Engineering Optimization
Vol. 43, No. 1, January 2011, 39–59

Taylor & Francis
Taylor & Francis Group

A non-dominated sorting hybrid algorithm for multi-objective optimization of engineering problems

Hossein Ghiasi*, Damiano Pasini and Larry Lessard

Department of Mechanical Engineering, McGill University, Macdonald Engineering Building, 817 Sherbrooke West, Montreal, QC, Canada H3A 2K6

(*Received 23 June 2009; final version received 15 February 2010*)

Among numerous multi-objective optimization algorithms, the Elitist non-dominated sorting genetic algorithm (NSGA-II) is one of the most popular methods due to its simplicity, effectiveness and minimum involvement of the user. This article develops a multi-objective variation of the Nelder-Mead simplex method and combines it with NSGA-II in order to improve the quality and spread of the solutions. The proposed hybrid algorithm, called non-dominated sorting hybrid algorithm (NSHA), is compared with NSGA-II on several constrained and unconstrained test problems. The higher convergence rate and the wider spread of solutions obtained with NSHA make this algorithm a good candidate for engineering problems that require time-consuming simulation and analysis. To demonstrate this fact, NSHA is applied to the design of a carbon fibre bicycle stem simultaneously optimized for strength, weight and processing time.

Keywords: multi-objective optimization; genetic algorithm; non-dominated sorting; hybrid algorithm

1. Introduction

The majority of engineering design problems require optimizing multiple conflicting objectives. For instance, designing a part made of composite materials calls for simultaneous optimization of the structural performance and the manufacturing cost or time. These objectives are often conflicting and strongly coupled (Le Riche *et al.* 2003, Park *et al.* 2005, Ghiasi *et al.* 2008) and thus the corresponding multi-objective optimization problem does not have a single optimum solution, but a set of solutions, called 'Pareto optimum'. Pareto optimum solutions represent the trade-off among multiple objectives. In their previous works, the present authors (Ghiasi *et al.* 2008, 2009b) showed that when one or a small number of Pareto solutions are required, using a globalized form of a classical simplex method and a multi-objective optimization approach can be more efficient than an evolutionary method. In contrast, when a large number of Pareto solutions are required, an evolutionary algorithm (EA) can perform better than a classical optimization method.

EAs are simple, robust and independent of gradient information, and use a population based approach that allows generating multiple solutions in a single run. EAs have received significant attention and a variety of these algorithms has been developed to solve multi-objective problems;

*Corresponding author. Email: hossein.ghiasi@mail.mcgill.ca

ISSN 0305-215X print/ISSN 1029-0273 online

DOI: 10.1080/03052151003739598
http://www.informaworld.com

examples include Pareto-archived evolution strategy (PAES; Sayin and Karabati 1999), strength Pareto-evolutionary algorithm (SPEA2; Zitzler *et al.* 2001) and multi-objective differential evolution (MODE; Babu and Anbarasu 2005). An overview of the earlier works using EAs for multi-objective optimization can be found in Fonseca and Fleming (1995) and in Coello (1999, 2002), who provided a comprehensive survey and a critical review of multi-objective EAs.

Among numerous multi-objective EAs, some of which are listed above, genetic algorithms (GAs) have attracted the most attention (Coello 1999) because of their ability in supporting parallel computation and addressing discontinuous, non-differentiable and non-convex functions with multiple peaks. The potential of GAs in multi-objective optimization was initially hinted by Rosenberg in the 1960s and later by Goldberg (1989); however, this area remained unexplored until recently. A comprehensive review of GA-based methods is outside the scope of this article due to the large number and diversity of the applications. The following paragraphs include only a brief review of the most popular multi-objective GAs.

Assigning a single measure of fitness to each individual is the main challenge in using a GA for multi-objective optimization. Hajela and Lin's genetic algorithm (HLGA; Hajela and Lin 1992) used the method of weighted-sum to perform this task. The complexity of determining appropriate weights was the main drawback of this method (Bingul 2006). The vector evaluated genetic algorithm (VEGA; Schaffer 1984) avoids this problem by creating a number of sub-populations and performing the selection according to each objective function, in turn. In this method no weighting coefficient is required, but the population may tend to split into different species, each particularly good in one of the objectives.

The multi-objective genetic algorithm (MOGA) by Fonseca and Fleming (1993) and its real-coded variation by Purshouse and Fleming (2001) used a rank-based fitness assignment procedure to assign a single fitness to each individual. In this method, each individual is ranked by the number of its dominating individuals. The Niched Pareto genetic algorithm (NPGA; Horn *et al.* 1994) is another example that uses a Pareto domination tournament. The tournament selection includes picking two (or more) candidate solutions at random and comparing them with a random comparison set and with each other. The candidate selected for reproduction is the one that is not dominated by the comparison set or the one that is better than the other candidate. Fonseca and Fleming's multi-objective genetic algorithm (FFGA; Fonseca and Fleming 1993) used niche formation methods with a modified fitness assignment that permitted the intervention of an external decision maker. Non-dominated sorting genetic algorithm (NSGA; Srinivas and Deb 1994) and its improved form NSGA-II (Deb *et al.* 2002a) are other examples of GA-based methods using Pareto optimality to rank individuals within the population. This algorithm is used in this article and is explained in the following sections.

More information about VEGA, MOGA, NPGA, NSGA and their comparative strengths and weaknesses can be found in Coello (2002) and Zitzler *et al.* (2000). By testing several functions, Zitzler *et al.* (2000) showed that NSGA could find the closest solutions to the Pareto front among other GA-based multi-objective optimization methods. The other methods ranked from best to worst are: VEGA, HLGA, NPGA and FFGA. In this article, an improved form of NSGA, called NSGA-II, is hybridized with a multi-objective adaptation of the Nelder-Mead simplex method (NM). The proposed hybrid method, called the non-dominated sorting hybrid algorithm (NSHA), maintains all the features of NSGA-II, while it improves the convergence rate and the scope of solutions.

The remainder of this article is organized as follows. First, the NSGA-II is explained and the improvements proposed in the literature are reviewed. Most of these modifications can be equally applied to the proposed hybrid algorithm. The next section describes our proposed hybrid method called NSHA. Section 4 compares NSHA with NSGA-II on some test problems. After demonstrating the superiority of NSHA over NSGA-II, the hybrid algorithm is applied to a composite design problem in Section 5. Finally, Section 6 concludes the article.

2. Non-dominated sorting GA

NSGA-II, proposed by Deb *et al.* (2002a), has been demonstrated to be one of the most efficient and popular algorithms for multi-objective optimization. This section briefly describes NSGA-II and its advantages and shortcomings and also the modifications and improvements suggested in the literature to improve its performance.

2.1. *NSGA-II*

NSGA-II (Deb *et al.* 2002a) is a multi-objective evolutionary algorithm that uses non-dominated sorting and a crowded-comparison approach to find a set of evenly distributed solutions to a multi-objective optimization problem. NSGA-II was proposed to reduce computational complexity, improve diversity of the solutions and add elitism to its ancestor NSGA by Srinivas and Deb (1994).

The optimization process in NSGA-II starts with a random population of solutions (or individuals) sorted by non-dominated sorting approach. In this sorting procedure, all non-dominated solutions are ranked 1 and are temporarily removed from the population. The next set of non-dominated solutions in the population is then defined and ranked 2. The procedure is continued until all the solutions are ranked. The population of parents is generated by applying a binary tournament selection to the current population. The binary tournament selection randomly picks two solutions from the current population and selects the better one with respect to the non-domination rank. Solutions at the same non-domination front are compared by a crowding distance, which is a measure of the density of the solutions at the neighbourhood of that solution. Next, a population of off-springs is generated by applying the genetic operators (*i.e.* recombination and mutation) to the population of parents. The next population is formed by taking the best solutions from the combined population of parents and off-springs. The selection criteria are first the non-domination rank and then the crowding distance. The procedure is terminated when a user-defined maximum number of generations is reached.

Simplicity, effectiveness, modularity and low number of user-defined parameters, are the main factors determining the popularity of NSGA-II among multi-objective optimization methods (Deb 2008); however, NSGA-II has also a few shortcomings. For instance, NSGA-II usually requires a large number of generations to approach the Pareto front. The population initially moves fast towards the real Pareto front but slows down further into the process and finally approaches the Pareto front only asymptotically (Lahanas *et al.* 2003). In addition to the low convergence rate, NSGA-II may not be efficient in generating a Pareto set that covers the entire true Pareto front within a reasonably low number of generations (Gao *et al.* 2008). The deficiency is caused by the limitation of the crossover and mutation operators for sufficiently intensifying the search.

