A Multi-Criteria based Selection Method using Non-dominated Sorting for Genetic Algorithm based Design

Erkan Gunpinar*,[1] Shahroz Khan[2]
[1]Mechanical Engineering Dept., Istanbul Technical University, Istanbul, Turkey
[2]Dept. of Naval Architecture, University of Strathclyde, Glasgow, United Kingdom

Abstract

The paper presents a generative design approach using a Genetic Algorithm (GA), which is structured based on a novel offspring selection strategy. The proposed selection approach commences while enumerating the offsprings generated from the selected parents. Afterwards, a set of eminent offsprings is selected from the enumerated ones based on the following merit criteria: space-fillingness to generate as many distinct offsprings as possible, resemblance/non-resemblance of offsprings to the good/bad individuals, non-collapsingness to produce diverse simulation results and constrain-handling for the selection of offsprings satisfying design constraints. The selection problem itself is formulated as a multi-objective optimization problem. A greedy technique is employed based on non-dominated sorting, pruning, and selecting the representative solution. According to the experiments performed using three different application scenarios, namely simulation-driven product design, mechanical design and user-centred product design, the proposed selection technique outperforms the baseline GA selection techniques, such as tournament and ranking selections.

Keywords: Generative design, Genetic algorithm, Mating selection, Optimization, Non-dominated sorting, Angle-based pruning

1. Introduction

Optimization is the process of finding an alternative that is as fully perfect, functional, or effective as possible. A designer comes up with a new idea and tries different variations on an initial concept to improve it. However, he/she may not always anticipate all possible variations, as his/her intuition is limited. Therefore, an algorithm-driven design process can empower designers and achieve the desired objectives within given constraints. Genetic algorithm (GA) is an optimization technique based on the principles of genetics and natural selection. It can be employed in various engineering tasks such as design and computer-aided engineering. Starting with an initial population consisting of distinct designs and their fitness values, the population evolves under the specified selection rules. The work in this paper focuses on the selection technique of GA that is used in crossover mating. Rather than employing a probabilistic-based selection technique, as used in the baseline techniques (such as ranking and tournament selections), a systematic selection approach is employed. Offsprings generated in this way can better scan the design space, and therefore, more desirable offsprings are likely to emerge in terms of the desired objectives.

The proposed approach is developed for single-point crossover (SPC), which can also be customized and employed for the other crossover operators, such as two-point crossover. In SPC, one crossover point is selected: Chromosomes from the beginning to the crossover point are copied from one parent, and the rest are copied from the second parent. Probabilistic-based selection methods produce mating pairs based on the individuals' fitness values. The individual with the lowest fitness value has the greatest probability of mating. However, such a probabilistic approach may generate similar offsprings, and thus, it does not guarantee the generation of distinct offsprings in the resulting population.

The GA selection method proposed in this paper aims to generate a set of offsprings (a solution consisting of individuals) from the mating pool based on the follow-

*__Email:__ gunpinar@itu.edu.tr **Adress:** Inonu Caddesi, No:65, Gumussuyu, 34437, Istanbul, Turkey **Tel:** +90-212-2931300 **Fax:** +90-212-2450795

ing five *quality criteria*:

1. **Space-filling offsprings**: The offsprings obtained after SPC mating should be as different from each other as possible, which enables users to better explore a design space consisting of many offsprings. In this way, a more desirable offspring (i.e., offspring with a lowest fitness value) can be obtained in the design stage;

2. **Non-resemblance in individuals with higher fitness values**: The offsprings should resemble the individuals with higher fitness values (i.e., bad individuals) as little as possible. In other words, the offsprings should reside in the design space positions that are far away from the bad individuals;

3. **Resemblance in individuals with lower fitness values**: The offsprings should resemble the individuals with lower fitness values (i.e., good individuals). Good individuals are likely to be close to the optimum solution (i.e., the individual with the minimum fitness value) in the design space;

4. **Non-collapsing offsprings**: If two offsprings does not share the same parameter value, this characteristic is described as *non-collapsing (NC) offsprings*. Running two collapsing experiments might provide similar results, and ultimately, causes a waste of computational effort [1]. Although it is not always possible to produce non-collapsing offsprings in GA, it is preferable to generate NC offsprings as much as possible; and

5. **Feasible offsprings**: A design space consists of *feasible* and *infeasible* designs. A design is feasible if all the design constraints are satisfied; otherwise, it is infeasible. Feasible individuals may produce infeasible offsprings in SPC mating, which is undesirable.

The probabilistic-based GA selection techniques (such as tournament selection) give higher priorities for the parents having lower fitness values to mate. The second and third quality criteria favors the generation of offsprings, which should resemble/not resemble in individuals with lower/higher fitness values. This approach is similar like those of the probabilistic-based GA selection techniques. However, this paper further investigates the use of the space-fillingness and non-collapsing criteria in the GA selection process. Based on all quality criteria, we also propose a multi-objective GA selection strategy using Non-dominated Sorting.

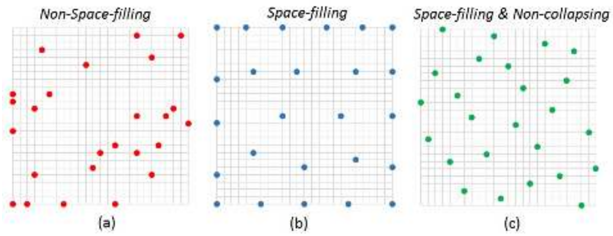

Figure 1: A two-dimensional ($2D$) case for the non-space-filling (a), space filling (b) and space-filling with non-collapsing (c) designs.

Fig. 1 illustrates a two-dimensional ($2D$) case for the non-space-filling (a), space filling (b), and space-filling with non-collapsing (c) designs. The second and third criteria are mainly considered by the baseline GA selection methods, whereas the first and fourth are not. A design test case (the vessel case [2]) is illustrated in Figure 2. While considering the above quality criteria to generate offsprings, their (minimum) fitness values obtained are investigated. An algorithm considering these criteria is executed 100 times. We have mostly seen decreases in or same fitness values at the end of the iterations in the algorithm runs. We think that the quality criteria considered enable to scan the design space well so that offsprings with minimum fitness values can be found. This claim is valid for 98 algorithm runs (among 100) for the test case in Figure 2.

In the proposed GA selection approaches, all possible offsprings in SPC mating are first generated. A desired number of offsprings is then sampled while considering the five criteria mentioned above. The research problem is formulated as a multi-objective optimization. A greedy approach is chosen to find the best offsprings, based on non-dominated sorting, pruning, and selecting the representative solution. Our method involves optimization process, therefore needs more computational time compared to the baseline GA selection methods. The proposed approach can be particularly useful in simulation-driven product design, in which high computational simulation times are required to analyze designs. For example, wind tunnel tests for car body and aircraft wings are costly and time-consuming, and therefore a limited number of designs can be tested. The proposed multi-criteria based GA selection algorithm can well scan the design space (unlike the probabilistic-based GA selection techniques such as ranking and tournament selections) and carefully select the designs, which can increase the possibility of exploring a more plausible design (i.e., a design with lower fitness value). Besides simulation-driven product design, the proposed technique is validated through mechanical design and

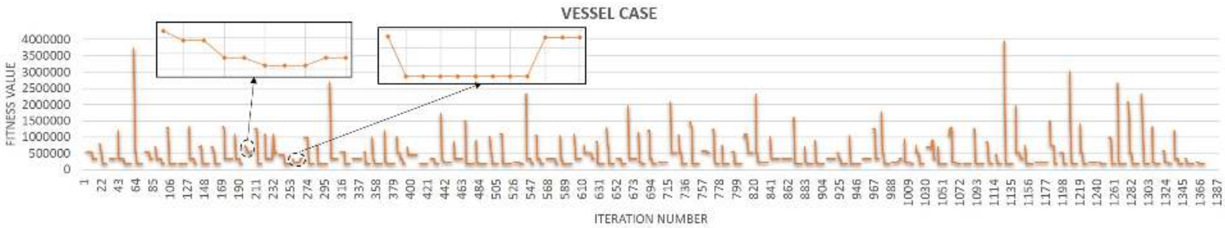

Figure 2: Plot of (minimum) fitness values obtained versus number of iterations for a test case (the vessel case [2]) when the quality criteria (such as space-fillingness, non-collapsingness) are considered in the offspring generation. An algorithm considering these criteria is run 100 times. (Minimum) fitness values (for the obtained offsprints) mostly decrease or are same at the end of iterations in the algorithm runs except two of the runs (shown with dashed black ellipses).

user-centered product design scenarios.

The remainder of this paper is organized as follows: Section 2 reviews the relevant literature. Section 3 explains the proposed sampling-based selection technique in crossover mating. The experiments and discussion are given in Sec. 4. Finally, concluding remarks and opportunities for future work are presented in Sec. 5.

2. Related works

The research in this paper is mainly relevant to generative design and GAs; thus, these fields represent the focus here.

2.1. Generative design

During the last decade, several advancements have been made in the field of generative design for various applications. Multiple techniques have been proposed by different researchers for architectural applications, and a few other techniques are based on the generative creation of a specific class of products. The development of generative systems has passed through various stages, mostly led by academic researchers and its theoretical implementation has already been widely recognized [3]. Krish [4] developed an exhaustive searched-based generative technique for creating design alternatives. In this technique, designs were randomly generated in the design space based on a user-defined threshold value, which was set on the Euclidean distance between the generated designs. A major drawback of this technique was that it was based on an exhaustive search and explored a limited region of the design space, thereby preventing the user from generating creative designs. Gunpinar et al.[5] introduced a generative design and drag coefficient prediction system for Sedan car side silhouettes based on computational fluid dynamics. A sketching system called *DreamSketch* was developed by Kazi et al. [6] to support generative design in the conceptual phase. In DreamSketch, a user generated an initial design by sketching, and its alternatives were then generated in the sketched context. To utilize this system, the user requires digital sketching abilities. An optimization based generative design algorithm was proposed by Khan and Awan [7] to explore continuous and discrete parametric design spaces. Despite being an efficient technique for creating visual variations of designs, [7] cannot be used to explore shapes for any performance objective.

A symmetric-based generative system was proposed by Sousa and Xavier [8] for the digital fabrication of geometric shapes, such as rhombicuboctahedrons, cuboctahedrons, and triangular prisms. Adam et al. [9] proposed a biologically motivated algorithm for the generative creation of leaf venation patterns. Shea et al. [10] and Turrin et al. [11] developed performance-driven generative design systems to create lightweight architectural structures. Different researchers have also proposed generative design techniques to create site layouts [12], as well as energy efficient and eco-friendly building designs [13]. Recently, Gunpinar and Gunpinar [14] proposed a design sampling technique for computer-aided design (CAD) models to produce space-filling designs via a particle tracing method. Khan and Gunpinar [15] also suggested another CAD model sampling technique based on the teaching-learning-based optimization of Rao et al. [2]. The designs sampled via their method have a semi-Latin Hypercube property and space-filling [1]. Khan also generated [16] space-filling designs via spatial simulated annealing [17] for customer-centered products. Finally, Dogan et al. [18] presented a sampling approach for deriving profiles of an existing product design using profile similarities and primitive shapes.

