Available online at http://ijdea.srbiau.ac.ir

Int. J. Data Envelopment Analysis (ISSN 2345-458X)

Vol.3, No.2, Year 2015 Article ID IJDEA-00322, 11 pages

Research Article

International Journal of Data Envelopment Analysis Science and Research Branch (IAU)

Consolidated Technique of Response Surface Methodology and Data Envelopment Analysis for setting the parameters of meta-heuristic algorithms - Case study: Production Scheduling Problem

Seyed Esmail Najafi[a], Reza Behnoud[b*]

(a) *Assistant Professor of Industrial Engineering Group, Faculty of Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran*

(b) *Ph.D. Student of Industrial Engineering, Faculty of Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran*

Received 16 January 2015, Revised 20 April 2015, Accepted 23 May 2015

Abstract

In this study, given the sequence dependent setup times, we attempt using the technique of Response Surface Methodology (RSM) to set the parameters of the genetic algorithm (GA), which is used to optimize the scheduling problem of n job on 1 machine (n/1). It aims at finding the most suitable parameters for increasing the efficiency of the proposed algorithm. At first, a central composite design was created and then using the data relating to the plan, the complete second-degree model was fitted. Then, by solving the developed non-linear programming model the optimal values of the parameters determined. The performance of algorithm, considering the obtained parameters as inputs of the common Data Envelopment Analysis (DEA), was measured. This way, we can decide on the most effective kinds of problems that can be solved by GA in a similar volume. This study can be used as a model of setting parameters of evolutionary and meta-heuristic algorithms using scientific techniques to prevent disadvantages relating to trial and error methods.

Keywords: sequence dependent setup time (Traveling Salesman Problem), setting genetic algorithm parameters, response surface methodology, data envelopment analysis, Anderson- Peterson ranking model.

[*] Corresponding author: reza.behnoud@gmail.com

1. Introduction

Solving Production Scheduling Problem (PSP) divides into two parts. First, with respect to costs, we should decide on the allocation of orders (jobs, materials, etc.) to production resources (machines, workshops, etc.). Second, we should decide on the sequence and order producing of producing order using production resources. Many scheduling problems when considering all real-world assumptions become computationally too complex so that they place in the class of NP-Hard problems, which causes finding optimum solution for medium-size and larger instances in a reasonable time impossible. In such cases, one way of finding acceptable solutions is using meta-heuristic algorithms such as GA, simulated annealing (SA), taboo search (TS) and so on. Since introduction of meta-heuristic algorithms, setting input parameters of them has been always a challenging problem in improving the quality of solutions resulting of them. In earlier researches, methods such as trial and error, design experiments, Taguchi and so on are used to set parameters of these algorithms.

Teoh et al. suggested solving traveling salesman problem (TSP) using a feedback network of threshold neurons method [11]. By directly consideration of restrictions on network dynamics, they drew salesman problem on a single-layer feedback neural network. Then, parameters of the proposed network were set using a genetic algorithm, which ensured permanent convergence of the intended neural network for the various problems. Here, setting parameters of the genetic algorithm is quite experimental and is done through 10 times running of GA for several experimental problem, aiming at statistical examination of algorithm in terms of resistance and compatibility. Hardas et al. [6], in order to solve the problem of allocation of the elements in the assembly of printed circuit boards used genetic algorithm. Then, in order to solve a sequence determination problem, including 10 elements embedded on a machine, they employed experiment designs to determine the best type of crossover, reproduction type, crossover rate and reproduction rate. Cheng and Chang [4] proposed a genetic algorithm to optimize the scheduling problem of flow-shop scheduling and using Taguchi experimental design predicted a combination plan for the optimum parameters of the proposed algorithm. They tested and optimized seven factors of the proposed algorithm, including initial solution, selection method, crossover approach, the mutation rate, population size, crossover rate, and the mutation method. Gholami et al. [5], considering the assumption of sequence dependent setup times and random failures of machinery, developed a genetic algorithm to solve the flow-shop scheduling problem. They used Taguchi method to set the parameters of the algorithm.