Deb (2008) provided a functional decomposition of NSGA-II into three main operations: (i) elitism that supports fast and reliable convergence towards better solutions, (ii) non-domination sorting that accentuates non-dominated solutions and assures a progress towards the Pareto front and (iii) crowding distance that put emphasis on less crowded solutions to maintain the diversity of the solutions. In order to overcome the above shortcomings of this algorithm, various extensions of NSGA-II are proposed based on the rational of altering at least one of its operators.

2.1.1. *NSGA-II with modified genetic operators*

Deb *et al.* (2002b) suggested an improvement in convergence of NSGA-II by using more than two parents to create one descendant. Iorio and Li (2004) replaced the real-coded crossover and mutation with a differential evolution scheme that used the difference between solutions to

perturb the population. Another modified operator was the jumping gene operator, entailing the replacement of a randomly selected part of the chromosome with a new randomly generated set of binary numbers (Agarwal and Gupta 2008, Kasat and Gupta 2003). To improve the diversity of the solutions, Yijie and Gongzhang (2008) forced the crossover operator to perform more likely on genes located far from each other within the design space. Murugana *et al.* (2009) recommended a controlled elitist and a virtual mapping procedure for the same purpose. Methods with modified operators have been tested on a number of test problems, where, in most cases, mixed improved convergence rate and spread of solutions were reported. Not a noticeable improvement in convergence and spread was reported for a general test problem.

Not only the genetic operators, but also the structure of the chromosome and the population has been altered to improve the performance of NSGA-II. Maneeratana *et al.* (2005) and Praveen Kumar *et al.* (2007) proposed a co-evolution of multiple species by splitting the population into a number of sub-populations or species that share a gene similarity. Hierarchical genotype encoding proposed by Kumar *et al.* (2009) is another modification. Tran (2005) suggested simultaneously running multiple populations with different population sizes, in order to automatically select the population size. In some cases, an improvement was reported in the quality of the solutions; however, additional computation due to working with multiple populations is the main drawback of these methods.

2.1.2. *NSGA-II with modified non-dominated sorting*

Altering the non-dominated sorting procedure can also affect the progress towards the Pareto-optimal front. Examples of this approach include ε-MOEA, a proper domination, fuzzy domination (Deb 2008) and quick sort (Zheng *et al.* 2004). The aims of these improvements are to reduce the time to converge to the Pareto front and to reduce the computational complexity, but the subsequent penalty is that part of the real Pareto front may be excluded.

2.1.3. *NSGA-II with modified crowding distance*

The diversity preservation procedure may also be altered in order to achieve a better distribution of solutions or to emphasise part of the Pareto front. Clustered NSGA-II (Deb 2008) claimed to find a better distribution of points by replacing the crowding distance operator with a K-mean clustering approach. Li *et al.* (2008) tried to achieve the same goal by replacing the crowding distance operator with an algorithm based on minimum spanning tree (MST). In another attempt, the selection procedure was modified to accept a point to be in the new population only if its distance to all current points in the new population is greater than a user-defined value (Ghomsheh *et al.*, 2007). Other similar attempts are reviewed by Deb (2008), including: projection-based diversity preservation, niching, omni-optimizer, extreme-point preference, and other methods. Although the modified algorithm can find a better distribution of the solutions, these methods generally require a longer computational time than the original NSGA-II.

2.2. *NSGA-II in hybrid algorithms*

One promising approach to increase the convergence rate and the solution diversity is to hybridize NSGA-II with a local search. The local search operator replaces or follows the mutation operator and helps to intensify the search in various areas pointed by the genetic mechanisms. This type of hybrid algorithm, called a memetic algorithm (MAs; Moscato 1989), has been shown to be more efficient than a genetic algorithm (Hart and Belew 1996). A review of some of these hybrid

methods can be found in El-Mihoub *et al.* (2006). This section provides a summary of the hybrid multi-objective optimization methods that uses NSGA-II as the global optimizer.

Hu *et al.* (2003) integrated a sequential quadratic programming (SQP) algorithm into NSGA-II. In this hybrid form, the SQP optimized a randomly chosen objective of one of the current solutions, while the remaining objectives were restricted to be less than or equal to their current value. It was judged that the hybrid form is successful regarding the convergence rate, and not deteriorating in terms of diversity of the solutions. Another hybrid form was proposed by Kumar *et al.* (2007), who resorted to SQP in order to locally improve one objective in the set of non-dominated solutions. A mixed improvement in performance was reported considering the convergence and diversity of the solutions. Gao *et al.* (2008) also proposed a hybrid form, where NSGA-II and SQP run almost independently, but with some exchange of information. The proposed hybrid form was found more efficient than NSGA-II in terms of convergence, particularly at the earlier steps.

SQP requires calculating the function gradient and the optimum step length at every iteration, an operation that can be costly for a practical engineering problem. To reduce the computational cost, Hernandez-Diaz *et al.* (2008) proposed using the gradient information only at the beginning of the search. The steepest descent method was adapted to generate a number of non-dominated points which formed the initial population for NSGA-II. Lahanas *et al.* (2003) used a similar approach but with a different local search called L-BFGS. The last two methods required considerably less computational time but presented a lower convergence rate than the hybrid forms using the local search during the optimization process.

One way to avoid time-consuming calculation of gradient information is to opt for a direct local search method. An example of hybrid methods with direct local search is PHC-NSGA-II (Bechikh *et al.* 2008) that uses Pareto hill climbing (PHC) as a local search. A higher convergence rate was reported compared to the original NSGA-II; however, not a significant improvement in the diversity of the solutions was achieved. Another similar hybrid form is S-MOGLS by Ishibuchi and Narukawa (2004), who used the r-Opt algorithm as a local search. The r-Opt is a heuristic optimization method that improves the current solution by sequentially replacing one, two or three adjacent genes in the chromosome. Xu *et al.* (2008) also used a similar hybrid form, where the r-Opt was run only during the initialization process. The local search methods used in these algorithms are not efficient and can be improved.

The simplex method by Nelder and Mead (1965) is one of the most common direct search methods and has been shown to be effective in producing a rapid initial improvement in the objective function values (Lagarias *et al.* 1998). Numerous hybrid forms of this simplex method with genetic algorithm have been proposed for single-objective optimization problems (*e.g.* Yen *et al.* 1998, Chelouah and Siarry 2003, Hongfeng *et al.* 2009). However, its application to multi-objective optimization problems is less studied because it requires aggregating multiple objectives into one single objective. Koduru *et al.* (2005) used the concept of fuzzy dominance to assign a single measure of fitness to each individual. In their proposed hybrid algorithm, called FSGA, K-means clustering was used to break up the population into closely spaced clusters. Some sufficiently populated clusters were chosen, from which a simplex was randomly selected for the application of the NM method. The major difficulty here is the computational time required for the clustering procedure. The second problem is the shape of the initial simplex, which may be poorly scaled, because the simplex is randomly selected. Finally, in this method NM is called after every generation of NAGA-II. Since the difference between the two subsequent populations is usually very small, calling the local search every iteration may not lead to a significant improvement.

Martinez and Coello (2008) alleviated the last problem by performing the local search only after a certain number of generations. In their proposed method, called NSS-GA, solutions with the best value for each objective were selected to be locally improved with respect to that objective. To avoid

the tendency toward the anchor points, an aggregating objective that minimizes a weighted sum of all the objectives was used. Although effective, the use of one aggregating objective is not able to provide an even distribution of solutions on the Pareto front. In addition, since the efficiency of the NM method strongly depends on the number of design variables (Han and Neumann 2006) this method is not efficient for problems with a large number of design variables.

The hybrid form presented in this article also uses NM method as the local search due to its good convergence rate and simplicity of the algorithm. NM method typically requires only one or two function evaluations per iteration (except in shrink, which is rare in practice), while many other direct search methods that use a finite-difference approximation of the function gradient, such as the derivative free conjugate directions method, model-based methods, implicit filtering, etc. (Nocedal and Wright 2006), require either $O(n)$ or $O(n^2)$ function evaluations per iteration. This figure is very important in applications where the function evaluation is expensive or time-consuming. Pattern search methods, such as coordinate search (Nocedal and Wright 2006), Hooke and Jeeves method (Hooke and Jeeves 1961) or method of Rosenbrock (Nocedal and Wright 2006), also require more function evaluations than the NM method, because of the line search that must be performed at each iteration.

The next section proposes a hybrid algorithm that improves the performance of NSGA-II and does not suffer from the shortcomings of the previous hybrid algorithms. The main features and benefits of the proposed hybrid algorithm include:

(1) the use of a multi-objective form of NM method as a local search,
(2) not having a tendency toward a certain region(s) of the Pareto front,
(3) taking a full benefit of the local search by its proper initialization,
(4) maintaining the efficiency of the local search even for high-dimensional problems,
(5) maintaining the simplicity of NSGA-II,
(6) not requiring additional user-defined parameters that need prior insight into the problem, and finally,
(7) using the same building blocks as NSGA-II and preserving its modular aspect.