In the literature, techniques like shape syntheses [19, 20], shape grammars [21] and L-systems [22] have been widely utilized by researchers to develop generative systems. A shape grammar is a generative method for representing and creating a design by embedding geometric logics/rules, and this approach has been utilized in different applications, such as architectural design [23], product design [24], 2D automotive design [25] and embroidery design [26]. Despite being its usage for different applications, shape grammar's usage is limited to industry. This is because of its computational complexity and difficulty in developing user interfaces [27]. Furthermore, shape grammar requires a different set of geometric rules for each application, which requires special expertise [4]. An L-system is a variation of shape grammar that has been used for different design problems, such as computer pattern design [28] and complex city planning/simulation [29]. L-systems are also based on the production rules, which are applied in the form of a string. In these techniques, designs are generated by applying string rewriting mechanisms [30]. Among the other methods, shape syntheses are superior in terms of creating a higher variation of a design. However, these techniques can only be utilized for creating alternatives of existing shapes, in which the system is first trained on a large dataset of existing shapes and are then synthesized to create variant alternatives.

2.2. Genetic algorithms

GAs represent one of the powerful meta-heuristic optimization techniques, which was originally developed by John Holland in the 1960s, and since then, they have been used ([31]), improved ([32, 33]), adapted ([34]), and hybridized ([35]) with other evolutionary algorithms for a wide variety of applications. Mostly, the revised or new GAs are proposed from researchers by making improvements on the mutation/crossover operator or on the selection techniques. The performance of GAs largely relies on these selection methods and operators. Over the years, there have been a lot of efforts made to improve their performance.

A unimodal distribution crossover operator (UNDX) was introduced by Ono et al. [36], which used multiple parents and created offspring around the center of mass of the parents. Deep and Thakur [37] proposed a real-coded Laplace crossover (LX) operator to improve the overall performance of the algorithm. A Taguchi-based simulated binary crossover operator was proposed by Subbaraj et al. [32] to improve exploitation and robustness of the algorithm. Elfeky et al. [38] developed a triangular crossover (TC) operator that can be used for the constrained optimization problems. In the TC, two parents were selected from the feasible region and one parent from the infeasible region. Recently, Marandi and Smith [33] proposed a fluid Genetic Algorithm (FGA) with an improved crossover operator called *Bron-An-individual*. The new operator enabled FGA to have better global learning and diversity rate. The choice of an appropriate selection method is essential, as the general performance of GA depends on it. The selection methods are usually implemented for reproduction (i.e., parent selection), as good parents can produce better offspring. Several selection methods, such as Genitor (steady state) selection, tournament selection, truncation selection, linear and exponential rank selection, roulette wheel selection, and stochastic universal sampling have been widely used for the different optimization problem. A detail description and comparison of these selection methods can be found in [39].

Zhong et al. [40] compared the tournament selection with the proportional roulette wheel. It has been found that the former was more efficient in convergence than the later. Julstrom [41] compared the computational efficiency of the linear and exponential ranking with the tournament selection method. Based on the results of his studies, it was found that tournament selection is computationally efficient compared with the ranking selection methods. Mashohor et al. [42] examined the performance of inspection systems using the deterministic, tournament, and roulette wheel methods. From the results, these researchers observed that the deterministic method was superior compared with the other two techniques. A comparative study of the proportional, ranking, tournament and Genitor selection methods was carried by Goldberg and Deb [43]. Their study showed that the ranking and tournament selection outperformed the others in terms of convergence speed.

Goh et al. [44] proposed a new selection method called *sexual selection*, which was inspired by Charles Darwin's selection concept. In this method, the sex of an individual was first determined based on problem-specific knowledge. A pair of individuals were then selected in a sequential fashion for matting. Moreover, Anand et al. [45] developed a novel, efficient selection method called *Alternis*. Here, the population was first sorted in descending order; the individuals were then arranged in alternating fashion, with some left-right arrangement according to their fitness values. Afterward, an individual along with its left and right individuals was chosen and placed in the matting pool. An improved roulette wheel selection method was proposed by Jadaan et al. [46] to increase the gain of resources, reliability, and diversity as well as to decrease the uncertainty in the selection process. Affenzeller and Wagner [47] developed a new self-adaptive selection method for GAs. Most of the work in the literature on the selection methods have been based on the comparison of their performance in different optimization problems. However, there is no substantial amount of research work done on creating new and effective selection techniques for GAs.

3. A sampling-based selection method for genetic algorithms

3.1. Problem formulation

A design is a variation of a product whose geometric model is represented using *design parameters*. Once the lower and upper bounds (i.e., parameter ranges) for the design parameters are set, a design space can be formed in which infinite number of designs exist. A design space, D, is an n-dimensional space where each design parameter stands for a dimension in D and n is an integer. Let X be a design, which is represented by the design parameters $(x_1, x_2, ..., x_n)$. The lower and upper bounds for the design parameters are denoted by $(l_1, l_2, ..., l_n)$ and $(u_1, u_2, ..., u_n)$, respectively. Furthermore, $(c_1, c_2, ..., c_w)$ denotes a set of design constraints where w is an integer.

Let P be the initial population containing Y designs, $(X^1, X^2, ..., X^Y)$, with their fitness values, $(f^1, f^2, ..., f^Y)$. The population is divided into two sub-populations based on the fitness values: $\overline{P}$ consists of $\overline{Y}$ good designs and $\underline{P}$ involves $\underline{Y}$ bad ones ($Y = \overline{Y} + \underline{Y}$). Using SPC, a new population is formed from the good designs in $\overline{P}$. $n-1$ SPC points can be defined for a design pair, and each SPC mating can yield $2*(n-1)$ different offsprings. The number of possible SPC mates is $\binom{\overline{Y}}{2}$. As a result, $2*(n-1)*\binom{\overline{Y}}{2}$ offsprings can be produced after SPC mating of the good designs in $\overline{P}$. If the numbers Y and n are high, the number of producible offsprings will be high. In this work, offsprings are sampled among all producible ones while taking the four criteria into account.

3.2. Sampling criteria

Five quality criteria are considered to choose K offsprings among the $2*(n-1)*\binom{\overline{Y}}{2}$ offsprings producible after the SPC mating, as detailed in the next sections.

3.2.1. Space-filling offsprings

A careful selection of the offsprings is of primary importance. *Space-filling offsprings* spread in the design space and allows having a global design space exploration. Audze-Eglais potential energy [48, 1] is employed for this and is based on the analogy of minimizing the forces between the charged particles. The potential energy is at the minimum, and therefore, the particles are in equilibrium. The potential energy E_1 between the chosen offsprings is computed using Eq. 1:

$$E_1 = \sum_{p=1}^{K-1} \sum_{q=p+1}^{K} \frac{1}{M(p,q)^2}, \tag{1}$$

where

$$M(p,q) = \sqrt{\sum_{i=1}^{n} (\overline{x}_i^p - \overline{x}_i^q)^2}. \tag{2}$$

Here, the function $M(p,q)$ computes the distance in the scaled design space $\overline{D}$ between the offsprings p and q. The coordinates $(\overline{x}_1, \overline{x}_2, ..., \overline{x}_n)$ are the scaled values varying between 0 (the lower bound) and 1 (the upper bound) for the design parameters $(x_1, x_2, ..., x_n)$. $\overline{x}_p^i$ and $\overline{x}_q^i$, respectively, denote the i^{th} coordinate of the offsprings X^p and X^q.

3.2.2. Non-resemblance in bad designs

To generate offsprings that are far away from the bad designs, the Audze-Eglais potential energy is again employed. Bad designs push offsprings away from themselves, which is favored using Eq. 3. Here, $M(p,r)$ computes the distance between the offspring p and the bad design r as in Eq. 2.

$$E_2 = \sum_{p=1}^{K} \sum_{r=1}^{\underline{Y}} \frac{1}{M(p,r)^2}. \tag{3}$$

3.2.3. Resemblance in good designs

In SPC, an offspring is generated by mating two good designs. To favor the generation of offsprings resembling the good designs, the metric in Eq. 4 is introduced. The energy E_3 is minimized if the parents of the offsprings have lower fitness values. Here, f_1^p and f_2^p denote the fitness values for the two parents of the offspring p.

$$E_3 = \sum_{p=1}^{K} (f_1^p + f_2^p). \tag{4}$$

3.2.4. Non-collapsing offsprings

Equation 5 is introduced to avoid collapsing offsprings. For every two different designs (p and q) in the offspring list, it is checked whether they share the same value for each design parameter. Collapsing designs are penalized using a piecewise function (g) in Eq. 6.

$$E_4 = \sum_{p=1}^{K} \sum_{q=p+1}^{K} \sum_{i=1}^{n} g(\overline{x}_i^p, \overline{x}_i^q), \tag{5}$$

where

$$g(\overline{x}_i^p, \overline{x}_i^q) = \begin{cases} 1, & \text{if } \overline{x}_i^p = \overline{x}_i^q \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

3.2.5. Feasible offsprings

The design space consists of feasible and infeasible designs. As SPC mating can produce infeasible offsprings from feasible designs; infeasible ones should be penalized during the offspring sampling stage. Penalty function methods [49] can be utilized for the satisfaction of the design constraints. An energy function, E_5, is computed separately for each offspring using Eq. 7. $h(c_i)$ is a piecewise function that has positive values if the constraint c_i is not satisfied for an offspring. Otherwise, it is zero (see Eq. 8). z_{c_i} denotes the equation for c_i, which can be 0 (greater/smaller than 0) for the equality (inequality) constraints. For example, if the constraint c_1 is $x_1 > x_3$, z_{c_1} is the absolute difference between x_1 and x_3, which is as follows: $z_{c_1} = |x_1 - x_3|$.

$$E_5 = \sum_{p=1}^{K} \sum_{i=1}^{w} h(c_i). \tag{7}$$

where

$$h(c_i) = \begin{cases} 0, & \text{if the constraint } c_i \text{ is satisfied} \\ \left|z_{c_i}\right|, & \text{otherwise.} \end{cases} \tag{8}$$

3.3. The offspring sampling technique

Let ζ be the offspring list containing all producible offsprings in SPC mating of the good designs in $\overline{P}$. The objective is to sample/choose K offsprings, ($X = X^1, X^2, \ldots, X^K$), in ζ based on the criteria in Sec. 3.2. A multi-objective optimization problem can be formulated as follows:

$$min \rightarrow E, \tag{9}$$

$$Subject\ to\ \ X \subseteq Z \in D. \tag{10}$$

where

$$E = (E_1(X), E_2(X), E_3(X), E_4(X), E_5(X))^T \tag{11}$$

Z in Eq. 10 denotes the feasible design space, which consists only of feasible designs. $*^T$ stands for the transpose operator. $E_1(X)$, $E_2(X)$, $E_3(X)$, $E_4(X)$, and $E_5(X)$ represent the five energy functions for the K offsprings X.