In the used method, controllable factors in internal orthogonal vectors and noise factors in external orthogonal vectors were placed. Afterwards, the values obtained from tests were converted to S/N ratio and then the optimal parameters of the algorithm were obtained. Naderi et al. [9] also used Taguchi method to set the parameters of the SA algorithm that was developed to solve hybrid flow-shop scheduling, assuming that sequence dependent setup times was given. They considered factors such as initial solution, coding pattern, cooling scheduling, initial temperature, the number of neighborhood searches at every temperature, initial and stopping temperatures as well as the structure of neighborhood searching as controllable factors of SA and by selecting different levels for each of these factors, they found optimum values using Taguchi method.

Other technique used to set the parameters of the meta-heuristic algorithms is the response surface methodology (RSM) technique. RSM is a combination of mathematical and statistical techniques that is suitable for modeling and analysis of problems in which the response variable is affected by several input variables, which aims at optimizing the responses [8]. Among the earliest researches done to set the parameters of meta-heuristic algorithm using RSM is the study by Wang and Wu [12], where a six-stage process based on RSM was developed to identify and optimize parameters of SA algorithm subject to the computational time constraints. In the research, they considering RSM followed a steepest descent method with respect to the constraints and when further improvement was impossible, they fitted a second-degree RSM and set optimal parameters of model so that satisfying model's constraints.

In this paper, we have used the genetic algorithm [7] to optimize the scheduling problem of allocation of n job to 1 machine (n/1) subject to the sequence dependent setup times. In addition, we have used RSM to find optimal values of the parameters of the developed algorithm. Then, performance of the proposed method is analyzed using a mathematical model for DEA.

Rest of the paper is organized as follows. Section 2 describes the genetic algorithm proposed to solve the scheduling problem (n/1) considering sequence dependent setup times. In Section 3, using an example of the scheduling problems, the process of setting parameters of genetic algorithms using RSM and DEA is described. Section 4 presents concluding remarks.

2.Genetic algorithm for scheduling problem

The production scheduling problem of n job on 1 machine is simplified by consideration of a series of simplification assumptions and leaving away values of its parameters from those existing in real-world problems. These simplifying assumptions are as follows:

1) all activities are available at workshop at the beginning of the planning period (when doing jobs, no new job enters),

2) descriptions of operations are pre-known and time need to complete job is definite and certain,

3) the setup time of jobs is independent of the sequence of them,

4) preemption is not permitted,

5) if there is some job to be done, machine should not be kept idle,

6) during the planning period, all machines are available and none of them breakdown because of preventive maintenance or accidental failure,

7) jobs on machines continue operating without occurring any fault, i.e. it is free of waste or rework.

Given the above 7 assumptions, the number of possible sequences on the machine decreases from infinitive to n!, but with releasing each of the above assumptions and closing environment of the problem to that in real-world problems, the computational and structural complexity of these problems increases significantly; so that when the volume of inputs are high, then exact mathematical programming algorithms, due to the high volume of calculations and unacceptable solution time, are not accountable to resolve the problems. In these cases, heuristic and meta-heuristic algorithms are suggested to overcome the computational complexity. In this article, we have removed the assumption of sequence independent setup times. In this case, the problem of n jobs on 1 machine subject to the sequence dependent setup times is NP-hard.

However, it is clear that in the scheduling problem of single machine, considering each sequence of jobs on machine, time associated to completion tasks is constant and equal to sum of the time of jobs done on machine. Considering sequence dependent setup times, makespan in addition to the sum of basic times needed to do jobs on machines also include the setup times spent on the between tasks.

