

A Data Mining-Based Strategy for Direct Multidisciplinary Optimization

Hongyi Xu, Ching-Hung Chuang, and Ren-Jye Yang
Ford Motor Co

ABSTRACT

One of the major challenges in multiobjective, multidisciplinary design optimization (MDO) is the long computational time required in evaluating the new designs' performances. To shorten the cycle time of product design, a data mining-based strategy is developed to improve the efficiency of heuristic optimization algorithms. Based on the historical information of the optimization process, clustering and classification techniques are employed to identify and eliminate the low quality and repetitive designs before operating the time-consuming design evaluations. The proposed method improves design performances within the same computation budget. Two case studies, one mathematical benchmark problem and one vehicle side impact design problem, are conducted as demonstration.

CITATION: Xu, H., Chuang, C., and Yang, R., "A Data Mining-Based Strategy for Direct Multidisciplinary Optimization," *SAE Int. J. Mater. Manf.* 8(2):2015, doi:10.4271/2015-01-0479.

1. INTRODUCTION

Multidisciplinary design optimization (MDO) has been widely studied and practiced in solving complex engineering design problems. Basically, existing methods of MDO falls into two categories. The first category is the metamodel-based optimization, which is long-established and widely practiced in engineering design problems [1,2,3]. However, metamodels always have the accuracy issue as it is an approximation of the real model [4]. The errors of the metamodels may mislead the optimization search, and result in biased results for design optimization. Real simulations are still needed to confirm the designs obtained from the metamodel-based optimization, which is referred as "virtual designs". The second category is to improve the efficiency of direct MDO [5, 6]. With the recent progresses in optimization algorithms and High Performance Computing (HPC), Direct MDO draws engineers' attention again as a promising optimization strategy. The basic idea of Direct MDO is to integrate optimization algorithms and the simulation codes directly. Real simulations are operated for each design evaluation such that true responses of new designs are always obtained throughout the entire optimal searching process.

Application of Direct MDO is not an easy task due to the complex design landscape and the extremely long computational time required by the high fidelity simulations. Heuristic search algorithms such as Evolutionary Algorithm (EA) [7], Genetic Algorithm (GA) [8] Simulated Annealing (SA) [9] and Particle Swarm Optimization (PSO) [10] algorithm are widely used in solving real engineering problems, because they are able to overcome the local optimums. With sufficient computational resources, the design space will be fully explored to find a large number of feasible designs of high

performances. One common feature of the heuristic search algorithms is the randomness in the process of generating new designs. For example, GA generates new designs (children) by applying random crossover and mutation on existing designs (parents); Simulated Annealing (SA) generates new designs by randomly perturbing existing designs' locations in the design space. Such randomness is the root cause of why heuristic algorithms have the ability of exploring a wide design space and avoiding being trapped in local optimums. However, the downside is the slow convergence. Oftentimes, the randomly generated new designs are very close to the existing designs that already appeared in the optimization history. Evaluation of such "near duplicates" is a waste of computation resources as it will not provide further knowledge about the design landscape. Therefore, "near duplicate" is identified as one of the major limitations to the MDO's efficiency.

In order to improve the optimal search efficiency by minimizing such ineffective function evaluations, multiple "smart" algorithms have been established and implemented in commercial software [11,12,13], which are able to adjust the algorithm parameters automatically at different locations on the complex design landscape. On the other hand, there are increasing interests in applying data mining and machine learning techniques to improve traditional optimization algorithms. Data mining-based methods provide additional knowledge in guiding optimal search, while the resultant computational costs are negligible compared to the advanced simulations in design evaluations. For example, clustering methods are used to find the nearest neighbor in the process of optimizing zero-skew routes [14]. It is also used for progressively searching of neighborhood of existing designs [15]. K-means method has been applied to determine the algorithm parameter (mutation rate and

crossover rate) automatically [16]. A few works have been accomplished on utilizing data mining techniques for eliminating the number of function evaluation. The promising design regions are identified before calling the time consuming function evaluation. Wang et al. [17] has developed a dynamic clustering-based differential evolution algorithm, which applies hierarchical clustering method on current generation of designs to identify the promising design region. The algorithm is able to gradually changes from explorative search at the early stage to exploiting solution with high precision in the end. However, this algorithm doesn't fully utilize the historical information of the entire optimization process. Only one generation is considered in the data mining step. Another question is the choice of cluster number in design clustering. The number of clusters is pre-determined without considering the dynamics of the dataset, so it lacks of adaptivity. This work only handles “box-constraint” problems. It is not applicable to the problems of searching feasible designs in the nonlinearly constraint design space.