There is no one best solution to a multi-objective optimization problem; rather, a set of trade-off solutions exist called *non-dominated or Pareto-optimal solutions* [50]. In other words, no solution dominates or is better than the other solutions in the set. In this work, a greedy approach is chosen to sample K offsprings, as summarized in Algorithm 1. Here, s denotes a solution consisting of K offsprings and S is a list containing solutions. Let M and L be the number of offsprings in ζ and Pareto-optimal solutions, respectively. Starting with a randomly generated solution consisting of K offsprings, the offsprings in the solution are replaced one by one with the offsprings in ζ and inserted into S, which contains $K*M$ new solutions at the end. The Pareto-optimal solutions are then found, which are denoted by S_p. The solutions are pruned using pruning techniques. This procedure is repeated until the stopping criterion (SC) is met. Finally, the *representative solution* is selected among the obtained Pareto-optimal solutions.

Algorithm 1 The Offspring Sampling Algorithm

Select K random offsprings (i.e., the solution s) in ζ.
Insert s into the solution list S^P.
while The algorithm is not stopped **do**
  for $h = 1$ to L **do**
    Set s to the h^{th} element of S^P.
    for $j = 1$ to K **do**
      for $k = 1$ to M **do**
        Replace the j^{th} offspring of s with the k^{th} offspring of ζ.
        Insert s into the list S.
      end for
    end for
  end for
  Find the Pareto-optimal solutions, S^P, for S.
  Prune the solutions in S^P and update them.
end while
Choose the representative solution.

3.3.1. Domination criteria

Let $\widehat{X}$ and $\widetilde{X}$ be the two solutions consisting of K offsprings. $\widehat{X}$ is said to dominate $\widetilde{X}$ (i.e., $\widehat{X} \prec \widetilde{X}$, $\widehat{X}$ *non-*

dominated by $\widetilde{X}$) if the following condition is true:

$$\begin{aligned}\widehat{X} \prec \widetilde{X} \Leftrightarrow [E_1(\widehat{X}) \le E_1(\widetilde{X})] \wedge [E_2(\widehat{X}) \le E_2(\widetilde{X})] \wedge \\ [E_3(\widehat{X}) \le E_3(\widetilde{X})] \wedge [E_4(\widehat{X}) \le E_4(\widetilde{X})] \wedge \\ [E_5(\widehat{X}) \le E_5(\widetilde{X})] \wedge \left[\sum_{i=1}^{5} [E_i(\widehat{X}) < E_i(\widetilde{X})]\right].\end{aligned}$$

Here, $\widehat{X}$ is no worse than $\widetilde{X}$ in all energies, and $\widehat{X}$ is better than $\widetilde{X}$ in at least one energy. A solution is said to be *Pareto optimal* if it is not dominated by any other solution.

3.3.2. Pruning solutions

Pruning algorithms are applied to select a subset of Pareto-optimal solutions. Two types of pruning are employed, which are outlined as follows:

- **Pruning noisy solutions:** During the offspring replacement (see line 8) in the sampling algorithm (Algorithm 1), offsprings that have same design parameter values can be obtained. This will produce a solution with an extremely high value of E_1 as the denominator in Eq. 1 becomes zero. Such solutions are undesirable, and thus, they should be pruned. If a solution has an energy value α times greater than the median of the solutions in any energy, it will be discarded. In this study's experiments, α is set to 100; and

- **Angle-based pruning:** An angle-based pruning algorithm with specific bias parameter is employed as that of Sudeng and Wattanapongsakorn [50]. The pruning is performed using Eq. 12. The solution $\widetilde{X}$ will be discarded if $\widehat{X}$ is not worse than $\widetilde{X}$ and the angle θ between $\widetilde{X}$ and $\widehat{X}$ is less than the threshold angle δ for at least one of the energies. The geometric angle in Eq. 13 is denoted by θ_i, where i is the i^{th} energy. ΔE_j is the difference between the i^{th} energy values for $\widetilde{X}$ and $\widehat{X}$. To determine the threshold angle δ_i in Eq. 14, all non-dominated solutions are first sorted in ascending order for each energy. The inter-quartile range of the sorted data for each energy is then calculated. Finally, the inter-quartile range of average distance of the i^{th} energy value between two consecutive non-dominated solutions is computed. τ ranging from 0 to 1 is the bias intensity of each energy. A higher value for τ indicates a less preferred energy.

$$\widehat{X} \prec \widetilde{X} \Leftrightarrow \sum_{i=1}^{5} ([E_i(\widehat{X}) \le E_i(\widetilde{X})] \wedge [|\theta_i(\widehat{X},\widetilde{X})| \le |\delta_i|]) > 0, \tag{12}$$

where

$$\theta_i = \tan^{-1}\left[\frac{\sqrt{\sum_{j=1,j\neq i}^{5} (\Delta E_j)^2}}{\Delta E_i}\right]. \tag{13}$$

$$\delta_i = \left[\tan^{-1}\left(\frac{IQS_i}{IQ_i}\right)\right]^{\tau}. \tag{14}$$

3.3.3. Selection of the representative solution

One way to select a single solution (i.e., the representative solution) from the set of Pareto-optimal solutions is to choose the one that minimizes the distance to the *ideal solution* similar to that described by Cheikh et al. [51]. The ideal solution represents the solution that simultaneously optimizes all the objectives being considered, and can be non-existing. To set the energies ($E_1 - E_5$) for the ideal solution, a large number (i.e., 10000) of solutions (i.e., a set of K offsprings) are first randomly generated, and the energies for the solutions are then computed. The energies for the ideal solution are the minimum energies among the energies of all randomized solutions. The representative solution of S^P has the closest proximity to the ideal solution. In other words, the distance between a solution and the ideal solution is the Euclidean distance between the energies of the two solutions. Here, each energy term represents a separate dimension. Note that all the energy values should be normalized before employing the distance function. Scaling is performed using minimum (i.e., lower bound) and median (i.e., upper bound) energy values for the 10000 randomized solutions. The maximum energy values in the randomized solution set is not considered as the upper bound, as one energy term can dominate another one due to the existence of the Pareto-optimal solutions involving very large energy values particularly for E_1.

3.3.4. Stopping criteria

The while loop in Algorithm 1 runs until one of the SCs is met; SCs are as follows:

- **SC1:** The energy values for the representative solutions in three consecutive runs are very similar.

The similarity means that the absolute difference between two energies is less than a threshold, such as 0.001;

- **SC2:** The algorithm enters a loop, producing the same solutions. Let S_{P_i} be the Pareto-optimal solutions obtained after the i^{th} iteration of the while loop in Algorithm 1. After a certain period, S_{P_i} and $S_{P_{i+a}}$ includes the same solutions following the a iterations, where a is an integer; and

- **SC3:** The execution time of the algorithm after the iteration is greater than the user-defined time t. If t is not defined by the user, the above two criteria should be satisfied to stop the proposed algorithm. This criterion is introduced because the processing time may be high for some experiments.

3.4. The extended offspring sampling algorithm

Let n_t be the number of iterations the while loop in the above algorithm does. The computational complexity for the algorithm is high, which is $O(n_t * L * K * M)$. Therefore, a more practical version of the algorithm is proposed. Instead of using all Pareto-optimal solutions (obtained after applying pruning methodology) in Line 4 of the offspring sampling algorithm 1, a representative solution is selected among them. Line 5 of the offspring sampling algorithm is revised as 5' and is as follows:

5'. Set h to be the representative solution of the list S^P.

The infeasible offsprings produced in the SPC mating are also removed, thus the generation of only feasible offsprings is guaranteed. Furthermore, noisy solutions are solely pruned in the extended algorithm as a single solution (i.e., the representative solution) is selected from the Pareto-optimal solutions, and therefore, there is no need to perform angle-based pruning. SC2 or the following stopping criterion is met for the convergence of the extended algorithm:

- **SC4:** The energy values for the representative solution do not decrease further in an algorithm run.

4. Test cases and problems

The proposed GA selection techniques are validated for different applications, namely simulation-driven product design, mechanical design and user-centered product design.

Table 1: Design parameter ranges for the dental implant model.

Parameter ranges			
$7.0 \leq x_1 \leq 11.0$	$2.05 \leq x_2 \leq 2.5$	$0.75 \leq x_3 \leq 1.5$	$0.2 \leq x_4 \leq 0.6$
$0.05 \leq x_5 \leq 0.2$			

4.1. Simulation-driven product design

CAE simulations (such as FEA and computational fluid dynamics (CFD)) can sometimes take a large amount of time to analyze a single design. Therefore, it is preferable to find a good solution by testing a limited number of designs via CAE analysis. This section involves there product design cases, where CAE analysis is crucial. A dental implant and a car chassis design cases will be outlined here.

4.1.1. Dental implant

A dental implant for the mandibular first molar tooth was utilized, which was employed by Usta and Onder [52] for material optimization using FEA [1]. Figure 3 (a) shows the CAD model for the implant with five design parameters. Here, the terms $x_1, x_2, \ldots, x_5$ denote the design parameters and represent the implant length, top radius, bottom radius, thread length, and thread height, respectively. Table 1 shows the design parameter ranges for this model.

Another application area for GA is the engineering optimization [53]. Here, the objective is to minimize the maximum stress in the bones. The initial population is obtained by randomly generating 30 designs with a non-collapsing property [1], which are tested using FEA. The population was divided into two sub-populations based on the maximum stress value (i.e., fitness value): If the value for a design was less than 10.0, the design was inserted into the population $\underline{P}$, which was used in SPC mating. Figure 3 (b) shows the FEA model [52], consisting of an implant, crown, abutment, screw, cortical bone, and trabecular bone. The generated mesh for the FEA model can be seen in Fig. 3 (c), and the load was 100 Newton axially applied on the crown, as shown in Fig. 3 (d). The side and bottom surfaces of the cortical bone were fixed, and thus, they had zero displacement.

4.1.2. Car chassis

A sprint car's chassis frame was also employed to validate the performance of the proposed technique [2]. The

[1] See `https://github.com/???`

[2] See `https://github.com/shahrozkhan66/Sprint_Race_Car_Chassis_Analysis`

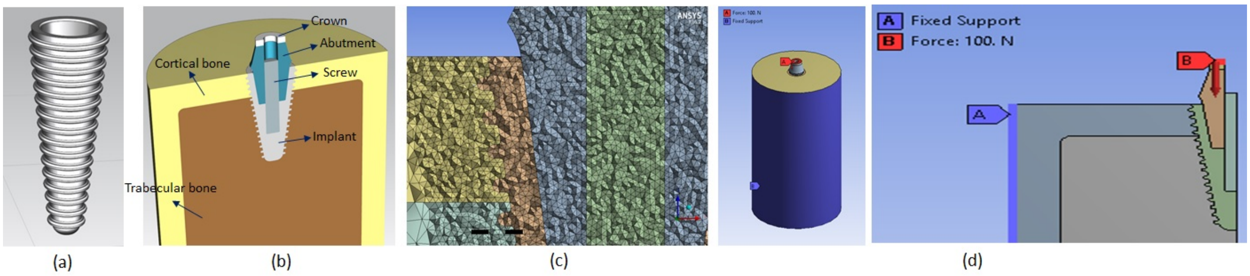

Figure 3: (a) Dental implant model. (b) An FEA model involves an implant, crown, abutment, screw, cortical bone and trabecular bone. (c) The mesh model for the FEA model. (d) Force loading conditions and boundary conditions. (Images taken from [52]).

Table 2: Design parameter ranges for the sprint car's chassis model.