3. Non-dominated sorting hybrid algorithm (NSHA)

The hybrid algorithm proposed in this article integrates the Nelder-Mead (NM) simplex method into NSGA-II in order to improve the quality of the solutions and to accelerate the advancement of the non-dominated front toward the true Pareto front. The next subsection describes the main optimization loop of NSHA.

3.1. *Main optimization loop*

As shown in Figure 1, NSHA starts with a randomly generated population with a user-defined size. Using this initial population, a few generations of NSGA-II are carried on. The local search is activated only after all the individuals of the current population are located at the first non-domination front. When activated, the local search operates only on a subset of α_l% of the current population. To increase the efficiency of the local search, only a randomly selected subset of three to five design variables are used for the local optimization and other variables are kept constant at their current value. For each selected solution, the local search generates an initial simplex, a regular hyper-polygon with a predefined size, a, where the selected solution is located at one of its vertices. The local optimization process is terminated when either a user-defined maximum number of function evaluations, $nf_{l,\max}$, is reached or any other stopping criterion indicating convergence to a local optimum is satisfied. The selected solutions in the current population are

n = number of design variables,
$nf_{l,\max}, nf_{g,\max}, nf_t$ = maximum number of function evaluations for local search, genetic algorithm, and total
a = size of the initial hyper-polygon for the local search,
n_{pop} = population size for genetic algorithm,
n_l = maximum number of points within the population that can be improved by the local search,
randomly initialize the first population, P_0
$k = 0$
while there is more than one level of non-domination in P_k
    $P_{k+1} = NSGA-II(P_k)$ improve current population with NSGA–II
    $k = k + 1$
end
nf = number of function evaluations performed by NSGA-II
while $nf < nf_t$
    $j = 1$
    $nf_{l,\max} = \min\{nf_{l,\max}, nf_t - nf\}$
    while $(j < n_l) \,\&\, (P_k(j)\_rank = 1) \,\&\, (nf < nf_t)$
        if $n > 5$, randomly select a subset of three to five variables to be optimized by the local search
        create a hyper-polygon simplex S with the size a based on $P_k(j)$
        $nf_l = 0$
        while $(nf_l < nf_{l,\max}) \,\&\, (no\ other\ stopping\ criteria\ for\ NSNM\ is\ met)$
            Improve S using NSNM
            $nf_l = nf_l + 1$
        end
        $nf = nf + nf_l$
        replace $P_k(j)$ with the best point in S
        set $j = j + 1$
    end
    set $nf_{g,\max} = \min\{nf_{g,\max}, nf_t - nf\}$
    $nf_g = 0$
    while $nf_g < nf_{g,\max}$
        $P_{k+1} = NSGA-II(P_k)$ improve current population with NSGA-II
        $k = k + 1$
        $nf_g = nf_g +$ number of function evaluations performed by NSGA-II
    end
    $nf = nf + nf_g$
end

Figure 1. Pseudo code of the main loop in NSHA.

then replaced by the improved solutions found by the local search, creating a locally improved population.

The locally improved population is used as an initial population for the next few generations by NSGA-II. The number of generations performed by NSGA-II before calling the local search, $nf_{g,\max}$, is defined by the user at the onset of the process. The values for $nf_{l,\max}$ and $nf_{g,\max}$ specify the share of NSGA-II and NSNM in the optimization process. $nf_{g,\max}$ should be selected sufficiently large to give NSGA-II the chance to improve the population before the next local search is being called. On the other hand, $nf_{l,\max}$ must be selected large enough to allow convergence of the local search. The main optimization loop is repeated until either the total maximum number of function evaluations, nf_t, or any other user-defined stopping criterion is reached.

S_0 = a given initial simplex
$nf_{l,\max}$ = maximum number of function evaluations for local search
$a_{\min}$ = minimum simplex size
$S = S_0$
$k = 0$
while (there is at least one solution in S dominating another solution in S) and ($k < nf_{l,\max}$) and (simplex size < $a_{\min}$)
    Improve *S* using NM optimization (Nelder and Mead 1965) using the sorting procedure as below:
        sort selected solutions using the non-dominate sorting (Deb *et al*. 2002a)
        maintain the order of the points if more than one point in each domination level
    end
    $k = k +$ number of function evaluations used during NM optimization
end

Figure 2. Pseudo code of the local simplex optimizer NSNM.

3.2. *Local search algorithm (NSNM)*

The local optimization method used in our hybrid algorithm is called non-dominated sorting Nelder-Mead (NSNM) method. The flowchart in Figure 2 shows NSNM algorithm, which is similar to Nelder-Mead simplex method but using the non-domination rank as the objective to be minimized. The crowding distance may not be used to rank the solutions located at the same non-domination front, thus these points are ranked by preserving their order in the original simplex. This method of ranking was tested on several random initial populations and was found to be efficient in reaching a diverse set of solutions on the Pareto front. Finally, the local optimization algorithm is terminated if one of the following stopping criteria is met:

(i) all of the points within the simplex are located at the same non-domination front,
(ii) the simplex size becomes smaller than a pre-defined value,
(iii) a user-defined maximum number of function evaluations is reached.

3.3. *Constraint handling method*

NSNM used the concept of constrained domination (Deb 2008) to handle a general non-linear constraint expressed as an inequality that must be greater than or equal to zero. In this ranking scheme, all infeasible solutions assume a non-domination rank higher than the last feasible solution. A crowding distance is assigned to the feasible solutions, while a value equal to the sum of all violated constraints is assigned to the infeasible solutions. Since the constraints are formulated as inequalities that must be positive or zero, the sum of the value of the violated constraints is a negative value that shows the extent of constraint violation. Solutions lying at a certain non-domination front are sorted in descending order by the crowding distance if the solutions are feasible or by the extent of the constraint violation if the solutions are infeasible. A more elaborate algorithm that also resorts to non-domination ranking for infeasible solutions is possible; however, Deb *et al.* (2002a) showed that the procedure explained in this paragraph is generally more effective.

4. Mathematical test problems

In this section, NSHA is applied to a set of constrained and unconstrained test problems from the literature and its performance is compared with the real coded NSGA-II, considering two

performance metrics introduced by Deb *et al.* (2002a). The first metric, γ, measures the extent of the convergence to a known set of Pareto-optimal solutions, while the second metric, Δ, provides information about the extent of spread achieved by the solutions. Both metrics are positive real numbers with zero representing the best performance. This section describes the test problems and compares the results obtained by NSGA-II and NSHA.

4.1. *Test problems*

Table 1 shows the list of unconstrained problems used to compare the performance of NSHA to NSGA-II. The first six test problems have two objectives and a known, continuous Pareto front. MOP4 has a discontinuous 2-dimensional (2D) Pareto front, whose closed-form expression is not available to the authors. The Pareto front of DTZL1 is 3-dimensional (3D) and the performance metrics may not be properly calculated for this test problem. Therefore, for MOP4 and DTZL1, only a visual comparison of the results is discussed.

Table 2 shows the list of selected constrained test problems. The first test problem is chosen due to its continuous 2D Pareto front, which enables using the performance metrics for a precise comparison of the two optimization algorithms. The second test problem is more complex because it requires consideration of six design variables and six constraints. The theoretical Pareto optimal front of this test problem consists of several discontinuous broken-lines.

For all test cases, a constant population of randomly generated 100 individuals is used and the optimization process is continued for up to 25,000 function evaluations, except for DTZL1, for which 100,000 function evaluations are performed. The maximum number of points participating in the local improvement process is 20% of the population size. The size of the initial simplex for the local optimization algorithm is set to 10% of the smallest edge of the hyper-cube surrounding the design domain, while the minimum simplex size at which the local search is terminated is chosen to be 0.1% of this value. The maximum number of function evaluations for the local search, $nf_{l,\max}$, and for the NSGA-II, $nf_{g,\max}$, are respectively limited to 100 and 2000 function evaluations. These choices provide almost equal contribution of the local search (*i.e.* $20 \times 100 = 2000$ function evaluations) and NSGA-II (*i.e.* half of the total number of function evaluations is used by NSGA-II and the other half by NSNM). Due to the stochastic nature of the algorithm, each test problem is solved several times and the averaged performance metrics are compared.

4.2. *Impact of the selective use of design variables*

The first point demonstrated here is that using a selected subset of the design variables for the local search is more efficient than using the entire of set of the design variables. Among several test problems, ZDT1 is chosen for this purpose. Three different scenarios are studied. The first scenario uses the NSGA-II. The second scenario, shown by NSHA-All, uses NSHA where the entire set of design variables is involved in the local search. The third scenario, shown by NSHA, uses a random subset of three to five design variables during the local optimization process.

Figure 3 shows that NSHA achieved a better convergence (lower γ) than the two other methods. When all 30 design variables of the ZDT1 are used during the local search (NSHA-All), the convergence rate of the local search is very slow, thus the local search is unable to locally improve the solutions and the function evaluations performed during the local search are ineffective.