This work is aimed at minimizing the completion time of tasks. Given the fact that the sum of completion jobs on machines is constant, we neglect it from the optimization and just to minimize the total setup times of sequence. In this case, taking into account the problem of $n/1//C_{\max}$ as well as the assumption of the sequence dependent setup times in scheduling environment, the problem exactly converts to the problem of traveling salesman problem (TSP). So, considering tasks as cities and considering the sequence dependent setup times as distance between cities in TSP problem we seek to find best tour, which start from a given city; passes all cities and then come back to the first city.

Such tour is equivalent to an optimal or near-optimal value of total setup times of sequence of jobs in scheduling problem, which by adding basic completion time of jobs to this value, the minimal makespan results. In this study, assuming constant completion times of jobs on machines, for the convenience, it is assumed that the setup times in sequence are equal to the makespan (i.e. we neglect total completion time of jobs on machines in calculations of makespan).

In this study, we use the well-known meta-heuristic algorithm of GA to solve the scheduling problem. The solutions are displayed as follows. Each chromosome (solution) is a permutation of the numbers of assigned jobs to machine. In order to produce initial solutions (chromosomes produced in the first stage), we have used the random method and to produce new chromosomes in each generation, the following operators are used: a) one-point crossover operator with roulette wheel selection, b) mutation operator with random selection strategy, c) reproduction operator with elitist strategy.

The calculation of the makespan is used as a measure of fitness of solutions in each generation. But what is important here is obtaining the best parameters for the genetic algorithm; so that provide us with the best solution at the shortest possible time. Then, using the technique of response surface methodology, we have set parameters of GA.

3. Setting GA parameters

The proposed genetic algorithm is implemented in MATLAB. After the implementation of genetic algorithm, we look for the best input parameters for it, aiming at finding the best solutions. For this purpose, the technique of response surface methodology is used to set parameters. In this section, the application procedure associated to response surface methodology to set the parameters of the genetic algorithm is fully described. Numerical example explained here is about the simulation problem of scheduling 29 jobs on 1 machine, equivalent to TSP with 29-cities [10]. In the following, full details of example including distance between cities, the best tour obtained from resolving this problem as well as the best traversed distance (fitness function value) obtained from the problem are presented.

First step: First, we must select the controllable variables of genetic algorithm, which can affect the solution of the algorithm. Intended parameters include percentage of crossover operator (*Pc*), percentage of mutation operator (*Pm*) (since *Pr* = 1-*Pc*-*Pm*, it is not needed to consider the parameter *Pr*), the number of produced solutions in each generation (*pop*) and stop condition (the maximum number of generations of GA that is shown by (*gen*)). the lowest fitness function or the minimum makespan is considered as a measure of optimization and level of response.

Step Two: In this step, we have employed a central composite design to design experiment. In the plan, for each of the four main factors, two levels is determined and used in experiments. Levels are shown as -1, when factor is at a low level, and +1, when intended factor is at a high level [12]. For parameters *pop, gen, Pc, Pm,* values of high levels are equal to 500, 500, 0.80, 0.1, respectively; and those values for the low levels are equal to 300, 300, 0.60 and 0.05, respectively. For *pop,* two levels of 300 and 500, for *gen* two levels of 300 and 500, for *Pc* two levels of 0.60 and 0.80, and for *Pm* two levels of 0.05 and 0.1 are considered. Values of high and low levels of each factor are obtained empirically and through the recurrent implementation of genetic algorithm. In addition to the main points, 7 center points and 8 axial points are also considered.

$$300 \leq pop \leq 500,\ \ 300 \leq gen \leq 500,\ \ 0.6 \leq P_c \leq 0.8,\ \ 0.05 \leq P_m \leq 0.1$$

Now considering main, center and axial points and regarding specified parameters, 31 experiments are conducted by running developed GA 31 times for the problem of *bays29* and recorded in Table 1.