Parameter ranges			
$15 \leq x_1 \leq 24$	$26 \leq x_2 \leq 35$	$58 \leq x_3 \leq 66$	$88 \leq x_4 \leq 95$
$2.5 \leq x_5 \leq 5$	$28 \leq x_6 \leq 35$	$0.5 \leq x_7 \leq 5$	$20 \leq x_8 \leq 24$
$22 \leq x_9 \leq 27$	$5 \leq x_{10} \leq 15$	$25 \leq x_{11} \leq 35$	$3 \leq x_{12} \leq 8$
$25 \leq x_{13} \leq 35$	$8 \leq x_{14} \leq 15$	$28 \leq x_{15} \leq 38$	$5 \leq x_{16} \leq 17$
$18 \leq x_{17} \leq 35$	$1 \leq x_{18} \leq 3.5$	$5 \leq x_{19} \leq 10$	$3 \leq x_{20} \leq 10$
$5 \leq x_{21} \leq 30$	$2 \leq x_{22} \leq 8$		

Table 3: Design parameter ranges for the honeycomb heat sink.

Parameter ranges			
$20.0 \leq x_1 \leq 40.0$	$6.0 \leq x_2 \leq 15.*$	$20.0 \leq x_3 \leq 40.0$	$0.0 \leq x_4 \leq 30.0$
$8000.0 \leq x_5 \leq 25000.0$			

chassis was designed according to the [54] and tested for shape optimization under the static torsional loading conditions. Figure 4 (a) shows the CAD model for the chassis frame with 22 design parameters. Here, the terms $x_1, x_2, \ldots, x_{19}$ are the design parameters, representing the horizontal/vertical dimensions of chassis's internal structure; x_{20} (r_{20}), x_{21} (r_{21}), and x_{22} (r_{22}) are the fillet radius of the chassis's boundary structure. Table 2 shows the ranges for the design parameters.

The objective for a chassis frame was to minimize the stresses produced under the static torsional loading conditions by rearranging the internal structure of the chassis and maintaining the out boundary of the chassis. In the chassis's structural analysis, the torsional test is one of the important tests, as this validates/rejects the chassis structure. For this test, the chassis was assumed to act as a cantilever beam with one end fixed and another end subject to torque about its longitudinal axis as shown in Fig. 4 (b). For the safe working of the sprint car, the chassis should able to resist the resultant shear stress. Like the dental implant model, for the initial population of designs, Latin Hypercube designs were randomly created and tested via FEA analysis under the clockwise moment of 316 Newton-meters around the longitudinal axis. The population was divided into two sub-populations, and designs with stress values less than $7E6$ Pascal were inserted into the population $\underline{P}$, which was used in SPC mating. The mesh results for the chassis are shown in Fig. 4 (c).

4.2. Mechanical design

Five different constrained benchmark mechanical design problems with linear and nonlinear constraints are used for the validation of the GA selection methods. A pressure vessel, a tension and compression spring, a welded beam, and a gear train test cases are described in Rao et al.'s work [2]. Furthermore, a honeycomb heat sink case is outlined in this section.

A heat sink example with hexagonal aluminum honeycomb fins, introduced by Subasi et al. [55], was also tested. The design parameters were the fin height x_0, fin thickness x_1, longitudinal pitch x_2, angle of attack x_3, and Reynolds number x_4. Figure 5 shows the CAD model for the honeycomb heat sink. Table 3 shows the design parameter ranges for the heat sink model.

The objective for the honeycomb heat sink design is to minimize the friction factor f, which has a mathematical model obtained using regression analysis in Subasi et al.'s work [55]. Latin Hypercube designs were randomly created and tested using the mathematical model. The population was divided into two sub-populations, and designs with frictional factors less than 0.4 were inserted into the population $\underline{P}$.

4.3. User-centered product design

In today's market, user preferences are important in product design [14]. GA can be used to learn these preferences via surveys. The designs can then be recommended to the users. For this purpose, a wine glass shape is designed using GA based on the SPC mating and proposed GA selection methods.

The wine glass model introduced by Gunpinar and Gunpinar [14], with 16 design parameters, is employed,

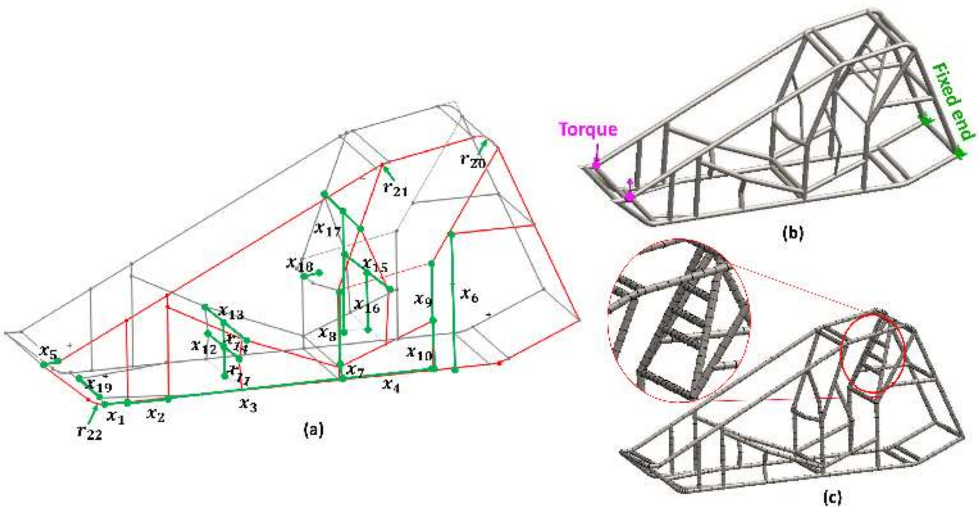

Figure 4: (a) Sprint car's chassis model represented using 22 design parameters. (b) Force loading conditions and boundary conditions. (c) The mesh model with circular beam elements for the sprint car chassis.

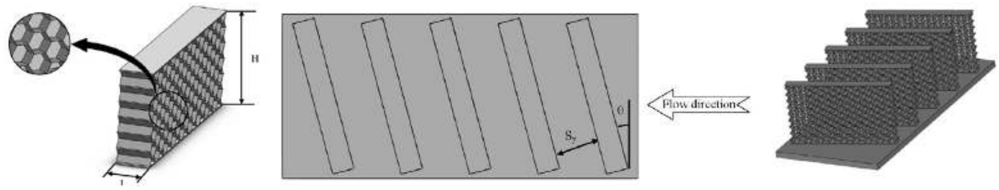

Figure 5: Structure of a honeycomb fin (left), top view (middle) and (c) perspective view of a heat sink configuration (right) (Images taken from [55]).

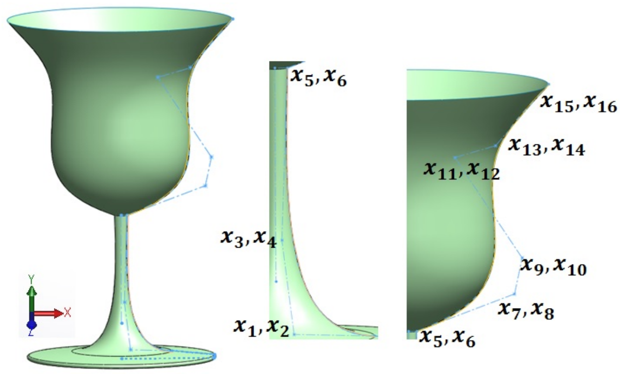

Figure 6: Wine glass model represented using 16 design parameters (Images taken from [14]).

Table 4: Design constraints for the wine glass model.

Geometric Constraints	
$\phi_1 : f(\phi_1) = x_2 + x_1/11.0 - 25.0/22.0 = 0$	$\phi_2 : f(\phi_2) = x_6 - x_4 \geq 0$
$\phi_3 : f(\phi_3) = x_{16} - x_{14} \geq 0$	$\phi_4 : f(\phi_3) = x_8 - x_6 \geq 0$
$\phi_5 : f(\phi_5) = x_{15} - x_{13} \geq 0$	

Table 5: Design parameter ranges for the wine glass model

Parameter ranges			
$0.0 \leq x_1 \leq 1.5$	$1.0 \leq x_2 \leq 1.5$	$0.0 \leq x_3 \leq 1.0$	$1.0 \leq x_4 \leq 15.0$
$0.5 \leq x_5 \leq 1.0$	$15.0 \leq x_6 \leq 25.0$	$0.5 \leq x_7 \leq 15.0$	$25.0 \leq x_8 \leq 35.0$
$0.5 \leq x_9 \leq 15.0$	$30.0 \leq x_{10} \leq 40.0$	$0.5 \leq x_{11} \leq 15.0$	$35.0 \leq x_{12} \leq 45.0$
$0.5 \leq x_{13} \leq 15.0$	$40.0 \leq x_{14} \leq 50.0$	$5.0 \leq x_{15} \leq 15.0$	$45.0 \leq x_{16} \leq 55.0$

as shown in Fig. 6. Two Bezier curves represent the glass, and $x_1, x_2, \ldots, x_{16}$ denote the X and Y coordinates of its points. Tables 4 and 5 show the geometric constraints and parameter ranges, respectively, for the model.

One of the popular application area of GA is the generation of user-preferred models, as described in Cluzel et al.'s work [56]. A wine glass model was employed in a user study, and three users scored the 20 designs based on their likes/dislikes. The population was divided into two sub-populations based on the participant scoring: If the score for a design was greater than 7.0, it was inserted into the population $\underline{P}$, which was utilized in SPC mating. The new populations, with a population size of 10, were then generated using the proposed GA selection technique and the baseline algorithms. The designs

Figure 7: User interface for the survey.

in the population were again scored by the participants. The scores were given based on a 0–10 scale (very poor: 0.0 – 2.9, poor: 3.0 – 4.9, fair: 5.0 – 6.9, good: 7.0 – 7.9, very good: 8.0 – 10.0). Note that the survey participants did not have any information about the techniques that were used to generate the models. Furthermore, the participants spent time observing several design options before starting the survey. Figure 7 depicts the user interface for the survey, which can be found on the web address https://goo.gl/forms/LiShOkMcDze8UQjN2. All the users were males without professional design experience, and they were aged 23-25 years. Finally, a different energy for E_3 was employed here as higher scores are better for the wine glass test case; the calculation is as follows, and this is comparable to that of Eq. 4:

$$E_3' = \sum_{p=1}^{K} (21 - f_1^p - f_2^p). \tag{15}$$

There were 120 designs in total, which were generated using the proposed selection technique with $\tau = 0.5$, $\tau = 0.75$, and $\tau = 1.0$, and the tournament, ranking and stochastic universal sampling (SUS) selection techniques. The designs generated using these techniques were randomly divided into two to prevent the users from scoring 120 designs at once. A 5-minute break was given after each part of the survey. To check the user's consistency, we duplicated the designs every 20 designs. The consistency score is expressed in percentiles and computed using Eq. 16, and let υ be the consistency score of a user for his selections, and υ_s is the difference in the scores given by the user for the duplicated design s.

$$\upsilon = 100 - (100 * \frac{\sum_{s=1}^{6} \upsilon_s}{6 * 10}). \tag{16}$$

5. Experiments and discussion

The results of the proposed GA selection techniques will be first given and compared with the baseline GA selection techniques. Computational time and convergence of the proposed methods will then be discussed.

