristics that recall some aspects of self-organization.

The DGSOT grows vertically and horizontally. In each vertical growth, the DGSOT adds two children to the leaf whose heterogeneity is greater than a threshold ($T_R$) and turns it into a node. At each horizontal growth, the DGSOT dynamically finds the proper number of children (sub-clusters) of the lowest level nodes. Each vertical growth step is followed by a horizontal growth step. This process continues until the heterogeneity of all leaves is less than a threshold $T_R$. At the beginning, all the data belongs to a root node. The behaviour of the algorithm is presented in Figure 4.

![Figure 4: DGSOT Algorithm](image-url)
In the algorithm, each leaf represents a cluster that includes all data associated with it. The reference vector of a leaf is the centroid of all data associated with it. Therefore, all reference vectors of the leaves form a Voronoi set of the original dataset, and each internal node represents a cluster that includes all data associated with its leaf descendants.

During the vertical growth, a leaf is converted into a node if Heterogeneity (Average distance between the leaf node and the input data associated) > $T_R$, and then two leaves are added to the node. In a horizontal growth, a cluster separation criterion is used (CS). This measure represents the relative separation of the centroids for each leaf. Both procedures are represented in Figure 5.

$$d_i = \frac{\sum_{j=1}^{D} d(x_j, n_i)}{D}$$

Where:

- $d(x, n)$: distance between data $x$ and the leaf $n$
- $D$: total # of input data assigned to the leaf $i$
- $n_i$: $i$th leaf

$$CS = \frac{E_{min}}{E_{max}}$$

Where:

- $E_{min}$: min length edge in the Minimum spanning tree (MST) of the two nearest centroids
- $E_{max}$: max length edge in the Minimum spanning tree (MST) of the two most far centroids

Another important aspect of the algorithm is the K-Level up distribution mechanism that allows the algorithm to reclassify misclustered data of earlier stages. For this step, and for a selected node, its k-level up ancestor node is determined. The sub-tree rooted by the ancestor node is taken into account and data associated with the selected node is distributed among all the leaves of the sub-tree. The algorithm ends when there is no possibility to grow vertically nor horizontally.

4. Numerical Example

This example considers the all-terminal network reliability problem. This case considers 10 nodes and three objectives to be optimized simultaneously (reliability, total system cost, and weight). The data for each link of the network is not presented in this paper but for more information about the data, the reader may refer to [1]. The results are presented in Fig 6. The first picture shows the original data and the second one shows the clustered data.

Figure 7 shows the selected solutions for each cluster. The set was reduced from 78 solutions into just 4 solutions.
5. Conclusions

In the present paper, the DGSOT algorithm is used as a post-Pareto analysis method to reduce the size of the Pareto set of optimal solutions. In this case, the decision maker can analyze a smaller set of representative solutions instead of the whole Pareto front.

References