Table 1: Results obtained from experiments

Pc	*Pm*	*pop*	*gen*	*makespan*
0.6	0.100	500	300	2021
0.8	0.100	300	500	2341
0.7	0.075	400	400	2145
0.9	0.075	400	400	2187
0.7	0.075	400	400	2138
0.7	0.075	400	400	2126
0.7	0.025	400	400	2206
0.6	0.050	300	300	2299
0.8	0.100	500	500	2046
0.7	0.075	600	400	2064
0.7	0.075	400	400	2142
0.5	0.075	400	400	2155
0.7	0.075	200	400	2399
0.8	0.100	500	300	2041
0.7	0.075	400	600	2120
0.6	0.050	500	300	2039
0.6	0.100	300	300	2285
0.7	0.075	400	400	2137

Table1(continued)

0.8	0.100	300	300	2349
0.8	0.050	500	300	2058
0.6	0.100	500	500	2020
0.7	0.125	400	400	2183
0.8	0.050	300	300	2363
0.6	0.050	300	500	2291
0.8	0.050	300	500	2354
0.7	0.075	400	200	2161
0.7	0.075	400	400	2139
0.6	0.050	500	500	2034
0.7	0.075	400	400	2139
0.6	0.100	300	500	2264
0.8	0.050	500	500	2053

Step Three: In order to implement technique of response surface methodology with respect to data obtained from designing experiments, we fit the second-degree model, which includes both mutual as well as pure second-degree effects of factors. This model is the most complete model of RSM. In fact, due to a curving in actual RSM, it is usually needed to second grade or higher models to approximate solution. To fit RSM, we used Minitab. The fitted model is as follows:

$$\hat{y} = b_0 + \sum_{i=1}^{4} b_i x_i + \sum_{i=1}^{4} b_{ii} x_i^2 + \sum_{\substack{i \\ i<j}} \sum_{j} b_{ij} x_i x_j$$

$$\hat{y} = 3162.2 - 473P_c - 3968.3P_m - 2.3pop - 0.2gen + 729.2{P_c}^2 + 21066.7{P_m}^2 + 550P_c \times P_m - 1.2P_c \times pop + 0.1P_c \times gen + 0.3P_m \times pop$$

It should be noted that after variance analysis of the full second-degree model, effect coefficients of pop2, gen2, Pm×gen and pop×gen are deleted from model, because their values was equal to zero. From ANOVA table, we can see F-value = 23.64 and P-value=0 for the regression model, which confirms the fitness of proposed RSM. Also, the square of correlation coefficient of model is equal to 95.39%, which shows that the fitted model was unable to describe only 4.61% of parameters, which is acceptable.

A useful way to check for normality is to plot the probability normal diagram of residuals. If error is normally distributed, the chart looks like a straight line. Thus, considering the normal probability chart in Figure 1, we can conclude that residuals are standard and almost normal and the obtained model is capable of describing data, because except for two points of first and last, the cumulative distribution of other residuals is similar to straight line. After obtaining coefficients, the fitted model is considered as the best one, and then we go to step four.

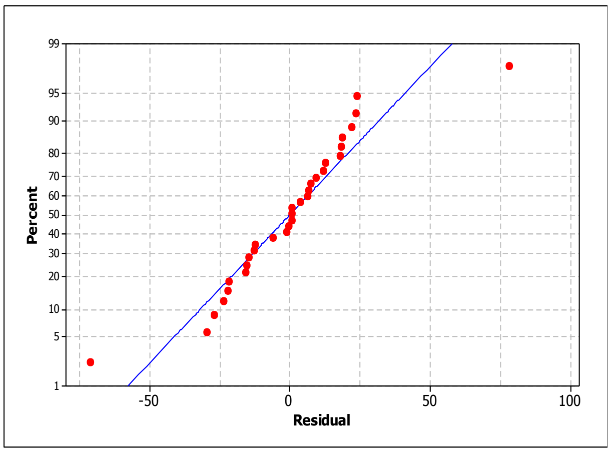

Figure 1: The normal probability plot of the second-degree model

Step Four: the fitted second-degree model obtained in the previous step is considered as the objective function of a non-linear optimization problem subject to upper and lower bounds of each of the parameters. To solve this nonlinear programming model, we used one of the operation research's software. The obtained solution gives us values of values of each of parameters. Here, we have used Lingo to solve the model.

$$\begin{aligned} min\, y = 3162.2 - 473P_c - 3968.3P_m \\ - 2.3pop - 0.2gen \\ + 729.2{P_c}^2 + 21066.7{P_m}^2 \\ + 550P_c \times P_m - 1.2P_c \times pop \\ + 0.1P_c \times gen + 0.3P_m \\ \times pop \end{aligned}$$

$$\begin{aligned} s.t:\ & 0.6 \le P_c \le 0.8 \\ & 0.05 \le P_m \le 0.1 \\ & 300 \le pop \le 500 \\ & 300 \le gen \le 500 \end{aligned}$$

Using Lingo, optimal values of parameters were obtained. The optimal values for the parameters *Pc*, *Pm*, *pop*, *gen* were 0.6710605, 0.08186419, 500 and 500, respectively.

$${P_c}^* = 0.6710605, \qquad {P_m}^* = 0.08186419,$$
$$pop^* = 500, \qquad gen^* = 500$$

In other words, to solve this scheduling problem, it is enough to set values of parameters with numbers obtained above. This way, GA with a high level probability results in appropriate solutions.

Step Five: Finally, to assess quality of solutions obtained from tuned GA, four samples of mentioned scheduling problems were optimized. Considering similarity of the scheduling problem and TSP, problems *gr24, fri26*, *bayg29* and *bays29* from the set of TSP problems were selected [10]. These problems are all from the same level of the number of jobs, i.e. number of cities of TSP, and thus of the same level of complexity. Complementary information on obtained solutions from the algorithm as well as the best available solutions of these problems that are obtained from literature is presented in Table 2.

As it is clear from Table 2, average value of results obtained from 5 times running of GA on the similar problem instances were acceptable compared to available best solutions on these problems in the literature.

In addition, in one of the replications of developed GA, we found the lower bound of 2020 for the fitness function of problem bays29, which reflects the good performance of the intended GA as well as goodness parameter setting by response surface methodology technique.

Step six: One of the most common models to assess performance is Data Envelopment Analysis, which is a mathematical programming method to evaluate decision making units (DMUs) [3]. This method based on a set of empirical observations estimates empirically the efficiency frontier through comparing the relative performance of each of units. Now, we set Table 5 in a way that can be employed in solving this mathematical programming model. Here, we aim at finding the most effective type of problem for being solved by genetic algorithm.

Giving attention to a few key points about inputs and outputs of Table 3 is important. The first point is that except the number of cities,

Table 2: Comparison the results obtained from adjusted GA with the best solutions available in the literature

Proble m	No. of jobs (cities)	Lower bound	Average of 5 runs of algorithm St: Pc= 0.67 ‹Pm=0.082 ‹ pop=500 ‹gen=500.	Deviation	CV
gr24	24	1272	1308	16.00	0.012232415
fri26	26	937	976	17.08	0.0175
bayg29	29	1610	1699	18.05	0.010623896
bays29	29	2020	2093	22.33	0.010621118

Table 3: Input and output values of DEA model

Problem (DMU)	n (I1)	Pm (I2)	Pc (I3)	gen (I4)	pop (I5)	Average of 5 solutions (O1)	Average run time (O2)
DMU1	24	0.082	0.67	500	500	$\frac{1}{1308}$	$\frac{1}{584}$
DMU2	26	0.082	0.67	500	500	$\frac{1}{976}$	$\frac{1}{642}$
DMU3	29	0.082	0.67	500	500	$\frac{1}{1699}$	$\frac{1}{724}$
DMU4	29	0.082	0.67	500	500	$\frac{1}{2093}$	$\frac{1}{711}$

other inputs of model have constant and the same values. In fact, we want to find out that by using the parameters obtained in the previous step, which type of problem has better performance. Second point is paying attention to reversing output values relative to the previous table. Clearly, DEA seeks for the greatest efficiency through the use of minimal resources (inputs) and obtaining maximal products (output). Therefore, if we want to use the objective function of TSP as the model output, some changes must be applied, because it has the type of cost and minimizing it is desirable. The same issue is also applicable about algorithm run time. For this reason, here aforementioned values are considered reversely.