5.1. *Results for the test cases and problems*

The results for the proposed selection techniques are compared with the tournament, ranking and SUS selection techniques. The results for the GA selection methods were obtained with the generation of 20 designs/offsprings (i.e., $K = 20$) in SPC mating. Note that the generated population using the techniques mentioned in this work (along with the initial population) for whole test cases can be found in the supplementary material of this paper.

5.1.1. *Simulation-driven product design cases*

Fig. 8 shows the results for the dental implant simulations. The objective here is to minimize the maximum stress on the dental bones. The best three designs (with the minimum maximum stresses) were produced by the extended offspring sampling ($SpmExt$). The best designs generated using the offspring sampling algorithm with $\tau = 0.75$ and $\tau = 1.0$ settings ($Smp0.75$ and $Smp1.0$, resp.) had lower maximum stress values than those generated using $Smp0.5$ and the baseline GA selection methods. Besides, the tournament selection method had better performance than the other selection techniques (i.e., ranking, SUS, and the proposed selection algorithm with $\tau = 0.5$). Fig. 9 shows the cortical bone models colored with von Mises stresses in MPA for the best two dental implant designs obtained using $Smp0.75$ and $Smp1.0$. Table 6 shows the energy

Table 6: Energy values for the genetic algorithm selection techniques.

Test cases	Methods	E1	E2	E3	E4	E5
Dental implant	*Smp0.5*	349.506	463.969	400.360	66	-
	Smp0.75	261.827	424.713	400.200	66	-
	Smp1.0	297.3	387.5	400.7	64	-
	SmpExt	361.8	383.5	402.4	64	-
	Ranking	300001515.5	490.0	342.4	74	-
	Tournament	300003399.3	423.9	338.4	73	-
	SUS	100003463.2	470.7	342.2	68	-
Chassis	*Smp0.5*	64.1	63.2	247.4	312	-
	Smp0.75	58.1	61.6	243.6	312	-
	Smp1.0	53.4	56.5	244.2	313	-
	SmpExt	55.5	56.8	243.6	312	-
	Ranking	79.4	47.0	219.6	317	-
	Tournament	100000056.1	48.1	226.3	297	-
	SUS	200000748.1	47.0	232.7	297	-
Vessel	*SmpExt*	592.9	269.6	268345881.0	52	0
	Ranking	300001491.2	346.9	116051491.1	59	0
	Tournament	300004302.3	349.6	91744169.7	60	186668.7
	SUS	300002357.6	352.3	107138699.3	59	0
Spring	*SmpExt*	3657.7	1300.1	8.7	46	0
	Ranking	702366800.0	3824.8	1.8	47	1.1
	Tournament	402090550.7	1183.7	1.7	49	0.8
	SUS	101107783.2	2167.0	1.9	46	1.9
Beam	*SmpExt*	3684.4	1070.9	356.9	63	0
	Ranking	400005591.8	1249.0	155.7	63	42073.2
	Tournament	600004778.8	1268.0	154.2	65	60353.3
	SUS	800041822.9	1146.9	153.4	63	61622.5
Train	*SmpExt*	1838.4	438.1	178665.0	119	0
	Ranking	700017024.0	434.1	134836.4	122	0
	Tournament	1000033620.7	426.3	134616.2	122	0
	SUS	1501124081.0	454.2	135373.1	122	0
Honeycomb heat sink	*Smp0.5*	404.4	279.1	15.4	66	-
	Smp0.75	250.0	221.2	15.3	66	-
	Smp1.0	284.2	241.5	15.2	64	-
	SmpExt	265.5	210.6	15.3	64	-
	Ranking	100001153.3	258.6	12.0	70	-
	Tournament	500000959.5	237.4	11.9	75	-
	SUS	400000616.4	232.9	12.2	70	-
Wine glass - 1	*Smp0.5*	33.9	17.3	150.0	-	0.0
	Smp0.75	23.7	20.1	138.0	-	0.0
	Smp1.0	26.2	18.5	141.0	-	0.0
	Ranking	259.6	18.9	107.0	-	723.4
	Tournament	100000657.3	17.7	119.0	-	719.1
	SUS	300004823.3	21.8	103.0	-	650.2
Wine glass - 2	*Smp0.5*	36.9	31.1	122.0	-	0.0
	Smp0.75	43.8	30.7	121.0	-	0.0
	Smp1.0	22.9	27.8	125.0	-	0.0
	Ranking	100000126.5	35.5	102.0	-	629.2
	Tournament	100000238.9	32.8	127.0	-	662.5
	SUS	1700003040.2	30.7	120.0	-	617.4
Wine glass - 3	*Smp0.5*	37.0	26.5	217.0	-	0.0
	Smp0.75	46.6	26.6	217.0	-	0.0
	Smp1.0	31.3	24.4	218.0	-	0.0
	Ranking	100000278.6	30.1	125.0	-	694.1
	Tournament	78.5	27.3	128.0	-	572.2
	SUS	700009659.6	28.7	128.0	-	664.8

values for the designs obtained using the GA selection techniques. For the implant model, the baseline GA selection techniques produced very large values of E_1 as they were probabilistic-based and did not take space-filling criterion (i.e., E_1) into account while generating designs, and therefore they could produce similar designs. The designs generated by $SmpExt$ and the baseline GA selection techniques had the lowest values of E_2 and E_3, resp. In case of E_4, $SmpExt$ and $Smp1.0$ had the lowest values.

The simulation results of the sprint car's chassis are shown in Fig. 10. The best chassis designs were obtained when the proposed selection algorithm with $\tau = 0.75$ and $\tau = 1.0$ was utilized (see Fig. 11). The best chassis design can be seen in Fig. 11. However, the best designs of all methods except that of $Smp0.5$ had similar maximum stress values. $SmpExt$ and $Smp1.0$ generated chassis designs having the minimum values of E_1 (see Table 6), while the tournament and SUS selection techniques produced designs with very large values of E_1. On the other hand, the designs obtained using the tournament and SUS selection techniques had the lowest values of E_2, E_3 and E_4.

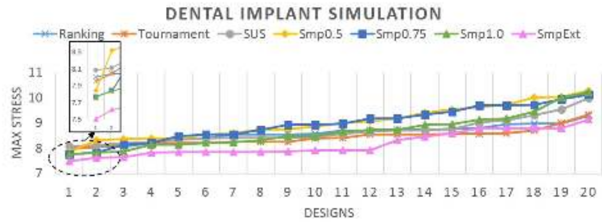

Figure 8: Simulation results for the dental implant designs generated using the GA selection techniques (i.e., maximum stress in the bones). $Smp0.5$, $Smp0.75$ and $Smp1.0$ denote the offspring sampling technique with the $\tau = 0.5$, $\tau = 0.75$ and $\tau = 1.0$ settings, respectively. $SmpExt$ stands for the extended offspring sampling technique.

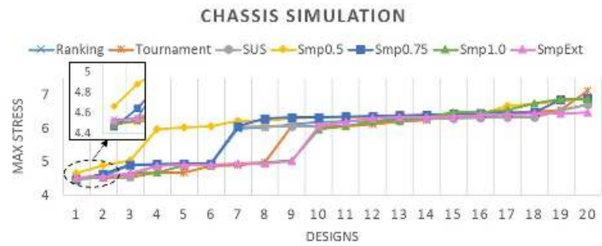

Figure 10: Simulation results for the car chassis models generated using the GA selection techniques (i.e., maximum stress).

5.1.2. Mechanical design problems

Fig. 12 shows the designs obtained for mechanical design problems. Except for the beam problem, $SpmExt$ algorithm produced the design having the minimum fitness value. The ranking, tournament and SUS selection methods generated the best designs only for two design problems. $SmpExt$ and Smp are compared using the honeycomb heat sink problem. $SmpExt$ and Smp with $\tau = 0.75$ ($Smp0.75$) generated the best solution. The best solution obtained using Smp with $\tau = 1.0$ had a fitness value closer to the best solutions obtained using $SmpExt$ and $Smp0.75$.

The energy values are also compared for the GA selection methods, and can be seen in Table 6. The baseline GA selection methods produced solutions with very high values of E_1. The solution generated by $SmpExt$ had a value of E_1 less than those obtained using $Smp0.5$ and $Smp1.0$, and greater than that of Smp0.75. Note that $SmpExt$ and $Smp0.75$ both generated the best so-

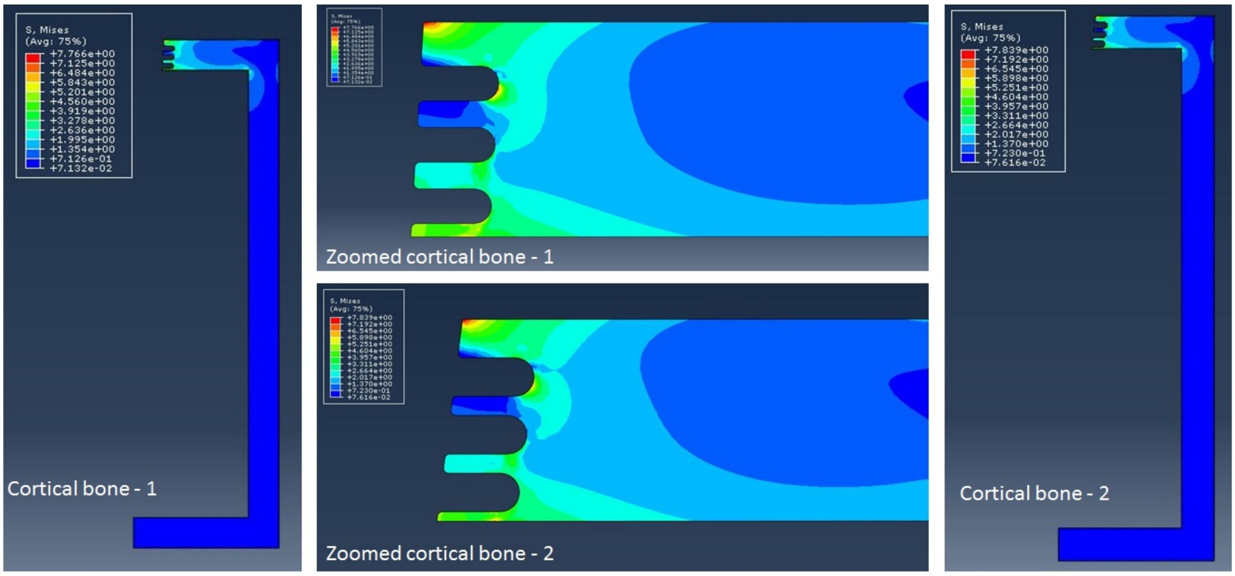

Figure 9: Cortical bone FEM results (i.e., von Mises stresses in MPA) for the best first two dental implant designs generated using the offspring sampling algorithm with the τ = 0.5 and τ = 0.75 settings.