With a given computation budget (total number of simulations), this work focuses on reducing the number of optimization iterations/generations, which is approximately proportional to the overall turnaround time. A large number of designs are evaluated in parallel in each iteration/generation to fully utilize the capability of parallel computing on HPC. In this paper, a data mining-based strategy is developed to improve the efficiency of multiobjective MDO. This strategy has the capability of fully utilizing the entire optimal search history as the database to identify ineffective new designs such as “near duplicates” before operating the time consuming function evaluation. The proposed strategy is applicable to improve any the traditional heuristic optimization algorithms, to achieve the goal of exploring larger design space and finding better designs within a small number of optimization iterations. The remainder of this paper is organized as follows: Section 2 introduces the data mining-based strategy. In Section 3, two benchmark problems are studied to demonstrate the proposed method. Section 4 is the summary.

2. DATA MINING-BASED STRATEGY

This section presents a clustering and classification-based method to identify the “near duplicates” in designs, and then to replace them with “non-duplicates” located in the promising design regions, before operating design evaluations (calling time-consuming advanced simulation codes).

2.1. Basic Idea and Workflow

The similarity between designs is quantified by their distances in the normalized design space. Figure 1 describes the definition of “near duplicates”. The existing known designs (“individuals”) are classified into multiple clusters. The feasible designs and infeasible designs are clustered respectively. Designs in the same cluster have high similarity in both design variable values and performances. For any newly generated designs in the optimization process, it will not be preferred for evaluation if its distance to the nearest cluster centroid is smaller than a predetermined value (e.g. 90% quantile of individual-to-centroid distances).

Given the data of previous designs and the correspondent performances (e.g. the initial design population generated by DOE), the promising region for future optimal search can be predicted from the distribution of existing designs in the design space. However, the non-intelligent traditional optimization algorithms are not able to consider such information when generating new designs. Take GA as an example, it is possible that some of the new designs from crossover and mutation are quite similar to historical designs that are already evaluated. The algorithm will still call the simulation codes to evaluate such “near duplicates”. Such unnecessary design evaluations lower the searching efficiency. It is preferred that new designs will explore the unknown design region so that the knowledge of design landscape is maximized within a given number of function evaluations. Figure 2 shows how the proposed strategy works in GA. Each newly generated design will be checked whether it is a “near duplicate” or not. If yes, the algorithm will go one step back and regenerate a new design until it is not similar to any existing designs with known performances.

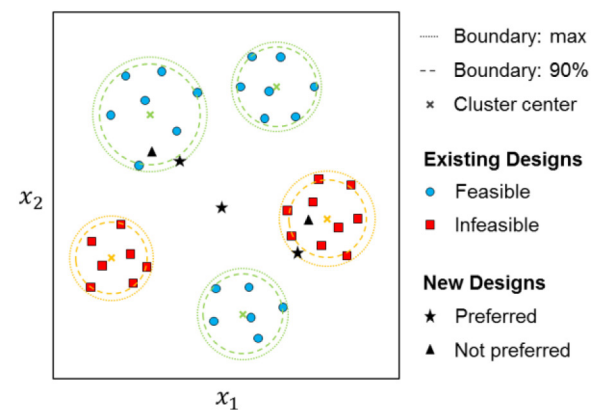


Figure 1. Similar designs (“near duplicates”, black triangles) and “non-duplicated” (black stars)