4.3. *Unconstrained test problems*

This section compares the performance of NSHA with NSGA-II when applied on some unconstrained test problems shown in Table 1. For each test function the performance metrics are

Table 1. Unconstrained test problems used to compare the performance of NSHA and NSGA-II.

Problem	n	$x_i \in$	Objective functions	Optimal solution	Reference
FON	3	$[-4, 4]$	$f_1 = 1 - \exp\left(-\sum_{i=1}^{3}\left(x_i - \frac{1}{\sqrt{3}}\right)^2\right)$ $f_2 = 1 - \exp\left(-\sum_{i=1}^{3}\left(x_i + \frac{1}{\sqrt{3}}\right)^2\right)$	$x_1 = x_2 = x_3 \in \left[-\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}\right]$	Deb *et al.* (2002a)
ZDT1	30	$[0, 1]$	$f_1 = x_1$ $f_2 = g(x)[1 - \sqrt{x_1/g(x)}]$ $g(x) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i$	$x_1 \in [0, 1]$ $x_i = 0,$ $i = 2, \ldots, n$	Deb *et al.* (2002a)
ZDT2	30	$[0, 1]$	$f_1 = x_1$ $f_2 = g(x)[1 - (x_1/g(x))^2]$ $g(x) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i$	$x_1 \in [0, 1]$ $x_i = 0,$ $i = 2, \ldots, n$	Deb *et al.* (2002a)
ZDT3	30	$[0, 1]$	$f_1 = x_1$ $f_2 = g(x)\left[1 - \sqrt{x_1/g(x)} - \frac{x_1}{g(x)}\sin(10\pi x_1)\right]$ $g(x) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i$	$x_1 \in [0, 1]$ $x_i = 0,$ $i = 2, \ldots, n$	Deb *et al.* (2002a)
ZDT4	10	$x_1 \in [0, 1],$ $x_i \in [-5, 5]$ $i = 2, \ldots, n$	$f_1 = x_1$ $f_2 = g(x)[1 - \sqrt{x_1/g(x)}]$ $g(x) = 1 + 10(n-1) + \sum_{i=2}^{n}(x_i^2 - 10\cos(4\pi x_i))$	$x_1 \in [0, 1]$ $x_i = 0,$ $i = 2, \ldots, n$	Deb *et al.* (2002a)

Table 1. (Continued)

Problem	n	$x_i \in$	Objective functions	Optimal solution	Reference
ZDT6	10	[0, 1]	$f_1 = x_1$ $f_2 = g(x)[1 - (f_1(x)/g(x))^2]$ $g(x) = 1 + 9\left[\sum_{i=2}^{n} x_i/(n-1)\right]^{0.25}$	$x_1 \in [0, 1]$ $x_i = 0,$ $i = 2, \ldots, n$	Deb *et al.* (2002a)
MOP4	3	$[-5, 5]$	$f_1 = \sum_{i=1}^{n-1}\left(-10\exp\left(-0.2\sqrt{x_i^2 + x_{i+1}^2}\right)\right)$ $f_2 = \sum_{i=1}^{n}\left(\lvert x_i\rvert^{0.8} + 5\sin(x_i)^3\right)$	A piecewise curve	Yijie and Gongzhang (2008)
DTZL1	7	[0, 1]	$f_1 = \frac{1}{2}x_1x_2g(x)$ $f_2 = \frac{1}{2}x_1(1-x_2)g(x)$ $f_3 = \frac{1}{2}(1-x_1)g(x)$ $g(x) = 1 + 100\left[5 + \sum_{i=3}^{7}[(x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5))\right]$	A linear optimal front	Deb *et al.* (2001)

Table 2. Constrained test problems used to compare the performance of NSHA and NSGA–II.

Problem	n	$x_i \in$	Objective functions	Constraints	Reference
CONSTR	2	$x_1 \in [0.1, 1]$ $x_2 \in [0, 5]$	$f_1 = x_1$ $f_2 = (1 + x_2)/x_1$	$g_1(x) = 9x_1 + x_2 - 6 \geqslant 0$ $g_2(x) = 9x_1 - x_2 - 1 \geqslant 0$	Deb *et al.* (2002a)
COK	6	$x_1, x_2, x_6 \in [0, 5]$ $x_3, x_5 \in [1, 5]$ $x_4 \in [0, 6]$	$f_1(x) = -25(x_1 - 2)^2 - (x_2 - 2)^2 - (x_3 - 1)^2$ $-(x_4 - 4)^2 - (x_5 - 1)^2$ $f_2(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2$	$g_1(x) = x_1 + x_2 - 2 \geqslant 0$ $g_2(x) = 6 - x_1 - x_2 \geqslant 0$ $g_3(x) = 2 + x_1 - x_2 \geqslant 0$ $g_4(x) = 2 - x_1 + 3x_2 \geqslant 0$ $g_5(x) = 4 - (x_3 - 3)^2 - x_4 \geqslant 0$ $g_6(x) = (x_5 - 3)^2 + x_6 - 4 \geqslant 0$	Yijie and Gongzhang (2008)

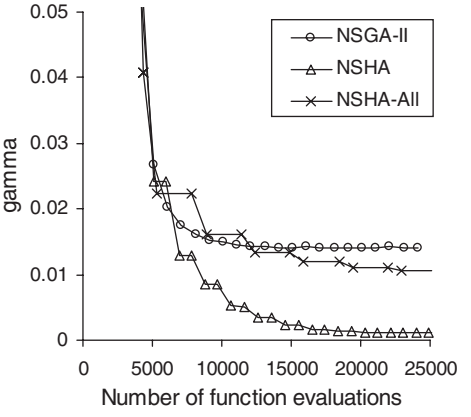

Figure 3. Convergence measure, γ, versus number of function evaluations performed when NSGA-II, NSHA, and NSHA. All are applied to the ZDT1 test problem. The results are averaged over 10 trials with random initial populations.

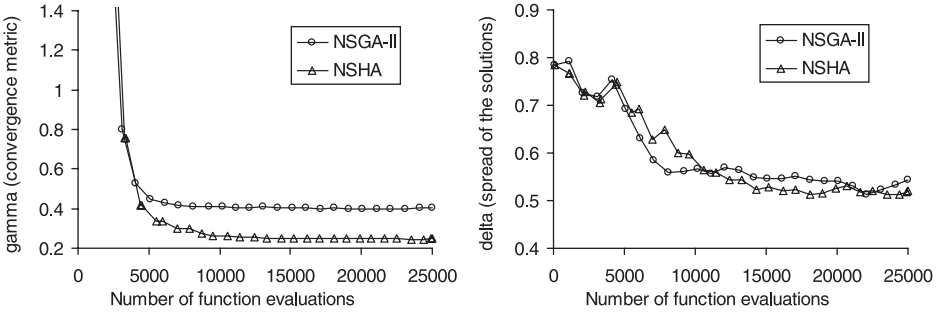

Figure 4. Average performance parameters γ and Δ for FON, ZDT1, 2, 3, 4, and 6 averaged over 10 trials for each test function.

calculated and averaged over ten trials with different random initial points. The results for the first six test functions (*i.e.* FON, ZDT1, 2, 3, 4 and 6), which all have two objectives and a continuous known Pareto front, are averaged and presented in Figure 4. The individual presentation of the results is omitted for the sake of brevity but it can be found in Ghiasi (2009).

Figure 4 shows that convergence of the two algorithms is generally the same at the very beginning of the optimization process. However, at later stages where the local search is activated, the solutions found by NSHA are closer to the real Pareto front than the one obtained by NSGA-II (showed by lower values for the convergence metric). The other promising point observed in the convergence properties of the two algorithms is that NSGA-II stopped progressing toward the Pareto front after 10,000 function evaluations, while NSHA continued the progress for a longer period, and even after 23,000 function evaluations, a small move toward the Pareto front was observed.

The individual assessment of the convergence metrics for these test functions shows that for simple and smooth test functions such as FON and ZDT6 (*i.e.* this function lacks the trigonometric term that exist in other ZDT test functions, thus the function is relatively smooth) the performance of the two algorithms is similar all along the process and both reach a solution set very close to the real Pareto front. However, for more complicated and noisy test functions, such as ZDT1, 2, 3 and 4, NSHA evidently finds solutions closer to the Pareto front and continues to converge toward the real Pareto front far beyond where the NSGA-II stops progressing.

Regarding the spread of the solutions, measured by Δ and shown in Figure 4, the two algorithms achieved the same overall spread of the solutions, because the crowding-comparison operator,

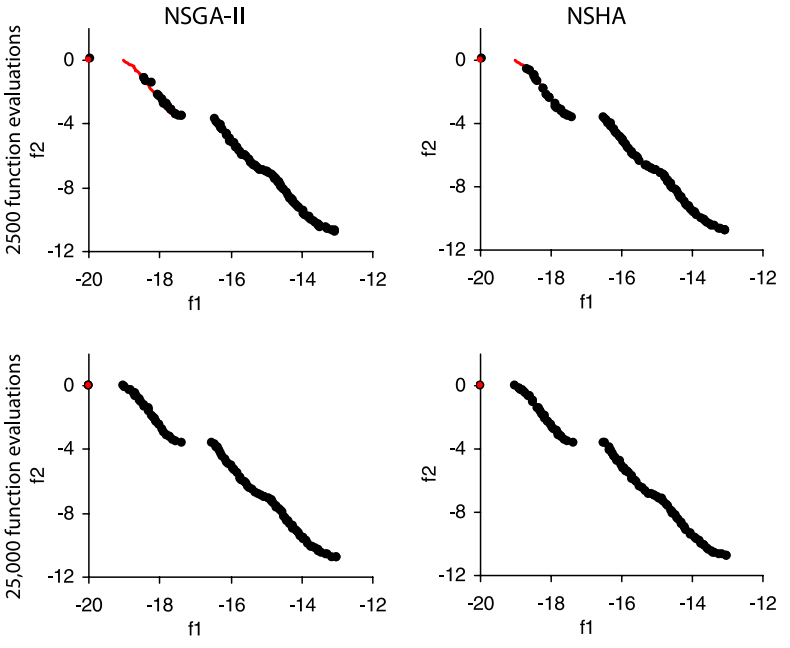

Figure 5. Non-dominated solutions obtained by NSGA-II and NSHA for MOP4 test problem, the solid line shows the real Pareto front.

which is responsible for achieving an even spread of solutions, is similar in both algorithms. Since during the local search no crowding distance is calculated, the distribution of the solutions may get worse, but it is improved as soon as a few generations of NSGA-II are performed. To avoid a poor spread of the solutions in the final set of solutions, the recommendation is to terminate the optimization process with a few generations of NSGA-II. The visual comparison of the solutions confirms that the two algorithms also performed similarly regarding the scope of the solutions, with a slightly better performance for NSHA.

MOP4 is used to compare the performance of the two algorithms on problems with a piecewise Pareto front. Since the performance metrics are not suited for problems with piecewise Pareto fronts, the results are visually compared in Figure 5. The non-dominated solutions obtained by the two algorithms are illustrated after 2500 and 25,000 function evaluations. It shows that both algorithms achieved almost the same quality of the solutions (spread and closeness to the real Pareto front) after 25,000 function evaluations; however, NSHA arrived to this solution faster than NSGA-II. The faster convergence is confirmed when the solutions found after 2500 function evaluations are compared; the solutions of NSHA are closer to the real Pareto front and cover a wider portion of the Pareto front compared to the solutions of NSGA-II.

The last unconstrained test problem is DTZL1, which has three objectives. The set of solutions obtained by the two algorithms after 100,000 function evaluations is illustrated in Figure 6, which shows that the solutions obtained by NSHA are dominating the ones obtained by NSGA-II (*i.e.* are closer to the origin). Generally, both algorithms performed poorly, as previously reported also by Deb (2008), who suggested improving the solutions by using ε-domination in NSGA-II. Since NSHA uses the same non-domination sorting algorithm, this modification can also be applied to NSHA to improve the distribution of the solutions for this test problem.

4.4. *Constrained test problems*

The constrained test problems shown in Table 2 are solved in this section. The first test problem is CONSTR, for which the two algorithms performed similarly, although the performance metrics

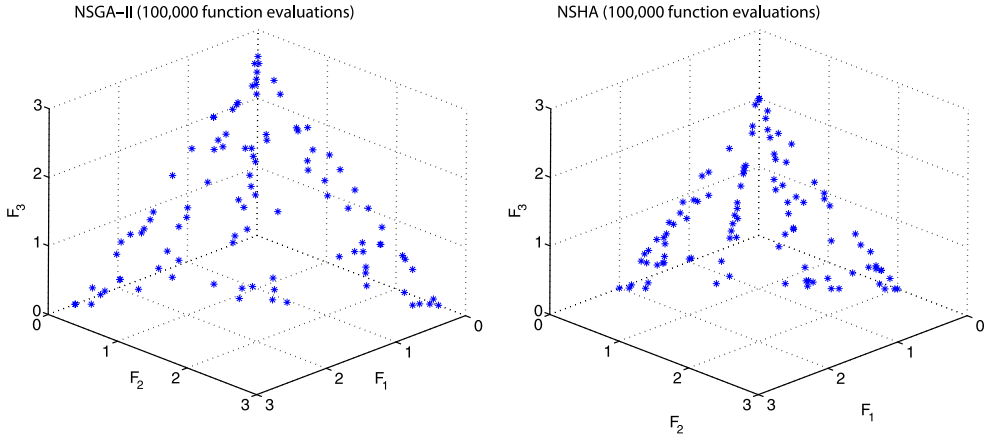

Figure 6. Non-dominated solutions obtained by NSGA-II (left) and NSHA (right) for DTZL1 test function after performing 100,000 function evaluations.

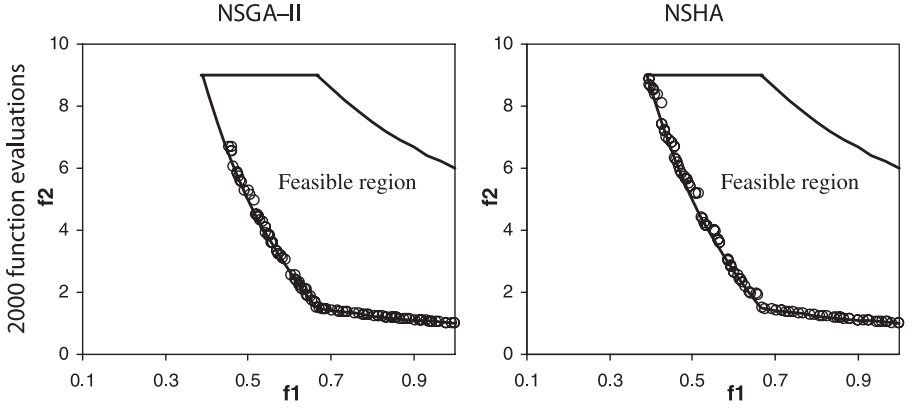

Figure 7. Solutions found by NSGA-II and NSHA for CONSTR after 2000 function evaluations.

were less stable for NSHA. As shown in Figure 7, the significant difference between the two algorithms is in the scope of the solutions. For small number of function evaluations (*e.g.* less than 5000), NSHA performed better than NSGA-II and provided a wider scope of solutions; however, the solutions obtained by the two algorithms after 15,000 (not shown here) are similar and cover the entire Pareto front of the problem.

The second constrained test problem, COK, also has a discontinuous piecewise Pareto front and includes six design variables and six constraints. Since the closed-form representation of the Pareto front is not available, the solutions obtained after 200,000 function evaluations are visually compared in Figure 8. This figure shows that the solutions obtained by NSHA cover almost the entire Pareto front, whereas NSGA-II could find only half the Pareto front.

4.5. *Sensitivity to the user-defined parameters*

NSHA requires a few parameters to be determined by the user, including the frequency of calling the local search and the number of individuals participating in the local search. In this section the ZDT1 test problem is used to study the effect of these user-defined parameters on the performance of NSHA.

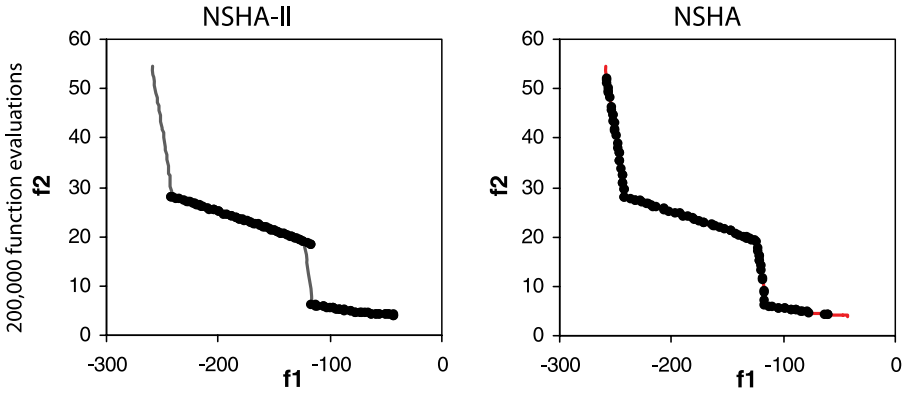

Figure 8. Progress of the solutions found by NSGA-II and NSHA toward the Pareto front of COK.

To examine the sensitivity of the convergence rate to these parameters, ZDT1 is solved by NSHA with the local search being called every 5, 10 or 20 generations and the numbers of solutions participating in the local search varied between 10, 20, 30 and 40% of the current population. The results showed that the user-defined parameters have a minor effect on the convergence of NSHA. For instance, calling the local search every 5 or 20 generations could change the convergence metric only by 3%. Similarly, changing the percentage of the population participating in the local search from 10% to 40% yielded a difference less than 15% in the convergence metric. The detailed results for this sensitivity analysis can be found in Ghiasi (2009). Considering the low sensitivity of the performance measures to these parameters, assigning a reasonable value to these two parameters is not a difficult task and does not require *a priori* insight into the problem. For a general problem, the authors suggest calling the local search every 10 generations and having the local search performed on 20% of the population.

5. Design of a composite part

Design and manufacturing of a composite part is used in this section as an example to demonstrate the ability of the proposed hybrid algorithm in solving a practical engineering problem. Composite materials are well-known for having many design variables, which gives the designer more opportunities to tailor the material for a given application. However, it requires solving a more complex design problem. Different optimization algorithms have been used to solve the corresponding design optimization problem (Ghiasi *et al.* 2009a). In this section, the application of NSHA in solving a multi-objective design problem, which takes into account both manufacturing and structural parameters, is demonstrated.

5.1. *Composite test problem*

The part studied in this section is a carbon fibre bicycle stem, the component of the bicycle that connects the handlebar to the fork, as shown in Figure 9. Bladder assisted resin transfer moulding (RTM) is used as the manufacturing method, because of the small size of the part, its complicated geometry with hollow sections, the high level of surface finish and fast production cycle required. In this manufacturing method, first, a dry pre-form consisting of four layers of continuous braided carbon fibre sleeves is placed around an inflatable bladder and is positioned inside a solid airtight mould. After inflating the bladder, an injection pump is used to push the

Figure 9. Stem is part of a bicycle that connects the handlebar to the fork.

resin through the fibres which are compressed between the bladder and the mould surface. The air exits through the strategically placed vents as the resin front advances. The part is ready to be de-moulded when the cure procedure is completed. This manufacturing process results in a strong interconnection between the structural and the manufacturing parameters; therefore, a multi-objective optimization method is required to solve the corresponding design problem.

5.2. *Optimization problem*

The corresponding optimization problem consists in finding the minimum weight and mould filling time and the maximum strength or minimum inverse Tsai-Wu strength ratio. The design variables consist of four braid diameters, injection pressure and bladder pressure. The optimization problem is formulated as follows:

$$\min_{d_i,\, P_{inj},\, P_{bladder}} \begin{cases} W(d_i,\, P_{inj},\, P_{bladder}) \\ T(d_i,\, P_{inj},\, P_{bladder}); \qquad i = 1, \ldots, 4 \\ S(d_i,\, P_{inj},\, P_{bladder}) \end{cases}$$

$$s.t. \begin{cases} 38.1mm < d_i < 63.5mm; \quad i = 1, \ldots, 4 \\ 100kPa < P_{inj} < 450kPa \\ 150kPa < P_{bladder} < 500kPa \\ P_{bladder} - P_{inj} \geqslant 50kPa \end{cases} \tag{1}$$

where W, T and S in this equation represent weight, filling time and the inverse Tsai-Wu strength ratio, respectively, d_i is the braid diameter, and p_{inj} and $P_{bladder}$ stand for injection and bladder pressures. The flowchart in Figure 10 shows the interconnection between the design variables and the objectives and the method to calculate each parameter. The function evaluation process consists of a structural analysis and a flow simulation. A finite element analysis in ANSYS® is employed for structural analysis, while the flow simulation is performed using a MATLAB® code developed by the first author. The function evaluation process for this problem is time consuming and takes about 100 seconds on a computer with Intel-Pentium4, 3.2GHZ, 2GB RAM. Therefore, the number of function evaluations must be kept as low as possible. As a result, the higher convergence rate of the hybrid optimization algorithm introduced in this article is an important asset for situations similar to this test problem.

5.3. *Results and discussion*

NSHA is applied to the stem design problem using a population of 40 individuals. The optimization process is terminated after 5000 function evaluations. Figure 11 shows the final solutions in the 3D criterion space and in the pair-wise 2D spaces. This figure gives insight into the problem by

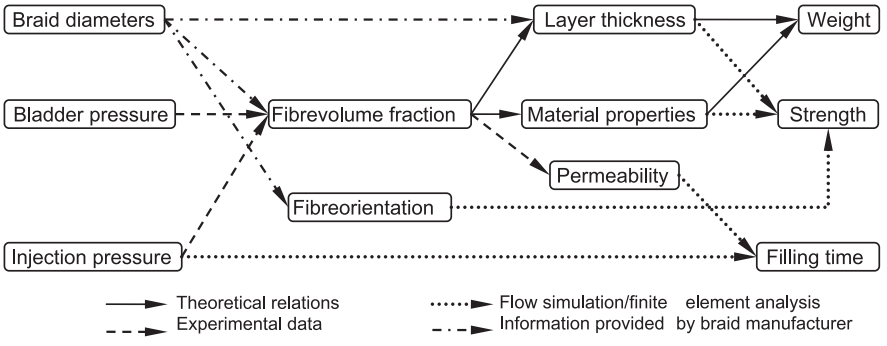

Figure 10. Interconnection among design variables, intermediate parameters and the objectives of the stem design problem.

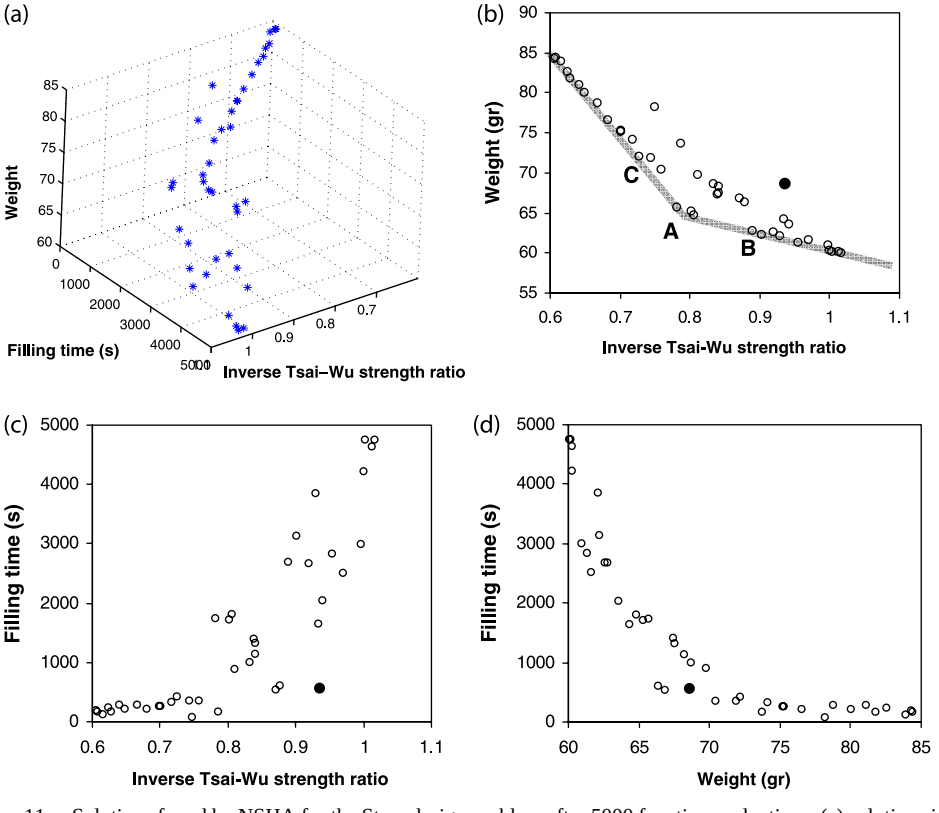

Figure 11. Solutions found by NSHA for the Stem design problem after 5000 function evaluations, (a) solutions in 3D criterion space (b)–(d) pair wise 2D plots of the solutions.

presenting the achievable values for the objectives. For instance, part (c) in this figure shows that there is no trade-off between strength and filling time, thus one may find a solution that has the minimum production time and the maximum strength at the same time; however, simultaneously achieving the maximum strength and minimum weight (Figure 11(b)) or minimum weight and minimum filling time (Figure 11(d)) is impossible. In addition to the presence and significant effect of the coupling terms, this figure also provides the numerical values for the achievable objectives. In addition, part (b) of this figure shows a knee point, marked by letter (A). This knee

point shows that decreasing the inverse Tsai-Wu strength ratio from 1.1 to 0.78 is associated with a smaller penalty in weight than improving the strength of the part beyond this value.

The solid circle shown in Figure 11 part (b) to (d) is the experimental design found in our laboratory after months of trial and error (Thouin 2004). This point is very close to the Pareto front regarding the weight and the filling time (part (d) in this figure); however, this point is not close to the Pareto front regarding the trade-off between weight and strength. The results found in this research can be used to find a solution better than the one found by experiments, while saving the large amount of time usually spent on designing a part by experiment.

The distribution of the solutions within the design space, which shows the values for design parameters, is also studied and a tendency toward the solution with the best strength is observed among the results. This tendency shows that the optimum design is dominated by the structural parameters and is closer to the best structural design than to the best design for manufacturing. Note that the proximity to the structural design is with respect to the design variables; therefore, it is not in conflict with the reported strong coupling in performance. The domination by structural design was also previously reported by Henderson *et al.* (1999) for a blade-stiffened composite panel.

To summarize the results of the practical design problem, the following three points are noted:

(1) The trade-off between structural performance and manufacturing concerns can be captured and quantified using this multi-objective optimization method.
(2) The solutions obtained by this method can help distinguishing the feasible combination of performances and finding the critical designs.
(3) A coupled design with a reasonable combination of structural and manufacturing performances can be achieved by making only small changes in the best structural design.

6. Conclusions

An efficient multi-objective optimization algorithm is required to solve multi-objective design problems. Evolutionary algorithms are a popular tool for this purpose, because of their ability in achieving more than one solution in a single run. Among these methods, NSGA-II has received a significant attention and popularity; however, like other evolutionary algorithms, it is slow in convergence and usually needs several function evaluations before a reasonably good set of solutions is reached.

In this article, NSGA-II was hybridized with a local search algorithm based on the Nelder-Mead (NM) simplex method to improve its convergence rate and quality of the solutions. In order to make NM capable of handling multi-objective problems, we used the non-dominated sorting algorithm used in NSGA-II. The constrained-domination was used to handle the inequality constraints in constrained optimization problems.

The proposed algorithm called non-dominated sorting hybrid algorithm (NSHA) was compared with NSGA-II on eight unconstrained and two constrained test problems from the literature. The performance of the two algorithms was compared using two performance metrics measuring the convergence to the real Pareto front and the spread of the solutions. Using several test problems, it was demonstrated that the hybrid algorithm significantly increases the convergence rate and the extent of the solutions. It was observed that to obtain the same quality of the results, NSHA requires a smaller number of function evaluations than NSGA-II. The local search integrated into the algorithm significantly intensifies the ability of the algorithm in searching the design space and reaching critical solutions, such as knee points and discontinuities in the Pareto front. Therefore, the hybrid algorithm was shown to be able to obtain a wider portion of the Pareto front, even when irregularities exist in the Pareto front of the problem. Since the diversity preservation mechanism

is only applied within NSGA-II, the similar spread of solutions was observed for both NSHA and NSGA-II.

To demonstrate the capability in handling practical engineering problems, NSHA was applied to the optimum design of a composite bicycle stem. Finite element analysis and resin flow simulation were required to evaluate the three objectives of this problem, which made the function evaluation process time consuming. The optimum solutions found by NSHA reveal the trade-off among conflicting objectives, the extreme values for each objective, and the critical design points. The results were found close to the solutions obtained by experiment and trial and error.

By intensifying the search and reducing the computational time, NSHA provides an effective tool which benefits from the robustness of a genetic algorithm and the high convergence rate of the Nelder-Mead method. This optimization tool is suitable for solving complex time-consuming multi-objective optimization problems. One of the major advantages of NSHA is that it maintains the simplicity and modular aspect of NSGA-II, while achieving a wider scope and higher quality solutions. Most modifications proposed in the literature to adapt NSGA-II to special applications, can be applied to NSHA to further improve its performance for a particular application.

References

Agarwal, A. and Gupta, S.K., 2008. Jumping gene adaptations of NSGA-II and their use in the multi-objective optimal design of shell and tube heat exchangers. *Chemical Engineering Research and Design*, 86, 123–139.

Babu, B.V. and Anbarasu, B., 2005. Multi-objective differential evolution (MODE): An evolutionary algorithm for multi-objective optimization problems. *Proceedings of the 3rd international conference on computational intelligence, robotics, and autonomous systems (CIRAS-2005)*, 13–16 December, Singapore. Available from http://discovery.bits.pilanri.ac.in/discipline/chemical/bvb/ciras_2005_mode_final.pdf [Accessed 24 April 2010].

Bechikh, S., Belgasmi, N., Said, L.B. and Ghédira, K., 2008. PHC-NSGA-II: A novel multi-objective memetic algorithm for continuous optimization. *20th IEEE international conference on tools with artificial intelligence (ICTAI 2008)*, 3–5 November, Dayton, Ohio, USA. Washington D.C.: IEEE Computer Society, 180–189.

Bingul, Z., 2006. Adaptive genetic algorithms applied to dynamic multi-objective problems. *Applied Soft Computing*, 7, 791–799.

Chelouah, R. and Siarry, P., 2003. Genetic and Nelder-Mead algorithm hybridized for a more accurate global optimization of continuous multi-minima functions. *European Journal of Operation Research*, 148, 335–348.

Coello, C.A., 1999. A comprehensive survey of evolutionary-based multi-objective optimization techniques. *International Journal of Knowledge and Information Systems*, 1 (3), 269–308.

Coello, C.A., 2002. Evolutionary multi-objective optimization: a critical review. *In:* M. Ehrgott and X. Gandibleux, eds. *Multiple criteria optimization, state of the art*, Annotated Bibliographic Surveys. Boston: Kluwer Academic Publishers, 117–146.

Deb, K., 2001. *Multiobjective optimization using evolutionary algorithms*. Chichester, UK: John Wiley and Sons Ltd.

Deb, K., 2008. A robust evolutionary framework for multi-objective Optimization. *Proceedings of the 10th annual conference on genetic and evolutionary computation* (*GECCO'08*), 12–16 July, Atlanta, Georgia, USA. New York: ACM Press, 633–640.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T., 2002a. Fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6 (2), 182–197.

Deb, K., Anand, A. and Joshi, D., 2002b. A computationally efficient evolutionary algorithm for real–parameter optimization. *Evolutionary Computation Journal*, 10 (4), 371–395.

Fonseca, C.M. and Fleming, P.J., 1993. Genetic algorithms for multi-objective optimisation: formulation, discussion and generalisation. *In:* S. Forrest, ed. *Proceedings of the 5th international conference on genetic algorithms*, June, Urbana, Illinois. San Francisco, CA: Morgan Kaufmann Publishers Inc., 416–423.

Fonseca, C.M. and Fleming, P.J., 1995. An overview of evolutionary algorithms in multi-objective optimization. *Journal of Evolutionary Computation*, 3, 1–16.

Gao, X., Chen, B., He, X., Qiu, T., Li, J., Wang, C. and Zhang, L., 2008. Multi-objective optimization for the periodic operation of the naphtha pyrolysis process using a new parallel hybrid algorithm combining NSGA-II with SQP. *Computers and Chemical Engineering*, 32, 2801–2811.

Ghiasi, H., 2009. *Multi-objective optimization of composite parts made by resin transfer moulding*. Thesis (PhD). Department of Mechanical Engineering, McGill University.

Ghiasi, H., Pasini, D. and Lessard, L., 2008. Constrained globalized Nelder-Mead method for simultaneous structural and manufacturing optimization of a composite bracket. *Composite Materials*, 42 (7), 717–736.

Ghiasi, H., Pasini, D. and Lessard, L., 2009a. Optimum stacking sequence design of composite materials part I: constant stiffness design. *Composite Structures*, 90 (1), 1–11.

Ghiasi, H., Pasini, D. and Lessard, L., 2009b. Pareto frontier for simultaneous structural and manufacturing optimization of a composite part. *Structural and Multidisciplinary Optimization*, 40, 497–511.

Ghomsheh, V., Khanehsar, M.A. and Teshnehlab, M., 2007. Improving the non-dominate sorting genetic algorithm for multi-objective optimization. *International conference on computational intelligence and security workshops* (*CISW 2007*), 15–19 December, Heilongjiang, China. Washington, D.C.: IEEE Computer Society, 89–92.

Hajela, P. and Lin, C.Y., 1992. Genetic search strategies in multicriterion optimal design. *Structural Optimization*, 4, 99–107.

Han, L. and Neumann, M., 2006. Effect of dimensionality on the Nelder-Mead simplex method. *Optimization Methods and Software*, 21 (1), 1–16.

Hart, W. and Belew, R., 1996. Optimization with genetic algorithm hybrids that use local search. In: *Adaptive individuals in evolving populations: Models and algorithms*. Santa Fe Institute Studies in the Science of Complexity, Vol. 26. Reading, MA: Addison-Wesley, 483–496.

Henderson, J.L., Gürdal, Z. and Loos, A.C., 1999. Combined structural and manufacturing optimization of stiffened composite panels. *Journal of Aircraft*, 36 (1), 246–254.

Hernandez-Diaz, A.G., Coello, C.A., Perez, F., Caballero, R., Molina, J. and Santana-Quintero, L.V., 2008. Seeding the initial population of a multi-objective evolutionary algorithm using gradient-based information. *2008 IEEE congress on evolutionary computation* (*CEC 2008*), 1–6 June, Hong Kong, China. Washington, D.C.: IEEE Computer Society, 1617–1624.

Hongfeng, X., Guanzheng, T. and Jingui, H., Large scale function optimization or high-dimension function optimization in large using simplex-based genetic algorithm. *Proceedings of the 1st ACM/SIGEVO summit on genetic and evolutionary computation, 2009* (*GEC'09*), 12–14 June, Shanghai, China. New York: ACM Press, 209–216.

Hooke, R. and Jeeves, T.A., 1961. Direct search solution of numerical and statistical problems. *Journal of the Association for Computing Machinery*, 8, 212–229.

Horn, J., Nafpliotis, N. and Goldberg, D.E., 1994. A niched Pareto genetic algorithm for multi-objective optimization. *Proceedings of the 1st IEEE conference on evolutionary computation*, 27–29 June, Orlando, Florida, USA. Piscataway, NJ: IEEE Neural Network Council, 82–87.

Hu, X., Huang, Z. and Wang, Z., 2003. Hybridization of the multi-objective evolutionary algorithms and the gradient-based algorithms. *Proceedings of the IEEE Congress on Evolutionary Computation*, 8–12 December, Canberra, Australia. New York: IEEE Press, 870–877.

Iorio, A.W. and Li, X., 2004. Solving rotated multi-objective optimization problems using differential evolution. *Proceedings of AI 2004: Advances in artificial intelligence*, 4–6 December, Cairns, Australia Berlin: Springer-Verlag (LNAI 3339), 861–872.

Ishibuchi, H. and Narukawa, K., 2004. Performance evaluation of simple multi-objective genetic local search algorithms on multi-objective 0/1 Knapsack problems. *Congress on evolutionary computation*, *CEC2004*, 19–23 June, Portland, OR. New York: IEEE Press, 441–448.

Kasat, R.B. and Gupta, S.K., 2003. Multi-objective optimization of an industrial fluidized-bed catalytic cracking unit (FCCU) using genetic algorithm with the jumping gene operator. *Computers and Chemical Engineering*, 27 (12), 1785–1800.

Koduru, P., Das, S., Welch, S., Roe, J., and Lopez-Dee, Z.P., 2005. A co-evolutionary hybrid algorithm for multi-objective optimization of gene regulatory network models. *2005 Genetic and Evolutionary Computation Conference (GECCO'05)*, 25–29 June, Washington, DC. New York: ACM Press, 393–399.

Kumar, A., Sharma, D. and Deb, K., 2007. A hybrid multi-objective optimization procedure using PCX based NSGA-II and sequential quadratic programming. *Special session & competition on performance assessment of multi-objective optimization algorithms* (*CEC-07*), 25–28 September, Singapore. New York: IEEE Press, 3011–3018.

Kumar, R., Izui, K., Yoshimura, M. and Nishiwaki, S., 2009. Multi-objective hierarchical genetic algorithms for multi level redundancy allocation optimization. *Reliability Engineering and System Safety*, 94, 891–904.

Lagarias, J.C., Reeds, J.A., Wright, M.H. and Wright, P.E., 1998. Convergence properties of the Nelder-Mead simplex method in low dimensions. *SIAM Journal of Optimization*, 9 (1), 112–147.

Lahanas, M., Baltas, D. and Zamboglou, N., 2003. A hybrid evolutionary algorithm for multi–objective anatomy-based dose optimization in high-dose-rate brachytherapy. *Physics in Medicine and Biology*, 48, 399–415.

Le Riche, R., Saouab, A. and Breard, J., 2003. Coupled compression RTM and composite layup optimization. *Composite Science and Technology*, 63 (15), 2277–2287.

Li, M., Zheng, J. and Wu, J., 2008. *Improving NSGA-II algorithm based on minimum spanning tree*. Lecture Notes in Computer Science. Berlin/Heidelberg: Springer-Verlag (LNCS 5361), 170–179.

Maneeratana, K., Boonlong, K. and Chaiyaratana, N., 2005. Co-operative co-evolutionary genetic algorithms for multi-objective topology design. *Computer-Aided Design & Applications*, 2 (1–4), 487–496.

Martinez, S.Z. and Coello, C.A., 2008. A proposal to hybridize multi-objective evolutionary algorithms with non-gradient mathematical programming techniques. *In:* G. Rudolph *et al*., eds. *Lecture Notes in Computer Science*. Berlin/Heidelberg: Springer-Verlag (PPSN X, LNCS 5199), 837–846.

Moscato, P., 1989. On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. *Caltech concurrent computation program*. C3P Report 826.

Murugan, P., Kannan, S. and Baskar, S., 2009. NSGA-II algorithm for multi-objective generation expansion planning problem. *Electric Power Systems Research*, 79, 622–628.

Nelder, J.A. and Mead, R., 1965. Simplex method for function minimization. *Computer Journal*, 7 (4), 308–313.

Nocedal, J. and Wright, S.J., 2006. *Numerical optimization*. Springer Series in Operations Research. 2nd edn. New York: Springer Science+Business Media, LLC.

Park, C.H., Lee, W.I., Han, W.S. and Vautrin, A., 2005. Multiconstraint optimization of composite structures manufactured by resin transfer moulding process. *Composite Material*, 39 (4), 347–374.

Praveen Kumar, K., Sharath, S., D'Souza, G.R. and Chandra, S.K., 2007. Memetic NSGA – a multi-objective genetic algorithm for classification of microarray data. *Proceedings of 15th international conference on advanced computing and communications* (*ADCOM 2007*), 18–21 December, Guwahati, India. Los Alamitos, CA: IEEE Computer Society, 75–80.

Purshouse, R.C. and Fleming, P.J., 2001. *The multi-objective genetic algorithm applied to benchmark problems – an analysis*. Research Report No. 796, Department of Automatic Control and Systems Engineering University of Sheffield, Sheffield, S1 3JD, UK.

Sayin, S. and Karabati, S., 1999. A bicriteria approach to the two-machine flow shop scheduling problem. *European Journal of Operational Research*, 113 (2), 435–449.

Schaffer, J.D., 1984. *Some experiments in machine learning using vector evaluated genetic algorithms*. Thesis (PhD). Vanderbilt University, Nashville, TN, USA.

Srinivas, N. and Deb, K., 1994. Multi-objective function optimization using non-dominated sorting genetic algorithms. *Evolutionary Computation*, 2 (3), 221–248.

Thouin M., 2004. Design of a carbon fiber bicycle stem using an internal bladder and resin transfer molding. Thesis (MEng). McGill University, QC, Canada.

Tran, K.D., 2005. Elitist non-dominated sorting GA-II (NSGA-II) as a parameter-less multi-objective GA. *IEEE SoutheastCon 2005*, 8–10 April, Fort Lauderdale, Florida, USA. New York: IEEE Press, 359–367.

Xu, H., Fan, W., Wei, T. and Yu, L., 2008. An or-opt NSGA-II algorithm for multi-objective vehicle routing problem with time windows. *4th IEEE conference on automation science and engineering*, 23–26 August, Key Bridge Marriott, Washington DC, USA. New York: IEEE Press, 309–314.

Yen J., Liao, J.C., Lee, B. and Randolph, D., 1998. A hybrid approach to modeling metabolic systems using a genetic algorithm and simplex method. *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics*, 28 (2), 173–191.

Yijie, S. and Gongzhang, S., 2008. Improved NSGA-II multi-objective genetic algorithm based on hybridization-encouraged mechanism. *Chinese Journal of Aeronautics*, 21, 540–549.

Zheng, J., Ling, C., Shi, Z., Xue, J. and Li, X., 2004. A multi-objective genetic algorithm based on quick sort. *In:* A.Y. Tawfik and S.D. Goodwin, eds. *Canadian AI 2004*, 17–19 May, London, Ontario, Canada. Berlin/Heidelberg: Springer-Verlag (LNAI 3060), 175–186.

Zitzler, E., Deb, K. and Thiele, L., 2000. Comparison of multi-objective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8 (2), 173–195.

Zitzler, E., Deb, K., Thiele, L., Coello, A.C. and Corne, D., 2001. *Proceedings of the 1st international conference on evolutionary multi-criterion optimization* (*EMO 2001*). Lecture Notes in Computer Science (LNCS 1993). Heidelberg: Springer.