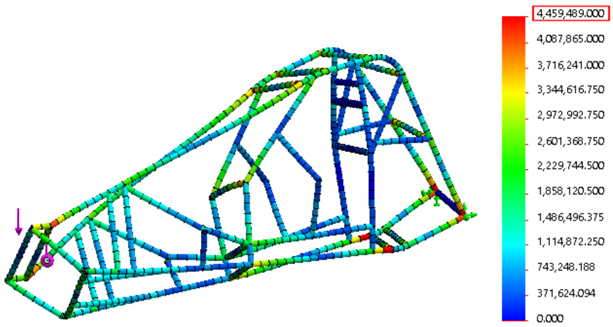

Figure 11: Best chassis design with a maximum stress of 4.459 MPa.

lutions in term of the fitness value. When E_2 values are compared, the solutions obtained using $SmpExt$ had the lowest values in three design problems. On the other hand, $SmpExt$ produced solutions with larger values of E_3 compared to the baseline methods. The solutions had the lowest values of E_4 when $SmpExt$ was utilized. Finally, most of E_5 values for the designs obtained from the GA baseline selection techniques were non-zero as the design constraints were not completely satisfied (see Table 6).

5.1.3. User-centered product design cases

Figure 13 shows the results for the wine glass user study conducted with three users. Here, 20 designs/offsprings (i.e., K = 20) were generated in SPC mating. Higher scores for the designs indicate the designs preferred by the users. According to the first user's scorings, the most preferred design was obtained when the proposed selection technique with τ = 0.5 was employed. The second most preferred designs were generated using the proposed selection technique with τ = 1.0 and the tournament selections. When the second user's scorings are observed, the proposed selection technique with τ = 0.75 and the SUS method generated the most preferred designs. The second most preferred design was also obtained using the proposed selection technique with τ = 0.75. According to the third user's scorings, the most preferred designs were obtained by the selection technique with τ = 0.75, ranking, and tournament selections. Figure 14 depicts the wine glass designs preferred by the first (a), second (b) and third (c) users. The consistency scores for the first, second, and third users were 93.3%, 88.3% and 95.2%, respectively. The selection times for the first population were 65, 75, and 192 seconds for the first, second, and third users, respectively. For the second population, they were 390-356 (the first-second part of the user study), 342-255 and 905-782 seconds, respectively.

The energy values for the designs obtained from the selection techniques can be seen in Table 6. $SmpExt$ and $Smp1.0$ generated chassis designs having the minimum values of E_1, while the designs obtained using GA baseline selection techniques had very large values of E_1. In case of E_2, all methods produced designs with similar values. On the other hand, the designs generated

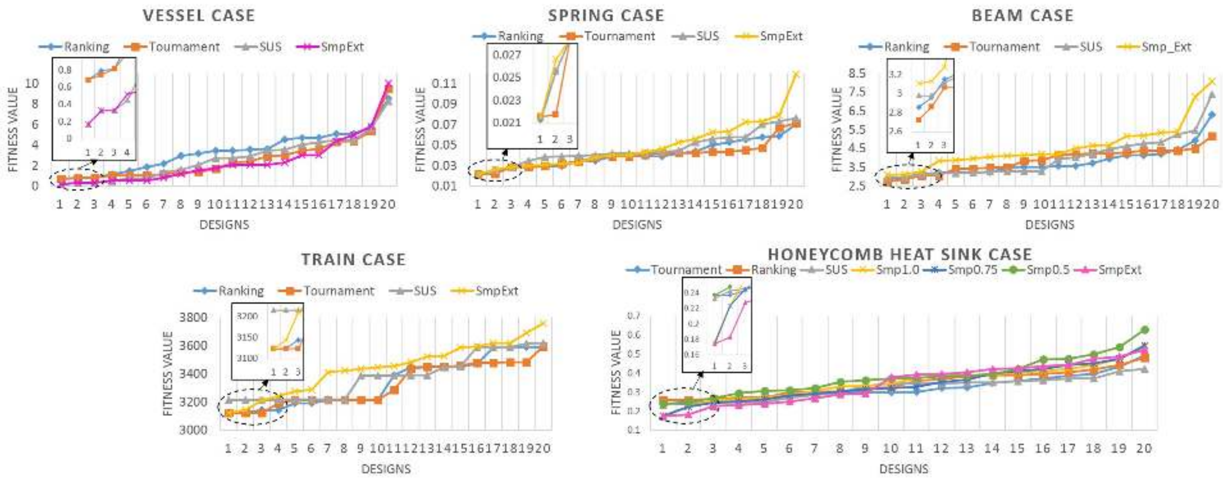

Figure 12: Results for the vessel (a), spring (b), beam (c), train (d), and honeycomb heat sink (e) cases.

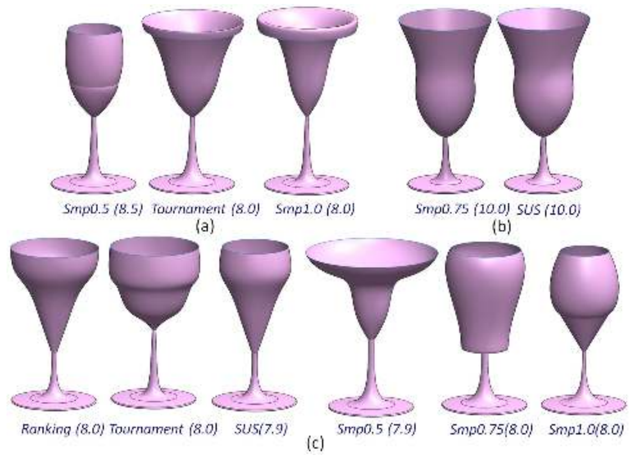

Figure 14: Wine glass models preferred by the first (a), second (b) and third (c) users.

Table 7: The designs with minimum fitness values obtained using the genetic algorithm selection methods.

Test cases	**GA selection methods**						
	Smp0.5	*Smp0.75*	*Smp1.0*	*SmpExt*	*Ranking*	*Tournament*	*SUS*
Dental implant	8.324	7.766	7.766	**7.503**	8.004	7.955	8.113
Chassis	4.661	**4.459**	4.479	4.518	4.53	4.503	4.513
Test cases	**GA selection methods**						
	SmpExt				*Ranking*	*Tournament*	*SUS*
Vessel	**0.17**				0.68	0.68	**0.17**
Spring	**0.021**				**0.021**	**0.021**	**0.021**
Beam	3.11				2.86	**2.72**	2.98
Train	**3123.63**				**3123.63**	3124.01	3215.84
Test cases	**GA selection methods**						
	Smp0.5	*Smp0.75*	*Smp1.0*	*SmpExt*	*Ranking*	*Tournament*	*SUS*
Honeycomb heat sink	0.237	**0.175**	0.177	**0.175**	0.258	0.237	0.233
Wine glass - 1	**10-8.5**	10-7	10-8	-	10-7	10-8	10-6
Wine glass - 2	10-8	**10-10**	10-9	-	10-9	10-9	10-10
Wine glass - 3	10-7.9	**10-8**	**10-8**	-	**10-8**	**10-8**	10-7.9

from the GA baseline selection techniques had lower values of E_3. Finally, it has been observed that the designs generated using the GA baseline selection techniques did not completely satisfy the design constraints in Table 4 (see E_5 values in Table 6).

5.2. Performance of the genetic algorithm selection techniques

We evaluated the performance of the GA selection methods. Table 7 shows the fitness values of the best designs generated using the methods for the test cases. Note that the best score for the design is subtracted from the maximum grade (i.e., 10) for the wine glass test case, as larger fitness values are preferable for this test case. $SmpExt$ and/or $Smp0.75$ generated the best designs in most cases. For the second wine glass user study, $Smp0.75$, $Smp1.0$, ranking, and tournament had the best designs. In contrast, the proposed method with $\tau = 0.5$ generated the best design in the first wine glass user study. When the overall results in Table 7 are seen, $Smp0.75$ exhibited better performance mostly compared with $Smp0.5$ and $Smp1.0$. It is thought that $Smp1.0$ prunes too many solutions, and suboptimal solutions can be obtained. While $Smp0.5$ prunes less solutions, and performs less iterations, thus only unmatured solutions can be obtained in a less time. The baseline GA selection methods had better performances for the vessel, spring, beam, train models, and the third wine glass user study.

5.3. Computational time and algorithm convergence

A PC with an Intel Core *i*7 6700 3.4 GHz processor and 16 GB memory was used in this study's experiments. The implementation was single-threaded. Ta-

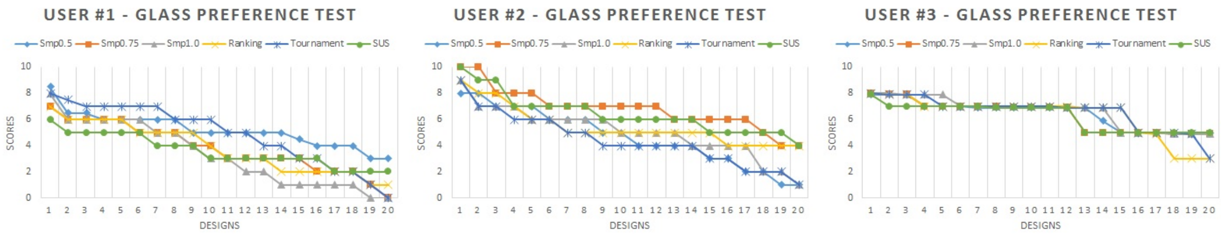

Figure 13: Results for the user preference tests on the wine glass models.

Table 8: Computational time (in seconds) for the genetic algorithm selection methods.

Test cases	**GA selection methods**						
	Smp0.5	*Smp0.75*	*Smp1.0*	*SmpExt*	*Ranking*	*Tournament*	*SUS*
Dental implant	227972.7	180569.9	15417.7	8673.6	< 1	< 1	< 1
Chassis	13730.3	129155.0	142597.1	139575.8	< 1	< 1	< 1
Test cases	**GA selection methods**						
	SmpExt				*Ranking*	*Tournament*	*SUS*
Vessel	5304.1				< 1	< 1	< 1
Spring	181.1				< 1	< 1	< 1
Beam	353.5				< 1	< 1	< 1
Train	1128.9				< 1	< 1	< 1
Test cases	**GA selection methods**						
	Smp0.5	*Smp0.75*	*Smp1.0*	*SmpExt*	*Ranking*	*Tournament*	*SUS*
Honeycomb heat sink	103144.7	91908.6	2502.1	7006.2	< 1	< 1	< 1
Wine glass - 1	302454.2	207782.0	22445.4	-	< 1	< 1	< 1
Wine glass - 2	215114.3	99522.0	1134.3	-	< 1	< 1	< 1
Wine glass - 3	256150.9	92704.3	1909.6	-	< 1	< 1	< 1

ble 8 shows the computational time for the selection techniques. The ranking, tournament, and SUS selection techniques exhibited less computational times than the proposed selection techniques did. The processing time depends heavily on the following parameters, as follows: the number n of design parameter, the number $\overline{Y}$ of designs in $\overline{P}$ (i.e., good designs), the bias intensity τ of the energies, and the number K of offsprings that will be chosen in the selection algorithm. Note that K offsprings are chosen among the $2*(n-1)*\binom{\overline{Y}}{2}$ offsprings. All energy calculations ($E_1 - E_5$) involve the number K, while the energies E_1, E_3, and E_4 contain the number n. When τ was set to 1.0, the pruning algorithm eliminated many solutions so that the algorithm converged faster. The processing time of $Smp1.0$ was less compared to those of when $Smp0.5$ and $Smp0.75$. When τ was set to 0.5, a smaller number of solutions were pruned; therefore, the computational time was higher than those of the other two τ settings. This was because the number of elements in S_P was higher (see Algorithm 1). $SmpExt$ and $Smp0.75$ produced most of the best designs in the experiments. We recommend using $SmpExt$ as a GA selection method as it had lower computational times than $Smp0.75$.

The convergence of $Smp1.0$ mostly happened when SC2 was satisfied, while SC1 was only met for the honeycomb heat sink test case for $Smp1.0$. In contrast, SC3 was mainly/intentionally satisfied for $Smp0.5$ and $Smp0.75$ as their single iteration time was high. On the other hand, $SmpExt$ was mostly converged when SC4 was met. Table 9 shows the number of iterations for the proposed GA selection algorithms in each test cases. Less numbers of iterations were observed in $Smp0.5$ due to their high computations costs so that the user set the processing time (i.e., SC3 was satisfied). Furthermore, the energy values after each iteration for the dental implant and chassis models were observed for the proposed GA selection techniques (see Figure 15). $Smp0.5$ and $Smp0.75$ converged after satisfying SC3, while $Smp1.0$ and $SmpExt$ converged when SC2 or SC4 was satisfied.

Table 9: Number of iterations for the proposed selection technique.

Test cases	**GA selection methods**			
	Smp0.5	*Smp0.75*	*Smp1.0*	*SmpExt*
Dental implant	3	8	23	12
Chassis	2	4	11	11
Test cases	**GA selection methods**			
	SmpExt			
Vessel	16			
Spring	16			
Beam	21			
Train	10			
Test cases	**GA selection methods**			
	Smp0.5	*Smp0.75*	*Smp1.0*	*SmpExt*
Honeycomb heat sink	3	9	8	19
Wine glass - 1	3	7	16	-
Wine glass - 2	4	9	23	-
Wine glass - 3	3	10	10	-

5.4. Multiple runs of the extended offspring sampling algorithm

As the extended algorithm ($SmpExt$) is greedy and starts with K random offsprings, the result may change for every algorithm run. The algorithm was executed 10 times for the vessel, spring, beam and train cases. Fig. 16 shows plots of the energy values versus number of iterations. For better visualization, there are gaps (i.e., void iterations) between consecutive iterations. The plots showed that similar energy values were

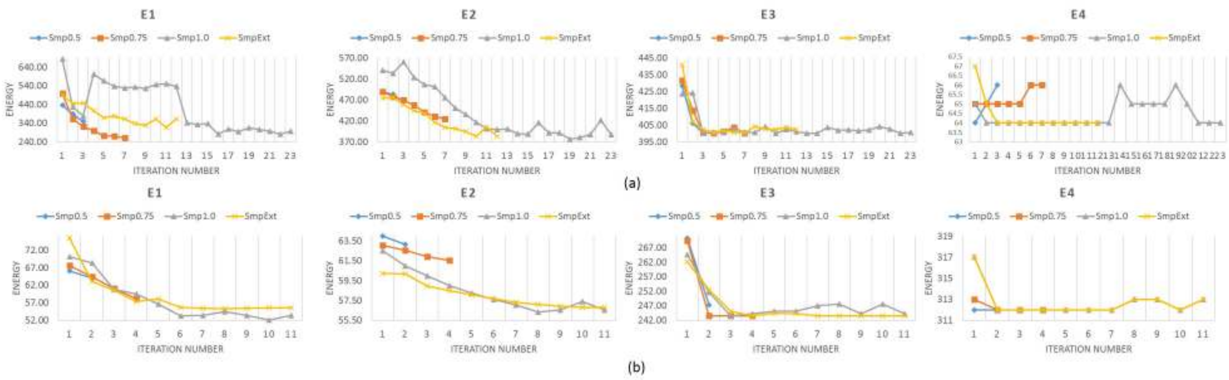

Figure 15: Energy values after the iterations for the dental implant (a) and chassis (b) models.

obtained in most algorithm runs. We have also executed $SmpExt$ 100 times for these cases and investigated the fitness values after the algorithm runs. Figures 2 and 17 show plots of (minimum) fitness values obtained versus number of iterations. We have observed that the fitness values tended to decrease or be the same at the end of iterations in the algorithm runs in most cases. This observation was valid for 393 algorithm runs in the experiments, while it was invalid only two and three runs for the vessel and train case experiments, resp. We think that the quality criteria contributed to such decrease in the fitness values. For example; the space-filling energy (E_1) strove for distributing the offsprings evenly in the design space. In this way, design space can be well scanned and designs with minimum fitness values can be found.

5.5. *Quality criteria contributions*

An ablation study has been performed using the spring and beam test cases to see the contributions of the quality criteria. The extended offspring sampling algorithm is executed without the energy $E1/E2/E3/E4$. Table 10 shows the fitness values for the best designs (i.e., designs having minimum fitness values). According to these experiments, it has been observed that the best designs obtained had higher fitness values without the first two quality criteria (i.e., space-fillingness and non-resemblance in designs with higher fitness values). Therefore, these criteria has contributed more than other two criteria in these experiments. While distributing designs (as much as) evenly by means of the space-filling criterion and placing them (as much as) far away from the designs with higher fitness values by using the second criterion, the design space can be well scanned so that desired design(s) can be obtained. However, we think that the other two criteria can also play an important role in some other test cases. Note that Design of Experiments (DOE, the fourth criterion [1]) is commonly used for the costly and time-consuming experiments, in which a limited number of designs can be tested. In any case, one can either remove or include the quality criterion/criteria based on the experiment.

6. Conclusions and future works

This paper proposed a sampling-based selection method that can be employed in crossover mating. All producible offsprings were first generated and a desired number of offsprings were chosen based on the quality criteria as follows: space-filling offsprings, offsprings non-resembling to the individuals with higher fitness values, offsprings resembling to the individuals with lower fitness values, non-collapsing offsprings, and feasible offsprings. A multi-objective optimization technique was employed based on non-dominated sorting, pruning, and selection of the solution having the minimum energy variance with the other solutions. The performance of the proposed selection method using three different application scenarios (simulation-driven product design, mechanical design and user-centered product design).

In future work, other crossover operators, such as multi-point and uniform crossover, will be integrated into the proposed selection technique. This will produce a greater number of offsprings so that better energy minimization for the quality criteria can be achieved. In addition, an interactive crossover mechanism will be studied, which will learn decision preferences in real time and reflect them in the generated offsprings. Finally, a user interface will be developed for the proposed selection algorithm to define the design parameters, their

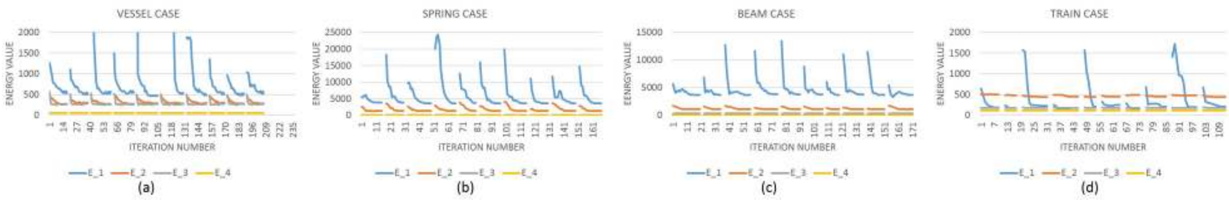

Figure 16: Energy values versus number of iterations for the vessel (a) (E_3 is scaled by multiplying with 1E-6), spring (b), beam (c), and train (d) (E_1 and E_3 is scaled by multiplying with 1E-1 and 1E-3, resp.) cases.

Table 10: The designs with minimum fitness values obtained in the ablation study while taking all or some energy terms into account.

Test cases	*E*1, *E*2, *E*3, *E*4	*E*2, *E*3, *E*4	*E*1, *E*3, *E*4	*E*1, *E*2, *E*4	*E*1, *E*2, *E*3
Spring	0.021	0.025	0.025	0.021	0.021
Beam	3.11	3.13	3.13	3.11	3.11

lower/upper bounds, and design constraints in its graphical user interface and to export the offsprings in a file.

Acknowledgements

The authors would like to thank The Scientific and Technological Research Council of Turkey for supporting this research (Project Number: 315M077), and Veysel Mert Usta and Gani Melik Onder to perform FEM tests for the dental implant models.

References

References

[1] F. Fuerle, J. Sienz, Formulation of the audze-eglais uniform latin hypercube design of experiments for constrained design spaces, Advances in Engineering Software 42 (9) (2011) 680–689.

[2] R. V. Rao, V. J. Savsani, D. P. Vakharia, Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems, Computer-Aided Design 43 (3) (2011) 303–315.

[3] K. Dorst, N. Cross, Creativity in the design process: co-evolution of problem–solution, Design studies 22 (5) (2001) 425–437.

[4] S. Krish, A practical generative design method, Computer-Aided Design 43 (1) (2011) 88–100.

[5] E. Gunpinar, U. C. Coskun, M. Ozsipahi, S. Gunpinar, A generative design and drag coefficient prediction system for sedan car side silhouettes based on computational fluid dynamics, Computer-Aided Design 111 (2019) 65–79.

[6] R. H. Kazi, T. Grossman, H. Cheong, A. Hashemi, G. Fitzmaurice, Dreamsketch: Early stage 3d design explorations with sketching and generative design, in: Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology, ACM, 2017, pp. 401–414.

[7] S. Khan, M. J. Awan, A generative design technique for exploring shape variations, Advanced Engineering Informatics 38 (2018) 712–724.

[8] J. P. Sousa, J. P. Xavier, Symmetry-based generative design and fabrication: A teaching experiment, Automation in Construction 51 (2015) 113–123.

[9] A. Runions, M. Fuhrer, B. Lane, P. Federl, A.-G. Rolland-Lagan, P. Prusinkiewicz, Modeling and visualization of leaf venation patterns, ACM Transactions on Graphics (TOG) 24 (3) (2005) 702–711.

[10] K. Shea, R. Aish, M. Gourtovaia, Towards integrated performance-driven generative design tools, Automation in Construction 14 (2) (2005) 253–264.

[11] M. Turrin, P. von Buelow, R. Stouffs, Design explorations of performance driven geometry in architectural design using parametric modeling and genetic algorithms, Advanced Engineering Informatics 25 (4) (2011) 656–675.

[12] J. J. L. Kitchley, A. Srivathsan, Generative methods and the design process: A design tool for conceptual settlement planning, Applied Soft Computing 14 (2014) 634–652.

[13] L. Caldas, Generation of energy-efficient architecture solutions applying gene_arch: An evolution-based generative design system, Advanced Engineering Informatics 22 (1) (2008) 59–70.

[14] E. Gunpinar, S. Gunpinar, A shape sampling technique via particle tracing for cad models, Graphical Models 96 (2018) 11–29.

[15] S. Khan, E. Gunpinar, Sampling cad models via an extended teaching-learning-based optimization technique, Computer-Aided Design 100 (2018) 52–67.

[16] S. Khan, E. Gunpinar, M. Moriguchi, Customer-centered design sampling for cad products using spatial simulated annealing, in: Proceedings of CAD'17, Okayama, Japan, 2017, pp. 100–103.

[17] B. Chen, Y. Pan, J. Wang, Z. Fu, Z. Zeng, Y. Zhou, Y. Zhang, Even sampling designs generation by efficient spatial simulated annealing, Mathematical and Computer Modelling 58 (3-4) (2013) 670–676.

[18] K. M. Dogan, H. Suzuki, E. Gunpinar, M. S. Kim, A generative sampling system for profile designs with shape constraints and user evaluation, Computer-Aided Design 111 (2019) 93–112.

[19] E. Kalogerakis, S. Chaudhuri, D. Koller, V. Koltun, A probabilistic model for component-based shape synthesis, ACM Transactions on Graphics (TOG) 31 (4) (2012) 55.

[20] M. Fisher, D. Ritchie, M. Savva, T. Funkhouser, P. Hanrahan, Example-based synthesis of 3d object arrangements, ACM Transactions on Graphics (TOG) 31 (6) (2012) 135.

[21] G. Stiny, Introduction to shape and shape grammars, Environment and planning B: planning and design 7 (3) (1980) 343–351.

[22] P. Prusinkiewicz, M. Shirmohammadi, F. Samavati, L-systems in geometric modeling, International Journal of Foundations of Computer Science 23 (01) (2012) 133–146.

[23] V. Granadeiro, L. Pina, J. P. Duarte, J. R. Correia, V. M. Leal, A general indirect representation for optimization of generative

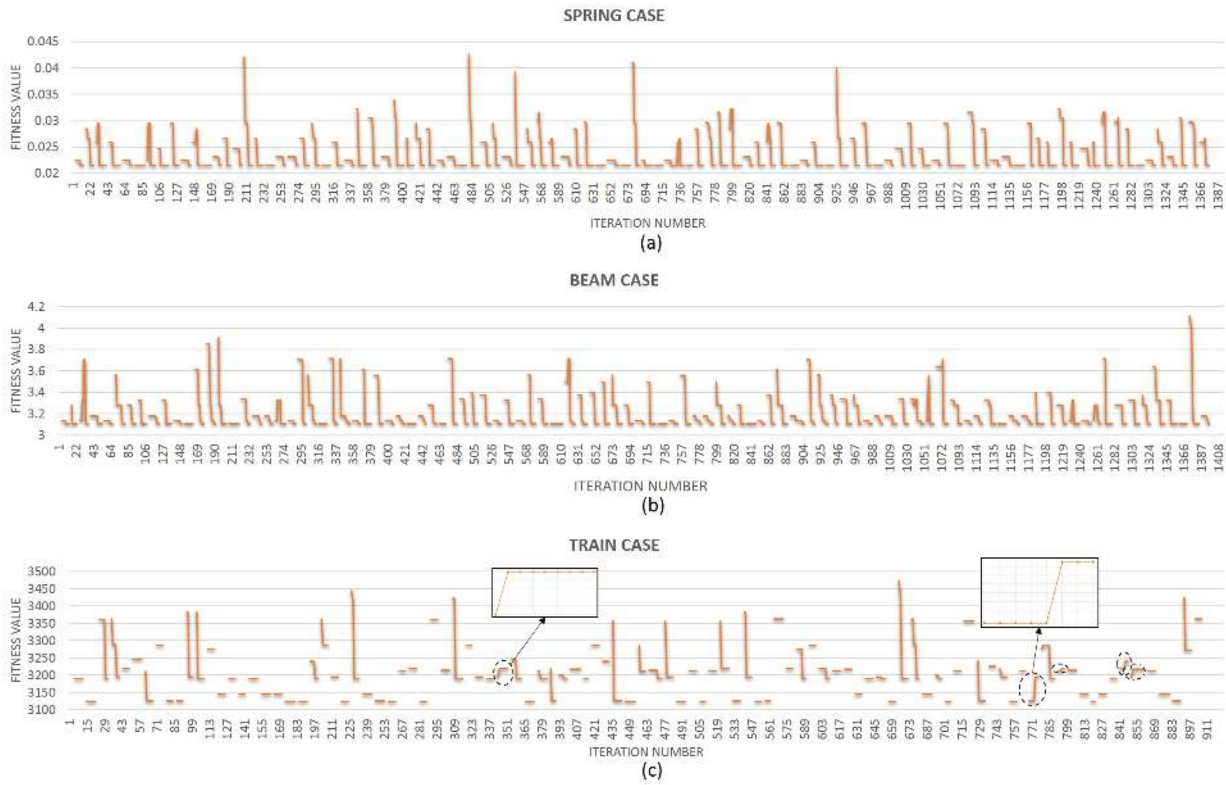

Figure 17: Plots of (minimum) fitness values obtained versus number of iterations for the spring (a), beam (b) and train (c) cases obtained using the extended offspring sampling algorithm. (Minimum) fitness values obtained mostly decrease or are same at the end of iterations in the algorithms runs except 5 algorithm runs (shown with dashed black ellipses) in the train case.

design systems by genetic algorithms: Application to a shape grammar-based design system, Automation in Construction 35 (2013) 374–382.

[24] M. C. Ang, H. H. Chau, A. Mckay, A. D. Pennington, Combining evolutionary algorithms and shape grammars to generate branded product design, in: Design Computing and Cognition, Springer, 2006, pp. 521–539.

[25] J. P. McCormack, J. Cagan, Designing inner hood panels through a shape grammar based framework, Ai Edam 16 (4) (2002) 273–290.

[26] J. Cui, M.-X. Tang, Integrating shape grammars into a generative system for zhuang ethnic embroidery design exploration, Computer-Aided Design 45 (3) (2013) 591–604.

[27] S. C. Chase, Generative design tools for novice designers: Issues for selection, Automation in Construction 14 (6) (2005) 689–698.

[28] W. Palubicki, K. Horel, S. Longay, A. Runions, B. Lane, R. Měch, P. Prusinkiewicz, Self-organizing tree models for image synthesis, ACM Transactions on Graphics (TOG) 28 (3) (2009) 58.

[29] G. Kelly, H. McCabe, Interactive generation of cities for real-time applications, in: ACM SIGGRAPH 2006 research posters, ACM, 2006, p. 44.

[30] V. Singh, N. Gu, Towards an integrated generative design framework, Design Studies 33 (2) (2012) 185–207.

[31] W. Yu, B. Li, H. Jia, M. Zhang, D. Wang, Application of multi-objective genetic algorithm to optimize energy efficiency and thermal comfort in building design, Energy and Buildings 88 (2015) 135–143.

[32] P. Subbaraj, R. Rengaraj, S. Salivahanan, Enhancement of self-adaptive real-coded genetic algorithm using taguchi method for economic dispatch problem, Applied Soft Computing 11 (1) (2011) 83–92.

[33] R. Jafari-Marandi, B. K. Smith, Fluid genetic algorithm (fga), Journal of Computational Design and Engineering 4 (2) (2017) 158–167.

[34] B. Vaissier, J.-P. Pernot, L. Chougrani, P. Véron, Genetic-algorithm based framework for lattice support structure optimization in additive manufacturing, Computer-Aided Design 110 (2019) 11–23.

[35] W. Abd-El-Wahed, A. Mousa, M. El-Shorbagy, Integrating particle swarm optimization with genetic algorithms for solving nonlinear optimization problems, Journal of Computational and Applied Mathematics 235 (5) (2011) 1446–1453.

[36] I. Ono, H. Kita, S. Kobayashi, A real-coded genetic algorithm using the unimodal normal distribution crossover, in: Advances in evolutionary computing, Springer, 2003, pp. 213–237.

[37] K. Deep, M. Thakur, A new crossover operator for real coded genetic algorithms, Applied mathematics and computation 188 (1) (2007) 895–911.

[38] E. Z. Elfeky, R. A. Sarker, D. L. Essam, Analyzing the simple ranking and selection process for constrained evolutionary optimization, journal of Computer Science and Technology 23 (1) (2008) 19–34.

[39] T. Blickle, L. Thiele, A comparison of selection schemes used in evolutionary algorithms, Evolutionary Computation 4 (4) (1996) 361–394.

[40] J. Zhong, X. Hu, J. Zhang, M. Gu, Comparison of performance between different selection strategies on simple genetic algorithms, in: Computational Intelligence for Modelling, Control and Automation, 2005 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on, Vol. 2, IEEE, 2005, pp. 1115–1121.

[41] B. A. Julstrom, It's all the same to me: Revisiting rank-based probabilities and tournaments, in: Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on, Vol. 2, IEEE, 1999, pp. 1501–1505.

[42] S. Mashohor, J. R. Evans, T. Arslan, Elitist selection schemes for genetic algorithm based printed circuit board inspection system, in: Evolutionary Computation, 2005. The 2005 IEEE Congress on, Vol. 2, IEEE, 2005, pp. 974–978.

[43] D. E. Goldberg, K. Deb, A comparative analysis of selection schemes used in genetic algorithms, in: Foundations of genetic algorithms, Vol. 1, Elsevier, 1991, pp. 69–93.

[44] K. S. Goh, A. Lim, B. Rodrigues, Sexual selection for genetic algorithms, Artificial Intelligence Review 19 (2) (2003) 123–152.

[45] S. Anand, N. Afreen, S. Yazdani, A novel and efficient selection method in genetic algorithm, International Journal of Computer Applications 129 (15) (2015) 7–12.

[46] O. Al Jadaan, L. Rajamani, C. Rao, Improved selection operator for ga., Journal of Theoretical & Applied Information Technology 4 (4).

[47] M. Affenzeller, S. Wagner, Offspring selection: A new self-adaptive selection scheme for genetic algorithms, in: Adaptive and Natural Computing Algorithms, Springer, 2005, pp. 218–221.

[48] P. Audze, V. Eglais, New approach for planning out of experiments, Problems Dynamics and Strengths 35 (1977) 104–107.

[49] J. Cai, G. Thierauf, Discrete optimization of structures using an improved penalty function method, Decision and Control 21 (4) (1993) 293–306.

[50] S. Sudeng, N. Wattanapongsakorn, Post pareto-optimal pruning algorithm for multiple objective optimization using specific extended angle dominance, Engineering Applications of Artificial Intelligence 38 (2015) 221–236.

[51] M. Cheikh, J. B., L. T., S. P., A method for selecting pareto optimal solutions in multiobjective optimization, Journal of Informatics and Mathematical Sciences 2 (1).

[52] V. M. Usta, G. M. Onder, Dental implant design for mandibular first molar tooth and material optimization with finite element analysis, Bachelor thesis, Istanbul Technical University (January 2017).

[53] M. Gen, R. Cheng, Genetic Algorithms and Engineering Optimization, John Wiley and Sons, Inc., 2007.

[54] W. S. C. Guide, Sprint car chassis (2018).
URL `http://www.world-sprintcar-guide.com/`

[55] A. Subasi, B. Sahin, I. Kaymaz, Multi-objective optimization of a honeycomb heat sink using response surface method, International Journal of Heat and Mass Transfer 101 (2016) 295–302.

[56] F. Cluzel, B. Yannou, M. Dihlmann, Using evolutionary design to interactively sketch car silhouettes and stimulate designer's creativity, Engineering Applications of Artificial Intelligence 25 (7) (2012) 1413–1424.