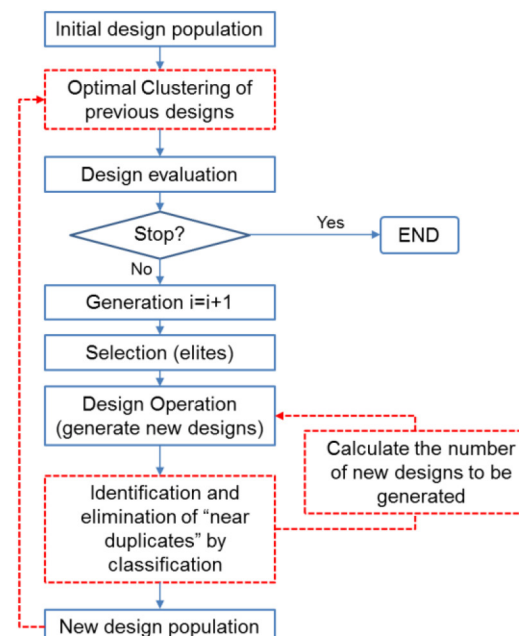


Figure 2. Flowchart of clustering and classification-based strategy

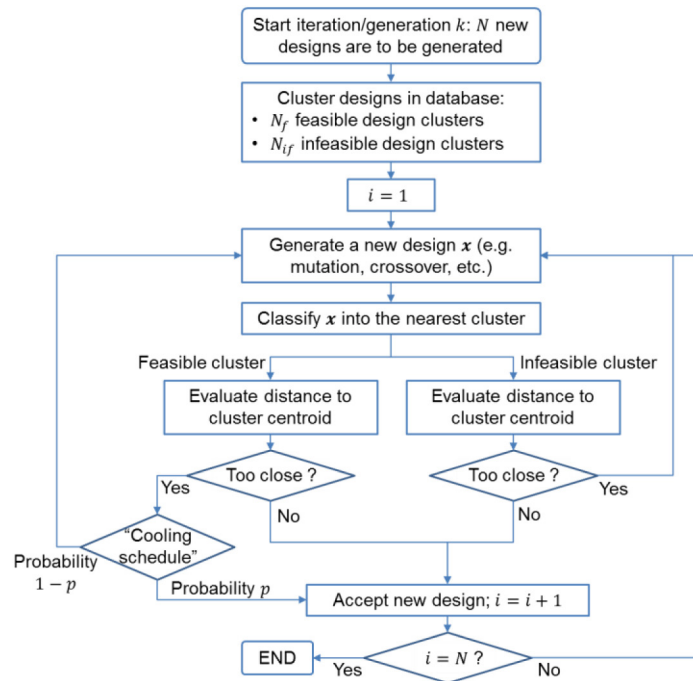


Figure 3. Workflow of the proposed strategy: identification and replacement of “near duplicates”

A more detailed description of the new features marked by red dashed lines in Figure 2 is provided as follows. Figure 3 describes how to apply data mining methods to improve the optimization process. Before running generation k of any heuristic optimization algorithm, all historical designs evaluated in the previous $k - 1$ generations are collected as one dataset. This dataset is referred as “database” in this paper. The purpose is to generate N new promising designs in generation k for evaluation.

1. The first step is to conduct cluster analysis on the database. The feasible designs and infeasible designs are divided into multiple clusters respectively, according to their similarity in the normalized design space. k -means method is employed for clustering. The optimal cluster number is automatically determined (more details are provided in Section 2.2).
2. Generate a new design x . Classify x into an existing cluster. For example, Naive Bayesian Classifier is recommended in this paper.
3. In case when x is classified into an infeasible cluster. Calculate its distance to the cluster centroid. If it is too close (e.g. smaller than the 90% quantile of individual-to-centroid distances), reject this design and go back to Step (2) to regenerate a new design; Otherwise, this new design is accepted for evaluation.
4. In case when x is classified into a feasible cluster. Calculate its distance to the cluster centroid. If it is far from the centroid (e.g. larger than the 90% quantile of individual-to-centroid distances), accept this design for evaluation. If it is too close to the centroid, it will be accepted with a probability of p , which means exploitative searching in feasible regions; on the other hand, it will be rejected with a probability of $1 - p$, and go back to Step (2) to regenerate a new design. The probability p is defined as:

$$p = 1 - e^{-\frac{k}{2 \cdot k_{max}}} \quad (1)$$

k is the number of the current iteration, k_{max} is the maximum number of iterations. This formulation is inspired by the “Cooling Schedule” of SA [9], which allows both explorative and exploitative search happen during the entire process. At the beginning of the optimization process, it has a higher probability of accepting new designs far from the existing clusters for exploration purpose. This probability will decrease along the optimization process to allow more exploitative search inside the feasible regions.

5. Repeat Step (2), (3) and (4) until N new designs are generated. Then operate design evaluations for these N new designs as generation k .

2.2. Optimal k -Means Clustering Method

In Step (1) of the proposed strategy, the optimal cluster number is identified for k -means clustering of design database. The optimal cluster number will lead to a small within-cluster variation (high similarity inside the same cluster) and a large between-cluster variation (low similarity between different clusters). Cluster analysis with the optimal cluster number minimizes the errors of miss-classification of new designs.

As shown in Figure 4, cluster analysis is operated for i_{max} times with cluster number $i=1,2,\dots,i_{max}$ respectively. i_{max} is the user-defined maximum cluster number. The averaged Silhouette Index $\bar{S}_i$ is evaluated for each i . Silhouette Index measures how similar an individual is to its own cluster [18]: