Global Journal of Pure and Applied Mathematics.
ISSN 0973-1768 Volume 11, Number 4 (2015), pp. 2601-2614

http://www.ripublication.com

A new heuristic algorithm for minimization of total cost including customer's waiting cost and machine setup cost for sequence dependent jobs on a single processor using GA

Neelam Tyagi[#1], Mehdi Abedi[*2], A.B.Chandramouli[^3], and Yaser Zarook[*4]

[#] *Corresponding Author- Department of Mathematics, Graphic Era University, Dehradun, Uttarakhand, India*
[1]*neelam24tyagi@gmail.com*
[*]*Mazandaran University of Science and Technology, Babol, Iran*
[2]*abedi_ac@yahoo.com*
[4] *yacerzarok@yahoo.com*
[^] *Department of mathematics, Meerut College, Meerut, Uttar Pradesh, India*
[3]*dr.abchandramouli@gmail.com*

Abstract

This article proposes the problem of scheduling N non-preemptive jobs on a single machine, each customer having different weights. The objective is to minimize the customer's weighted waiting time cost and machine setup cost in processing of sequence dependent jobs. The waiting time of a customer is specified as the time which machine is processing other customer's jobs while the customer wait for the machine to process his job. Here, some attributes for the optimal sequence are considered and it is demonstrated that the problem is NP-hard. A new Integer Non Linear Programming (INLP) for the problem is formulated and using this formulation we can quickly find optimal solutions for small size problems. However, the problem being NP-hard and, a Genetic Algorithm(GA) is proposed to solve problems of large size in a reasonable computational time.

Keywords: single machine scheduling, customer waiting time, setup cost, genetic algorithm.

I. INTRODUCTION

In the past few decades, a notable number of researchers in the scheduling area have been focused on the problems in Just-In-Time (JIT) environment[12]. They typically fall into the following two categories (Baker[8]: in first category is cleared due date, in second category is not cleared due date. Objectives with clear due dates are often considered favorable so far as the functions become detachable at the point of the corresponding due dates, It becomes easy for both mathematical behavior and algorithm design[14]. Problems of second categories with property of functions not detachable are usually found more difficult to solve. In this study we discuss a problem in the second category. ‘‘Variance minimization’’ is one of the first set of models of non regular measurement with not clear due date. It was primarily proposed by Merten and Muller [1], and it has ripped large regard in the past four decades (Gowrishankar et al [7] & Bagchi, [18]. They consider the problem of minimizing Completion Time Variation (CTV) and Waiting Time Variation (WTV) in a single machine environment. They indicated that the sequence minimizing 1||CTV was antithetic to the sequence minimizing 1||WTV and thus constitute the balance between these two problems. Eilon and Chowdhury[17] demonstrate that all the optimal sequence of 1||WTV should be V shaped, that is, the jobs before the smallest job are schematization in a depreciatory order of their processing times, and the jobs after the smallest job are schematization in an accessional order of their processing times. That is shown in Fig.1, in the same paper, they present the prime set of heuristics for the 1||WTV problem.

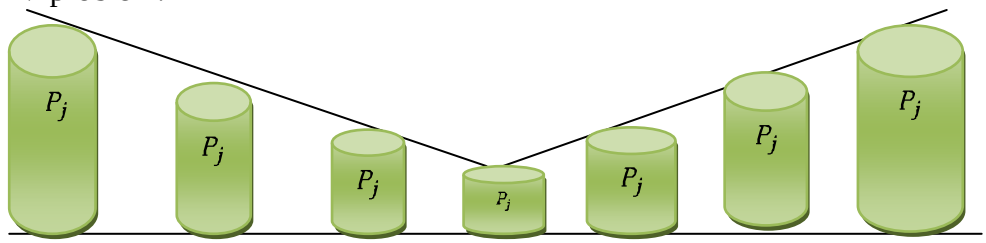


Fig. 1. optimal sequence of 1||WTV should be V

Li et al.[11], Viswanathkumar and Srinivasan [4]& Ye et al.[10] studied the optimal solution for 1|| WWTV (Weighted Waiting Time Variance). In their work they further demonstrate that optimal solution should be V-shaped in terms of P_j/v_j if the jobs are ‘‘agreeably weighted’’, that is, $p_i > p_j$ always implies $v_i < v_j$. (Fig.2)

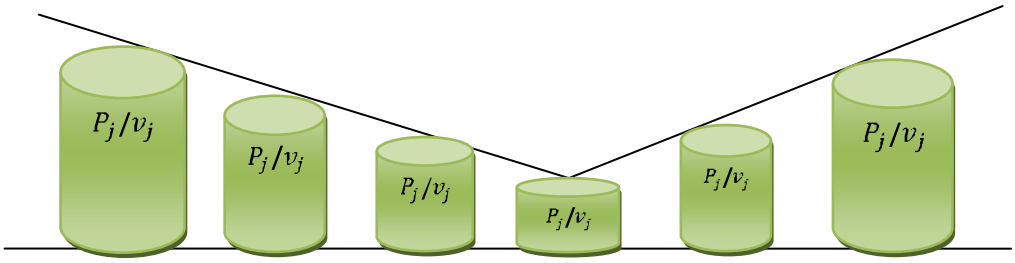


Fig.2: optimal solution of 1||WWTV

In the same work when $p_i = p_j \; \forall \; i, j$, afterward the optimal solution is ^-shaped in condition weights, the ^- shaped feature is demonstrate in Fig.3. ^-shape means that the jobs before the largest-weighted one are processed in a ascending order of their weights, and the jobs next the largest-weighted one are scheduled in an descending order of their weights (Mittenthal et al, [6]).

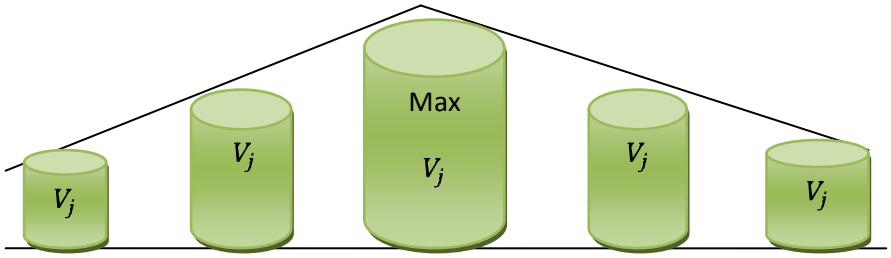


Fig.3: Optimal solution is ^-shaped in condition weights

In this paper, we consider customer's waiting time instead of job's waiting time, namely waiting time is calculated for any customer such that each customer has some of the jobs specified. Our objective is minimization of total weighted waiting time cost of customers and total machine's setup cost for locomotion between jobs in a sequence 1| | Total weighted waiting time +Total Setup cost.

The rest of the article is formed as follows:

Section 2 provides some notations and definitions then contains an integer nonlinear programming formulation of the problem. Section 3 contains discussion about a simple test problem. Section 4 introduces a genetic algorithm for the problem presented. The reported results of computational experiments to measure the efficiency of algorithms are introduced in Section 5. Eventually in Section 6 conclusion and some directions for future research are presented.

II. DEFINITIONS & NOTATIONS

In this paper a CNC(Computer Numerical Control) machine is considered that must process some of the jobs (N jobs) that belong to M customers such that sequence obtained must minimize total cost (setup cost & waiting time cost) such that, in this paper, customer's waiting time is considered instead of job's waiting time, namely waiting time is calculated for any customer, instead of each job in the 1||WWTV problem (Weighted Waiting Time Variance) Li et al.[11]are presented this problem with objective function is following:

$$Min: \sum(V_j \; (w_j - w_{avr})\text{^}2)\text{^}1/2$$

But in this paper, each customer has some of the job specified. Our objective is minimization of total customer waiting time cost and total machine's setup cost for locomotion between jobs in a sequence where objective function is the following:

$$Min = \sum_{j=1}^{M} v_j * w_j + \sum_{k=1}^{N-1} c_k$$

This paper is under the following assumption:

- All of the jobs are simultaneously available at time zero and job preemption is not allowed.
- The machine could not be idle when at least one non-assigned job exists.
- There is machine's set-up time for processing from one job to next job.

There are *n*! feasible job sequences, it was proved that 1||WTV problem is NP-hard by Elion and Chowdhury [17] then is concluded that this problem is NP-hard. Some descriptions refer to the indices, parameters and here, notations, parameters and decision variables are reported;

A. ***Index:***

$i = 1, 2 \ldots n$ Job index
$j = 1, 2 \ldots m$ Customer index
$k = 1, 2 \ldots n$ Job position index

B. ***Parameters:***

p_i=processing time job *i*
$\{M_j\}$ =set of jobs belong to customer *j*
v_j=cost for unit waiting time of customer *j*
Matrix machine's setup cost sequence dependent for jobs

C. ***Decision variable:***

x_{ijk}=1 if job *i* of customer *j* in the position job *k'th* be processed otherwise=0.
w_j=waiting time for customer *j'th*
e_j=last position's job for customer *j*
c_k=machine's setup cost from position *k* to the *k+1*

D. ***Integer Non Linear Programming (INLP)***

$$Min = \sum_{j=1}^{M} v_j * w_j + \sum_{k=1}^{N-1} c_k$$

$$St:$$

$$\sum_{i=1}^{N} \sum_{j=1}^{M} x_{ijk} = 1 \ \forall k$$

$$\sum_{k=1}^{N} \sum_{j=1}^{M} x_{ijk} = 1 \ \forall i$$

$$e_j = max_k\{\sum_{i=1}^{n} k * x_{ijk}\}\ \forall j$$

$$w_j = \sum_{k=1}^{e_j} \sum_{j' \neq j}^{m} \sum_{i=1}^{n} x_{ij'k} * p_i\ \forall j$$

$$c_k = \sum_{i=1}^{n} \sum_{i'=1}^{n} \sum_{j=1}^{m} \sum_{j'=1}^{m} x_{ijk} * x_{i'j'k+1} * set\ up(i,i')\ \forall k = 1,2,\dots N-1$$

$$x_{ijk} = 0 \text{ or } 1$$

Row 1: objective is total setup cost plus total customer waiting time cost.
Row 2: this constraint expresses that only one of the jobs is processed at one position.
Row 3: this constraint presents that each of the jobs of the customer is processed in a position (no preemption)
Row 4: this constraint specifies last position job for customer j'th.
Row 5: this constraint calculates waiting time for customer j'th, namely time that customer j needs to the machine but machine is not available.
Row 6: this constraint calculates total setup cost from job one to the last job in a sequence.
Row 7: binary variables

III. EXAMPLE

Since this problem is NP-hard, we solved a small size example by LINGO. Here, we considered an example; with six jobs and two customers such that each job belongs to a specified customer. Processing time for each job is specified and which is available at time zero, there are setup cost sequence dependent jobs in TABLE I

$$Job = \{1,2,3,4,5,6\}\ N = 6$$

$$Pi = \{3,2,2,4,1,2\}$$

$$Customer = \{1,2\}\ M = 2$$

$$M1 = \{1,2,3\}\ M2 = \{4,5,6\}$$

TABLE I MACHINE'S SETUP COST DEPENDENT SEQUENCE JOBS

Setup cost	**1**	**2**	**3**	**4**	**5**	**6**
1	0	10	15	10	2	1
2	10	0	15	5	3	1
3	15	15	0	1	5	2
4	10	5	1	0	15	16
5	2	3	5	15	0	14
6	1	1	2	16	14	0

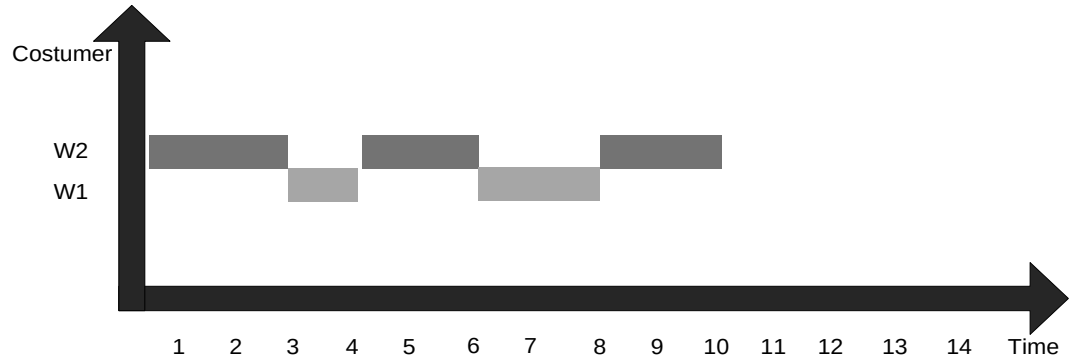


Fig.4 The Gantt chart for optimal sequence (1-5-2-6-3-4)

In the above Gantt chart (Fig.4), waiting time cost for customer 1' equals five (W_1=3), indeed this time is while machine process job5 and job6,or waiting time customer 2'th equal eleven (W_2=7),equal is process time jobs:1,2,3. Optimal solution of given example is obtained by mathematical model in lingo 9 commercial and it is reported in TABLE II. (Run time =1':05")

TABLE II RESULT OF EXAMPLE BY LINGO

X_{ijk}	K	1		2		3		4		5		6	
I	J	1	2	1	2	1	2	1	2	1	2	1	2
1		1	0	0	0	0	0	0	0	0	0	0	0
2		0	0	0	0	1	0	0	0	0	0	0	0
3		0	0	0	0	0	0	0	0	1	0	0	0
4		0	0	0	0	0	0	0	0	0	0	0	1
5		0	0	0	1	0	0	0	0	0	0	0	0
6		0	0	0	0	0	0	0	1	0	0	0	0
C_k		2		3		1		2		1		0	

Optimal sequence jobs: 1-5-2-6-3-4.

IV. GENETIC ALGORITHM

In this section, we present in detail the different components of the genetic algorithm implemented in this paper to solve our problem. Genetic algorithms (GAs) have been used successfully to find optimal or near-optimal solutions for a wide variety of optimization problems since its introduction by Holland [5]. GA starts with an initial set of solutions, called Initial Population. In the population, each solution is called a chromosome (individual), the chromosomes are evolved through successive iterations, called Generations, by genetic operators (elitism, crossover and mutation) that mimic the principles of natural evolution([13], [15]). In a GA, a fitness value is assigned to

each individual according to a problem- specific objective function. Since, this problem is NP-hard and the size of a real problem is very large, a GA is proposed for solving large-size problems in a reasonable computational time, commonly; any genetic algorithm is consists of five basic components (Michaelwicz, [20]).
The genetic algorithm presented in this paper is in tune with the genetic algorithms mentioned generally. Fig.5 demonstrates the construction of the proposed genetic algorithm whose components are considered in detail in the following subsections.

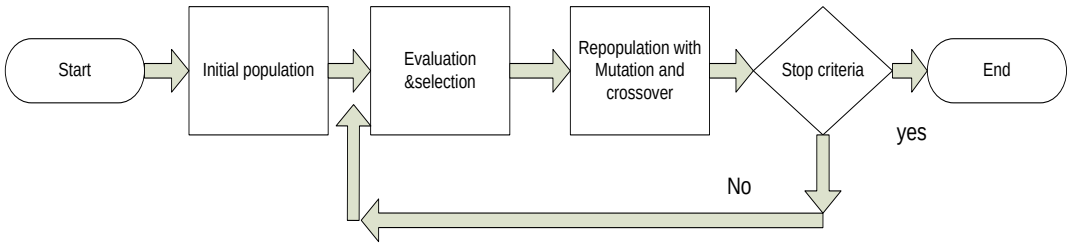


Fig.5 Flow chart of the proposed GA

A. ***Representation of solutions***

Here, we use random key representation scheme proposed by James and Bean [2] to generate initial population at the *population size*, proposed chromosome contains a number of genes equal to number of jobs, each of the genes value is placed as random number between [0, 1].

Fig. 6 presents an example as a chromosome (coding) and then the decoding of a hypothetical example with six jobs and two costumers.

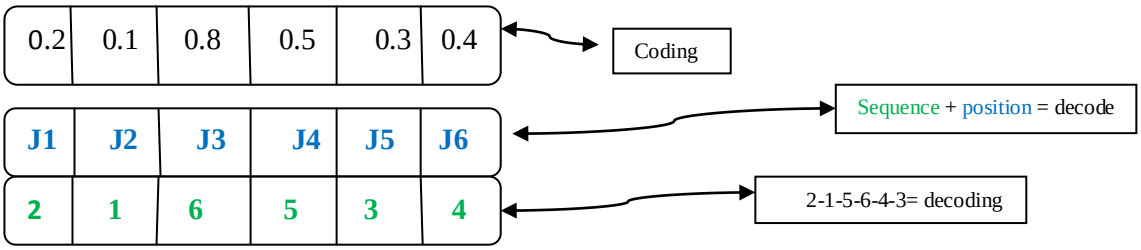


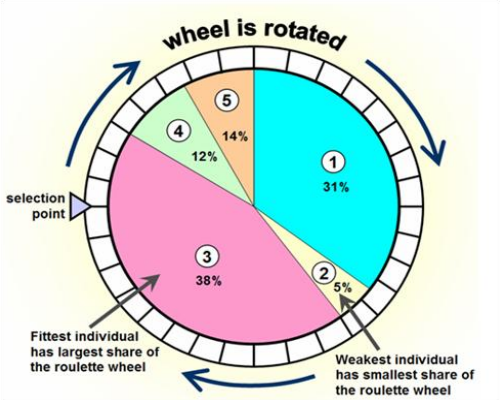
Fig. 6. Encoding & decoding

B. ***Evaluation & selection***

In order to mimic the natural process of survival of the fittest, the fitness evaluation function assigns to each member of the population a value reflecting their relative superiority. In our problem, solutions with lower costs are preferable. Therefore, the objective function is inverted as following, in other words, a chromosome with lower objective function value is worthier.

$$fitness = \frac{1}{objective\ value}$$

Selection strategy is based on roulette wheel that fitness of each chromosome is determined according to the size of its segment on the roulette wheel; a structure of roulette wheel is shown in Fig.7.



No.	Fitness	% Of Total
1	6.82	31
2	1.11	5
3	8.48	38
4	2.57	12
5	3.08	14
Total	22.05	100

Fig. 7. Structure of roulette wheel

C. ***GENETIC OPERATORS***

After calculation of the fitness value, two parents are selected with Roulette wheel mechanism, and then operations of production for offspring are applied as following:

1) *Summation Crossover:* In this operator, summation of value's genes corresponding parents together until it produces one offspring. That is shown in Fig.8.

0.4	0.1	0.5	0.6	0.2	0.3
0.5	0.1	0.2	0.3	0.8	0.7
0.9	0.2	0.7	0.9	1	1

Parents

Summation crossover

J1	J2	J3	J4	J5	J6
4	1	2	3	5	6

Decode =2-3-4-1-5-6
Offspring

Fig.8. Summation crossover

2) *Cutline crossover:* The second crossover is cutline crossover whose integer number is randomly generated from 1 to *N*-1 to specify cut point. The created offspring from cutline crossover operator per each gen before cut point inherits from parent 1; otherwise, it inherits from parent 2 (Zandie and Rashidi, [9] which is shown in Fig.9.

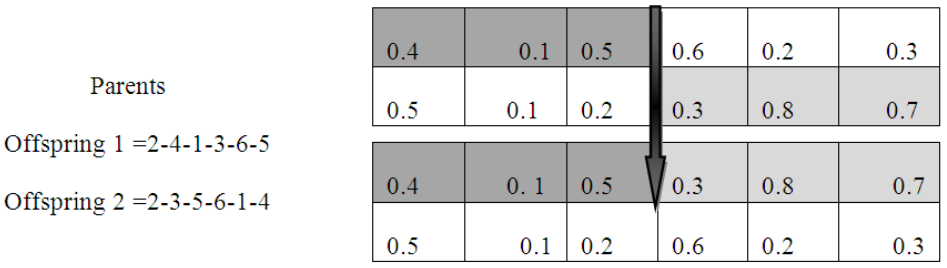


Fig. 9. Cutline crossover operator

3) *Mutation & elitism operator:* To prevent the solution sequence to converge to local optimum, the mutation operator must be used to diversify in search space. Mutation brings accidental properties to the children that do not exist in parents. There are several mutation operators such as swapping, inversion, insertion and shift mutation (Torabi et al. [16]. Swapping mutation is applied in this paper and the best solution in each generation is transferred to the next as Elitism. Fig.10 illustrates it graphically.