Given above explanations, we have a model with four decision making units, five inputs and two outputs. We look for a decision-making unit (DMU) without changing values of four categories of inputs and by increasing values of outputs to be the most effective one. Given above information and assumptions, we establish the following model of constant return to scale of DEA for all DMUs and then solve the problems by Lingo.

$$max\, \varphi$$

$$s.t\colon \sum_{j=1}^{4} \lambda_j x_j \le x_p$$

$$\sum_{j=1}^{4} \lambda_j y_j \ge \varphi y_p$$

$$\lambda_j \ge 0;\ \ j = 1,2,\dots,n;$$

After running the model for each of the four decision-making units, results recorded in Table 4 is obtained.

Table 4: Values of efficiency and weights of DMUs of DEA model

DMU	φ	λ_1	λ_2	λ_3	λ_4
DMU1	1	1	0	0	0
DMU2	1	0	1	0	0
DMU3	1.24	1	0	0	0
DMU4	1.22	1	0	0	0

If obtained φ from the above model for each of DMUs was equal to 1, that unit is efficient and the solution is optimum, but if φ is a number greater than 1, then we can say that there is another solution is in the existing production possibility set that is better than intended solution and therefore it is not optimum. As can be inferred from Table 4, decision making units of 1 and 2 are efficient but units 3 and 4, given values obtained for φ, are inefficient. In other words, problems 24 and 26 of TSP are the best problems to be solved and obtaining the best possible solution of developed GA using the set parameters.

However, we are only looking for a unique model to be solved by our algorithm. Therefore, we must choose between two types of efficient problems. For this purpose, we used ranking methods of effective DMUs, which is explained at the following.

Step Seven: At this stage, we look for the best problem to be solved by genetic algorithm as a representative of a certain volume of production scheduling problems, which are subject to the sequence dependent setup times (TSP). In the previous stage, two units (two problems) determined as being effective. We are currently looking for rankings of efficient units to choose only one of them as a final best solution.

Andersen and Petersen [1] proposed a method for ranking efficient units, which made determination of the most efficient unit possible. By this technique, score of efficient units in input-oriented model can be more than one, and thus efficient unit as inefficient units can be ranked. Using this method, the intended decision-making units are deleted from the set of production possibility set and then we apply DEA on the remaining DMUs. Andersen-Petersen's model (AP) for efficient units is as follows:

$$max \sum_{r=1}^{2} u_r y_{rp}$$

$$s.t\colon \sum_{i=1}^{5} v_i x_{ip} = 1$$

$$\sum_{r=1}^{2} u_r y_{rj} - \sum_{i=1}^{5} v_i x_{ij} \le 0; \;\; j = 1,2,3,4 \;\; j \ne p$$

$$u_r \ge 0; \;\; r = 1,2;$$

$$v_i \ge 0; \;\; i = 1,2,3,4,5;$$

The results of efficient units including 24 and 26 cities are shown in Table 5.

Comparing AP efficiencies of two units 1 and 2, we can see that unit 2 with efficiency of 1.34 has the highest rate of efficiency and is ranked as the best unit. So, we come to the conclusion that the best problem for being solved by genetic algorithm with the greatest efficiency is TSP with 26-city (Scheduling of 26 jobs on 1 given machine subject to the sequence dependent setup times).

