Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia CIRP 00 (2014) 000–000

www.elsevier.com/locate/procedia

Variety Management in Manufacturing. Proceedings of the 47th CIRP Conference on Manufacturing Systems

# Increased Robustness of Product Sequencing Using Multi-Objective Optimization

Anna Syberfeldt*[a], Patrik Gustavsson [a]

*University of Skövde, PO Box 408, SE-54148 Skövde, Sweden*

* Corresponding author. Tel.: +46500448577. *E-mail address:* anna.syberfeldt@his.se

**Abstract**

Almost all manufacturing processes are subject to uncontrollable variations, caused, for example, by human operators or worn-out machines. When optimizing real-world product sequencing problems, it is of importance to find solutions that are robust, that is, whose performance remains relatively unchanged when exposed to uncertain conditions. In this paper, an extension of the traditional method of handling variations through replications is suggested that aims at finding solutions with an increased degree of robustness. The basic idea is to use standard deviation as an additional optimization objective and transform the single-objective problem into a multi-objective problem. Using standard deviation as an additional objective aims to focus the optimization on solutions that exhibit both high performance and high robustness (that is, having low standard deviation). In order to optimize the two objectives simultaneously, a multi-objective evolutionary algorithm is utilized. The proposed method for improved robustness is evaluated using a real-world test case found at the company GKN Aerospace in Sweden. GKN Aerospace manufactures a variety of different components for aircraft engines and aero derivative gas turbines. The company has recently installed a new workshop, and the focus of the study is on the x-ray stations in this workshop. For performing optimizations the company has created a simulation model that realistically mimics the workshop. As an optimization technique, a multi-objective evolutionary algorithm called NSGA-II is being used. The algorithm considers the mean value and standard deviation from replications of the stochastic simulation as objectives, optimizing both of them simultaneously. Results from the study show that the optimization is able to successfully find robust solutions using the proposed method. However, the general increase in algorithm performance expected with the proposed method is absent, and possible reasons for this are discussed in the paper.

*Keywords:* Product Sequencing, Robustness, Multi-objective optimization, Evolutionary Algorithm

## 1. Introduction

In a manufacturing workshop the need of sequencing rises when several products compete for the capacity of the same machine. Sequencing means that products queuing in front of a machine are assigned priorities that indicate their internal order of processing. Since the sequencing of products significantly influences the performance of the manufacturing process, it is of interest to optimize. Sequencing optimization is, however, non-trivial due to the NP-hard complexity of the problem, which means that the time required computing an optimal solution increases exponentially with the number of products [1]. NP-hard problems are known to be associated with a high computational cost since finding an optimal solution requires an exhaustive search. Explicitly evaluating all combinations of priorities to find the optimal sequencing is virtually impossible, especially in real-world problems. For obtaining results within a reasonable time period, the optimization must be undertaken using an inexact technique that is able to efficiently explore the search space and find near-optimal solutions. One such technique that has been proven to successfully optimize complex sequencing problems is evolutionary algorithms [2-3]. Evolutionary algorithms are population-based metaheuristics utilizing a global search strategy influenced by biological theories of genetics and reproduction (for further information about evolutionary algorithms, see [4]).

Since almost all manufacturing processes are subject to uncontrollable variation (caused, for example, by human operators, worn-out machines or a fluctuating demand), it is important that the evolutionary algorithm is able to find robust solutions when applied to real-world sequencing problems [5-7]. A robust solution is one whose performance remains relatively unchanged when exposed to uncertain conditions, as illustrated in Figure 1.

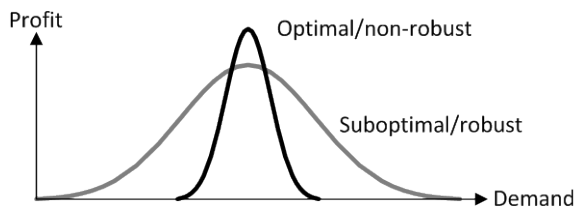

Fig. 1. Illustration of robustness.

The common technique for dealing with uncertainty is to use the average value obtained from repeated evaluations of a solution [5]. Evaluating a solution $n$ times reduces the uncertainty by a factor of $\sqrt{n}$, however, at the expense of an n times higher computational cost. Due to the linear increase in computational cost, the evaluation of a solution can usually only be replicated a few times, and consequently the uncertainty is not eliminated – only reduced. In this paper an extension of the traditional method of handling uncertainty through replications is suggested that aims at finding solutions with an increased degree of robustness. The basic idea is to use standard deviation as an additional optimization objective and transform the single-objective problem into a multi-objective problem. Using standard deviation as an additional objective aims to focus the optimization on solutions that exhibit both high performance and high robustness (that is, having low standard deviation).

In order to optimize the two objectives simultaneously, a multi-objective evolutionary algorithm is utilized. Instead of only seeking a single optimum, multi-objective algorithms maintain a set of Pareto-optimal solutions that represent the best trade-off between the optimization objectives. Evolutionary algorithms are well suited for the Pareto-based approach as they, contrary to many other optimization techniques, can capture multiple trade-off solutions in a single optimization run since they maintain a population of solutions.

The next section continues with a deepened discussion of the hypothesis that a multi-objective evolutionary algorithm can efficiently find near-optimal, robust solutions to product sequencing problems by considering the mean value and standard deviation as objectives. In Section 3 the optimization algorithm NSGA-II is described. Section 4 presents an evaluation of the proposed method on a real-world problem of product sequencing found at the company GKN Aerospace. The results from the study are discussed in Section 5, while Section 6 summarizes the study and outlines future work.

## 2. Robust product sequencing

In a product sequencing problem, variable perturbations mean that the order of which products are processed in the machines cannot be fully predicted, but may vary from time to time although the product priorities remain the same.

To ensure a sequencing of products that results in a high performance of the manufacturing system although variations are present, robust solutions must be sought after. Robust solutions mean that small changes in the production, e.g. late products, machine breakdowns, etc. will not affect the outcome significantly. Finding robust solutions is, however, non-trivial since the theoretically global optimal solution might not be optimal when considering robustness. The example in Figure 2 illustrates an optimization problem, to minimize $f(x)$, which is sensitive to variable perturbation in certain areas. The figure shows both the average objective values and the objective values from one sample. The solution $x1$ is the global optimal solution, but due to the variance of the objective values between different replications this would seldom be recommended in practice. However, solution $x2$ is insensitive to variable perturbation, i.e., has low variance and is therefore a robust optimal solution.

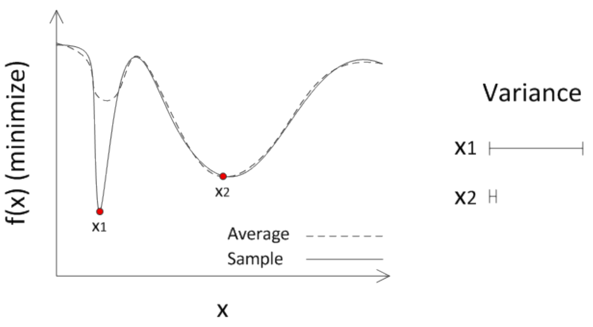

Fig. 2. Example of a sample versus a robust optimal solution for a function $f(x)$.

From an optimization perspective, variations in the manufacturing system mean that repeated evaluations of the same solution will result in different objective values. These unpredictable variations in the evaluation output are harmful to the optimization process since the evolutionary algorithm can be misdirected to propagate inferior solutions [5][8]. Robustness must therefore be considered in the optimization for successful results with real-world applicability.

One way to measure robustness is to replicate the evaluation of a solution and calculate the standard deviation. The standard deviation is calculated according to Equation 1 where $s_N$ stands for the standard deviation with $N$

replications, $x_i$ is the i:th replication objective value, and $\bar{x}$ is the mean objective value of all the replications.

$$s_N = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}$$

Eq. 1. Standard deviation based on samples.

The idea of this paper is to use standard deviation as an additional objective in the optimization process, hence treating the problem as a multi-objective problem. The difficulty with multi-objective problems is that there is usually no single optimal solution with respect to all of the objectives, as improving the performance of one objective means decreasing the performance of another [8]. One way to handle conflicting objectives is to derive a set of alternative trade-offs, so called Pareto-optimal solutions [9]. Figure 3 illustrates the Pareto concept for a minimization problem with two objectives $f_1$ and $f_2$. In this example, solutions A-D are non-dominated, i.e., Pareto optimal, since for each of these solutions no other solution exists that is superior in one objective without being worse in another objective. Solution E is dominated by B and C (but not by A or D, since E is better than these two in $f_1$ and $f_2$, respectively). Different Pareto ranks can also be identified among solutions. Rank 1 includes the Pareto-optimal solutions in the complete population, and rank 2 the Pareto-optimal solutions identified when temporarily discarding all solutions of rank 1, and so on.

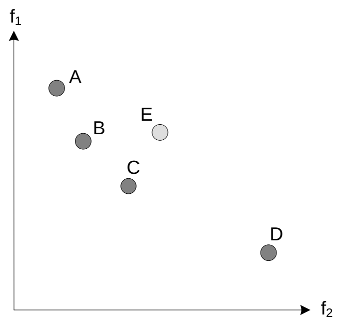

Fig. 3. Illustration of dominance.

The proposed method of achieving robust, near-optimal solutions through a multi-objective approach is implemented in the study in the evolutionary algorithm NSGA-II for testing (it should, however, be pointed out that the method as such is general and also applicable to other algorithms.) A description of NSGA-II follows in the next section.

## 3. Optimization algorithm

Several different multi-objective evolutionary algorithms have been suggested including, for example, MOGA (multiple objective genetic algorithm) [10], NPGA (niched-pareto genetic algorithm) [11], PAES (pareto-archived evolution strategy) [12] and NSGA-II (non-dominated genetic algorithm) [13]. In this study NSGA-II is implemented since it is a commonly used state-of-the-art algorithm within multi-objective evolutionary optimization and has proven to successfully solve complex real-world problems within various domains [14].

NSGA-II belongs to the class of evolutionary, population-based global search techniques. In evolving a population of solutions, evolutionary algorithms apply biologically inspired operations for selection, crossover, and mutation. The operators are applied in a loop, and an iteration of the loop is called a generation. In the pseudo code below, the basic steps involved in this evolutionary process are presented.

```
Initialize population
Evaluate the fitness of solutions in the population
repeat
  Select solutions to reproduce
  Form a new population through crossover and mutation
  Evaluate the new solutions
until terminating condition
```

In this study a solution consists of a set of product priorities, and for the initial population solutions are generated randomly. During each generation, a proportion of the solutions in the population is selected to breed offspring for the next generation of the population (that is, create new solutions). Contrary to an ordinary evolutionary algorithm, NSGA-II selects solutions for the next generation of the population based on Pareto ranks. Rank 1 includes the Pareto-optimal solutions in the complete population, and rank 2 the Pareto-optimal solutions identified when temporarily discarding all solutions of rank 1, and so on. More specifically, the selection of solutions for the next generation is done from the set R, which is the union of the parent population and the offspring population (both of size N). Pareto-based sorting (also called non-dominated sorting) is applied to R, and the next generation of the population is formed by selecting solutions from one of the Pareto fronts at a time. The selection starts with solutions in the best Pareto front, then continues with solutions in the second best front, and so on, until N solutions have been selected. If there are more solutions in the last front than there are remaining to be selected, crowding distance is calculated to determine which solutions should be chosen. All of the remaining solutions are discarded. The selection procedure is illustrated in Figure 4 (adopted from [15]).

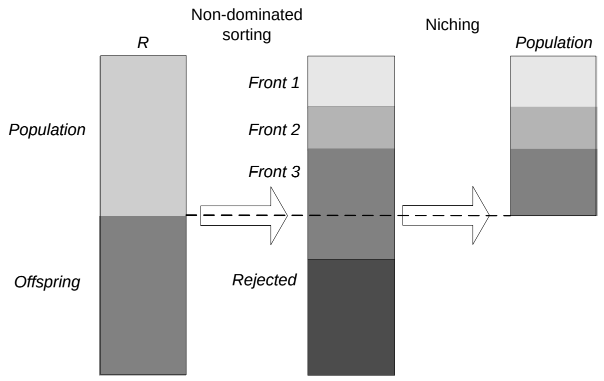

Fig. 4. NSGA-II process.

From the solutions selected through the process described in Figure 4, new solutions are created to form the next generation of the population. For the creation of each new solution, two parent solutions are chosen through tournament selection in which the highest ranking solution located in the least crowded area is the one chosen for mating. Through crossover between the parents, a new solution (an offspring) is produced. Occasionally, the new solution can undergo a small mutation in order to keep the diversity of the population large and avoid local minima. A mutation involves changing an arbitrary part of a solution with a certain probability, in this case swapping the priorities of two solutions. When an offspring population has been formed, the procedure starts all over again, and the process is repeated until a termination condition is fulfilled.

## 4. Real-world test case

For evaluating the real-world applicability of the proposed method, it is applied on a product sequencing problem found at the company GKN Aerospace in Sweden. GKN Aerospace manufactures different components for aircraft engines and aero derivative gas turbines in both commercial and military markets. The focus of the study is a newly installed workshop that produces turbine frame structures which demands high quality controls. The workshop consists of washing machines, x-ray stations, liquid penetrant testing, automatic laser and plasma welding machines, manual welding stations, CNC machines and manual burring machines. For the study, the x-ray stations have been selected to focus on as these are considered a critical point in the production flow. In the x-ray stations the next product to process when a station becomes free is currently selected manually by an operator. For a human to determine the best product sequencing is, however, not trivial due to the complexity of the manufacturing process in combination with a fluctuating inflow and variable operation times. This fact has raised a need to perform automatic optimizations of the product sequencing. The aim of the optimization is to decrease delays, as the products being processed are highly capital-intensive, and on-time deliveries are important.

To be able to perform optimizations, experts at the company have created a simulation model of the workshop. The simulation model is created using the SIMUL8 software and contains all machines, operators, fixtures and buffer zones present in the real workshop. The machines have setup times, operation times and breakdown settings that represent the real machines. The stochastic variations present in the workshop are also mimicked in the model in order to make it as realistic as possible. These variations mean that the products may enter the operations in different orders when running several simulations, although their priorities and all other input data remain the same between the runs. This in turn means that the optimization results (in the form of product delays) will vary between each run. By running several replications of the same scenario, a standard deviation representing the variation can be calculated and this value used in the proposed method for finding robust solutions.

The simulation is used by the NSGA-II algorithm to evaluate delays in the system given a specific product sequencing (set of product priorities). The simulation-optimization process is presented in Figure 5. In this iterative process the NSGA-II algorithm generates a solution and feeds it to the simulation model, which computes performance. Based on the evaluation feedback obtained from the simulation model, the NSGA-II algorithm generates a new set of parameter values and sends these to the simulation. A single simulation run takes approximately one second including input and output processing. The company's optimization time budget was set to three hours, which allowed for 10,000 simulations.

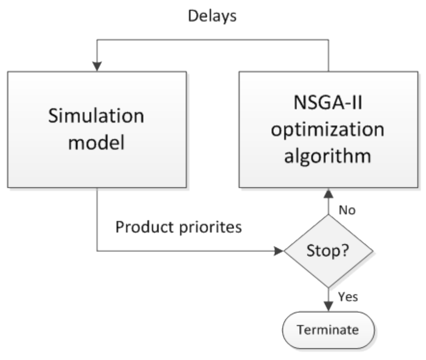

Fig.5. Simulation-optimization process.

## 5. Algorithm configuration

The NSGA-II algorithm, as all evolutionary algorithms, requires a genetic encoding (representation) of a solution. There are three main types of encodings for evolutionary algorithms: binary encoding (string of bits), value encoding (string of variables, e.g. integers, real values, characters, etc.) and permutation encoding (string of ordered, unique integers). In this study, permutation encoding is used since it enables a direct representation of a set of product priorities without transformations.

In the algorithm an order crossover is used in the evolutionary process to create offspring solutions (Figure 6). An ordered crossover is necessary due to the strict requirement of uniqueness of values in a solution (no integer can be repeated as this would result in the same priority of two products). In the ordered crossover, two parents, $p1$ and $p2$, are used to create two offspring, $o1$ and $o2$. The crossover operation starts by copying a range of the string from $p1$ to $o1$; then, the rest of the numbers will be copied from $p2$ to *offspring 1* in the order that they appear and only numbers that have not already been copied. The same process applies to $o1$ but with $o2$ as the first parent and $p1$ as the second. It can be noted that the parents are selected through binary tournaments in which two solutions in the population are randomly picked, and the one with the best quality is selected to become a parent.

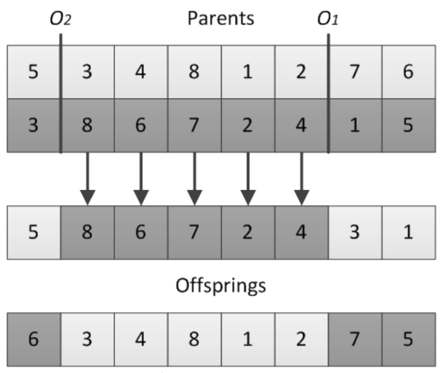

Fig. 6. Example of ordered crossover.

The created offspring are muted using a partial shuffle mutation. In this procedure, integers are randomly swapped in a range as shown in Figure 7. First, two offsets are randomly generating, and then all integers in the range between the first and the second offset are swapped randomly.

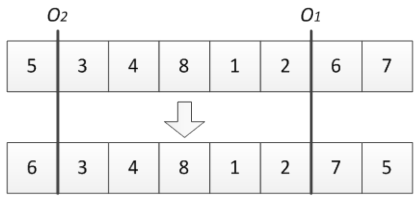

Fig. 7. Example of mutation.

The mutation, as well as the crossover, is performed by a certain probability, and these parameter values must be set as part of designing an experiment. The NSGA-II algorithm also requires two additional parameter values: population size and number of offspring. Since the optimal parameter values are dependent on the nature of the problem, the values to use must be found through experimental examination of alternative values for the specific problem [16-17]. To find a good parameter configuration for the optimization problem considered in this study, three alternative settings have been tested for each of the relevant parameters:

- Population size: 20, 60, and 100
- Number of offspring: 20, 60, and 100
- Mutation probability: 5%, 10%, and 15%
- Crossover probability: 50%, 70%, and 90%

All combinations of the different settings were tested for all parameters, which means that a total of 81 experiments were performed. The optimal settings found are presented in Table 1.

Table 1. Algorithm settings.

| Parameter | Setting |
|---|---|
| Population size | 100 |
| Number of offspring | 100 |
| Mutation probability | 10% |
| Crossover | 90% |
| Mutation | 10% |

For the test case, experts at GKN Aerospace have specified a typical production setup including 59 products. Results from the optimization of these products are presented in the next section.

## 6. Results and discussion

The result from optimizing the test case using the proposed method of using standard deviation as an additional objective is presented in Table 2 (average of ten replications). Note that both objectives in the table are to be minimized. For company integrity reasons the numbers have been proportionally scaled, and units are not specified. For comparison the optimization is also run without standard deviation as an additional objective. As can be seen in the table, there is virtually no difference between the two approaches when it comes to delay (objective 1). However, when it comes to standard deviation (objective 2), the proposed method is significantly superior. As the standard deviation becomes smaller, the variations between runs decrease and the greater the robustness; this means that solutions with improved robustness are found using the proposed method, compared to when optimizing the problem in a traditional way (only using the mean value of objective 1).

Table 2. Optimization results.

| | Objective 1: Delay | Objective 2: Standard deviation |
|---|---|---|
| Optimization using the proposed method (with standard deviation as an additional objective) | 281.1 | 30 |
| Traditional optimization (using only the mean value of objective 1) | 287.4 | 49 |

Although the results are positive in the sense that the robustness is improved with the proposed method, an improvement in the first objective (delay) also was expected. The reason for this expectation is because it might be argued that with a high degree of variation in the solutions, the performance of the optimization algorithm is reduced as it more or less degenerates into a random search. Hence, focusing on solutions with low variation, as in the proposed method, was expected to improve the general performance of the algorithm. A reason for the non-existing improvement might be that in this specific case study both objectives are to be minimized. In using a two-dimensional graph with axes that are equidistant, the search space would be similar to Figure 8. This is because the standard deviation should not (if disregarding skewness) be larger than its mean value since the value cannot go below zero. The figure shows that when decreasing the mean value of objective 1, the standard deviation is also decreased. This means that the two objectives are not conflicting – and conflicting objectives are a desirable problem feature for multi-objective evolutionary algorithms [15]. Thus, in an optimization problem where the objective is to minimize a value, then the additional objective standard deviation does not improve the search performance. If, however, the objective were to maximize a value, such as improve the throughput per hour in a manufacturing process, then the additional objective to minimize the standard deviation might improve the search performance since then the objectives would be conflicting.

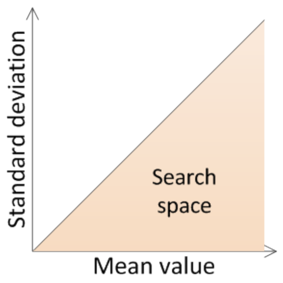

Fig. 8. Search space on mean value and standard deviation.

## 7. Summary and future work

The premise of this study is that almost all manufacturing processes are subject to uncontrollable variations, caused, for example, by human operators or worn-out machines. When optimizing real-world product sequencing problems, it is of importance to find solutions that are robust, that is, whose performance remains relatively unchanged when exposed to uncertain conditions. In this paper an extension of the traditional method of handling uncertainty through replications is suggested that aims at finding solutions with an increased degree of robustness. The basic idea is to use standard deviation as an additional optimization objective and transform the single-objective problem into a multi-objective problem. In order to optimize the two objectives simultaneously, a multi-objective evolutionary algorithm is utilized. The proposed method is evaluated using a real-world test case found at the company GKN Aerospace in Sweden. GKN Aerospace manufactures a variety of different components for aircraft engines and aero derivative gas turbines. The company has recently installed a new workshop, and the focus of the study is on the x-ray stations in this workshop. For performing optimizations, the company has created a simulation model that realistically mimics the workshop. As an optimization technique, a multi-objective evolutionary algorithm called NSGA-II is being used. The algorithm considers the mean value and standard deviation from replications of the stochastic simulation as objectives, optimizing both of them simultaneously. Results from the study show that the optimization is able to successfully find robust solutions using the proposed method when applied on the real-world test case. More test cases are, however, needed to confirm the general applicability of the method, and such studies will be in focus for the future.

An interesting notification from the study is that the general increase in algorithm performance expected with the proposed method is absent. A possible reason for this is discussed in the paper, but this aspect needs to be further investigated in order to improve the gain of the proposed method. Future work will therefore focus on experimenting on problems where the mean value is to be maximized and the standard deviation is to be minimized. Preferably, these problems should be real-world test cases in order to verify the industrial applicability of the method.

Another aspect of focus for future work is to create a simple-to-use graphical user interface for operators on the shop floor since these are not experts in either simulation or optimization and therefore cannot run an advanced software system. The most appealing approach might be to use a web-based approach as this allows easy access from everywhere given an internet connection. An advantage of a web-based system is that it also makes it possible to completely separate the graphical user interface from the simulation/optimization system, allowing for them to be refined separately.

## References

[1] Cormen, T.H., L. C. R. R. and Stein, C. (2001) Introduction to Algorithms. USA: 2nd edn, MIT Press.

[2] Gockel, N. and Drechsler, R. (1997) Influencing parameters of evolutionary algorithms for sequencing problems. In Proceedings of IEEE International Conference on Evolutionary Computation: 575-580.

[3] Cagnina, L., Esquivel, S. and Gallard, R. (2004) Particle swarm optimization for sequencing problems: a case study. In Proceedings of Congress on Evolutionary Computation: 536 – 541.

[4] Bäck, T., Fogel, D. and Michalewicz, Z. (eds) (1997) Handbook of Evolutionary Computation, Oxford University Press.

[5] Jin, Y. and Branke, J. (2005) Evolutionary optimization in uncertain environments – a survey, IEEE Transactions on Evolutionary Computation 9(3): 303–317.

[6] Bui, L., Abbass, H. and Essam, D. (2005) Fitness inheritance for noisy evolutionary multi-objective optimization, Proceedings of Genetic and Evolutionary Computation Conference,Washington, DC, USA, pp. 779–785.

[7] Branke, J., Meisel, S. and Schmidt, C. (2007) Simulated annealing in the presence of noise, Journal of Heuristics 14(6): 627–654.

[8] Büche, D., Stoll, P., Dornberger, R. and Koumoutsakos, P. (2002) An evolutionary algorithm for multi-objective optimization of combustion processes, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 32(4): 460–473.

[9] Tan, K. C. and Goh, C. K. (2008) Handling uncertainties in evolutionary multi-objective optimization, Proceedings of IEEE World Congress on Computational Intelligence, Vol. 5050 of Lecture Notes in Computer Science, Springer, Hong Kong, China, pp. 262–292.

[10] Fonseca, C. and Fleming, P. (1993) Genetic algorithms for multiobjective optimization: Formulation, discussion, and generalization, Proceedings of the Fifth International Conference on Genetic Algorithms, Urbana-Champaign, Illinois, pp. 416–423.

[11] Horn, J., Nafploitis, N. and Goldberg, D. (1994) A niched pareto genetic algorithm for multi-objective optimisation, Proceedings of the First IEEE Conference on Evolutionary Computation, Orlando, Florida, pp. 82–87.

[12] Knowles, J. and Corne, D. (2000) Approximating the non-dominated front using the pareto archived evolution strategy, Evolutionary Computation Journal 8(2): 149–172.

[13] Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T. (2000) A fast and elitist multi-objective genetic algorithm NSGA-II, KanGAL Report 2000001, Indian Institute of Technology Kanpur, India.

[14] Zhoua, A., Qub, B-Y., Lic, H, Zhaob, Z., Nagaratnam Suganthanb, P. and Zhangd, Q. (2011) Multiobjective evolutionary algorithms: A survey of the state of the art. Swarm and Evolutionary Computation 1(1): 32–49.

[15] Deb, K. (2004) Multi-Objective Optimization using Evolutionary Algorithms, second edn, JohnWiley & Sons Ltd.

[16] Goldberg, D. (1989) Genetic algorithms in search, optimization and machine learning, Addison-Wesley Publishing Co, Boston,Massachusetts.

[17] Tran, K. (2006) An Improved Multi-Objective Evolutionary Algorithm with Adaptable Parameters, PhD thesis, Graduate School of Computer and Information Sciences, Nova Southeastern University.