Table 5: The values of efficiency and weights of efficient DMUs in Anderson-Peterson model

DMU	$\theta_{A.P}$	u_1	u_2	v_1	v_2	v_3	v_4	v_5
1	1.19	0	695.5	0.041	0	0	0	0
2	1.34	1308	0	0.038	0	0	0	0

The proposed procedure can be used as a scientific technique for setting parameters of meta-heuristic algorithms, instead of using trial and error method, when the optimal values of parameters are unknown.

4. Conclusion

In this paper, we studied the optimal values of parameters of GA used in optimization of the problem of scheduling n jobs on 1 machine (n/1), subject to the assumption of the sequence dependent setup times. For this purpose, we used Response Surface Methodology (RSM). In addition, regarding that this scheduling problem is equivalent to the traveling salesman problem, in order to evaluate parameters of the algorithm, we used an example of TSP existing in the literature. The algorithm was used. Afterwards, a number of problems of the same level were optimized. The results obtained were used as inputs and outputs of mathematical model of DEA. Then, we tried to find the best representative of problem sets to get the best response from the proposed algorithm. Generalizing the proposed procedure to all the meta-heuristics, we can be largely assured about the accuracy of the parameters used in algorithms as well as obtaining an appropriate and acceptable solution in the lowest time and with the least possible error. Finally, we can be ensured that we have avoided as much as possible of the risks and errors associated to trial and error method in setting parameters of meta-heuristics.

References

[1] Andersen, P., & Petersen N. C. (1993). A Procedure for Ranking Efficient Units in Data Envelopment Analysis. *Management Science*, 39(10), 1261-1264.

[2] Charnes, A., Cooper, W. W., Lewin, A. Y., & Seiford, L. M. (1997). Data Envelopment Analysis: Theory, Methodology and Applications. *Dordrecht: Kluwer Academic Publisher*.

[3] Charnes, A., Cooper, W. W., & Rhodes, E. (1978). Measuring the Efficiency of the Decision Making Units. *European Journal of Operational Research*, 2(6), 429-44.

[4] Cheng, B. W., & Chang, C. L. (2007). A Study on Flowshop Scheduling Problem Combining Taguchi Experimental Design and Genetic Algorithm. *Expert Systems With Applications*, 32(2), 415-421.

[5] Gholami, M., Zandieh, M., & Alem-Tabriz, A. (2009). Scheduling Hybrid Flowshop with Sequence-dependent Setup Times and Machines with Random Breakdowns. *International Journal of Advanced Manufacturing Technology*, 42(1-2), 189-201.

[6] Hardas, C. S., Doolen, T. L., & Jensen, D. H. (2008). Development of a Genetic Algorithm for Component Placement Sequence Optimization in Printed Circuit Board Assembly. *Computers & Industrial Engineering*, 55(1), 165-182.

[7] Holland, J. H. (1975). Adaptation in Natural and Artificial Systems. *University of Michigan Press*, Ann Arbor.

[8] Montgomery, D. C. (2005). Design and Analysis of Experiments. *John Wiley & Sons*, 6th Ed, New york,.

[9] Naderi, B., Zandieh, M., & Roshanaei, V. (2009). Scheduling Hybrid Flowshops with Sequence Dependent Setup Times to Minimize Makespan and Maximum Tardiness. *International Journal of Advanced Manufacturing Technology*, 41(11-12), 1186-1198.

[10] Reinelt, G. (1991). TSPLIB-A Traveling Salesman Problem Library. *ORSA Journal on Computing*, 3(4), 376-384.

[11] Teoh, E. J., Tan, K. C., Tang, H. J., Xiang, C., & Goh, C. K. (2008). An Asynchronous Recurrent Linear Threshold Network Approach to Solving the Travelling Salesman Problem. *Neurocomputing*, 71(7-9), 1359–1372.

[12] Wang, T. Y., & Wu, K. B. (1999). A Parameter Set Design Procedure for the Simulated Annealing Algorithm under the Computational Time Constraint. *Computers & Operations Research*, 26(7), 665-678.