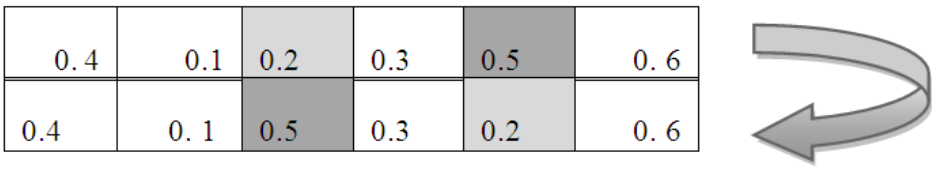


Fig. 10. mutation operator

4) Adjustment of parameters & termination condition: The proposed algorithm uses a predetermined value is tuned with design of experiments as: number of generation, population size, crossover rate and mutation rate. The algorithm finishes if no improvement is achieved for the *max_no_improve*. TABLE III shows parameters of GA

TABLE III. GA Parameters

Parameter	**value**
Population size	250
Max-gen	150
Max-no-improve	30
Crossover rate	0.85
Mutation rate	0.05
Elitism rate	0.1

V. COMPUTATIONAL EXPERIMENTS

To evaluate the performances of INLP and the Genetic Algorithm, the efficiency of the algorithm is measured with using a number of randomly produced problems in small, medium and large size (Vikas and Anshu, 2012). Since the difficulty of the problem depends on the number of jobs (*N*) and the number of customers (*M*), two parameters are controlled in several levels, in which *N* can be one of five values (5, 6, 10, 20 and 30) and *M* can be one of five values (2, 3, 5, 8 and 10). Then test problems are solved with INLP and GA. Now, the performance of genetic algorithm is reported on the test problems with predetermined parameters, in the tables 4-6:

TABLE IV SMALL SIZE PROBLEMS

Row	**N**	**M**	**Time INLP**	**Objective INLP**	**Best objective GA**	**Averae objective GA**	**Average time GA**	**GAP %**
1	5	2	0:20''	24	24	24	0:1.1''	0
2	5	3	0:35''	42	42	42	0:1''	0
3	5	5	1':45''	65	65	65	0:1.2''	0
4	6	2	0:40''	24	24	24	0:4''	0

TABLE V MEDIUM SIZE PROBLEMS

Row	**N**	**M**	**Time INLP**	**Objective INLP**	**Best objective GA**	**Average objective GA**	**Average time GA**	**GAP %**
5	6	3	3':00''	32	32	32	0:4''	0
6	6	5	8':30''	46	46	46	0:5''	0
7	10	2	1800^{+}	---	70	70	0:4.56''	0
8	10	3	1800^{+}	---	129	129	0:4.92''	0
9	10	5	1800^{+}	---	228	228	0:5.19''	0
10	10	10	1800^{+}	---	293	293	0:5.76''	0

TABLE VI LARGE SIZE PROBLEMS

Row	N	M	Time INLP	Objective INLP	Best objective GA	Average objective GA	Average time GA	GAP %
11	20	2	1800^{+}	---	135	136	0:17.14''	0.7
12	20	3	1800^{+}	---	277	278.8	0:17.06''	0.6
13	20	5	1800^{+}	---	541	542.5	0:18.31''	0.2
14	20	8	1800^{+}	---	814	819	0:50.76''	0.6
15	20	10	1800^{+}	---	1142	1144.3	0:53.2''	0.2
16	30	2	1800^{+}	---	224	226.1	0:50.55''	0.9
17	30	3	1800^{+}	---	632	647.7	0:51.86''	2.4
18	30	5	1800^{+}	---	1145	1167.2	0:51.86''	1.9
19	30	8	1800^{+}	---	1172	1177	0:55.55''	0.4
20	30	10	1800^{+}	---	2147	2172.8	0:57.88''	1.2

From tables 4 until 6 the efficiency of genetic algorithm is inferred for different size problems to obtain optimal solution for small size problems and near optimal solution with little GAP for large size problems in reasonable time. Now, genetic algorithm and INLP are compared in the CPU time for all the test problems in the Fig.11.

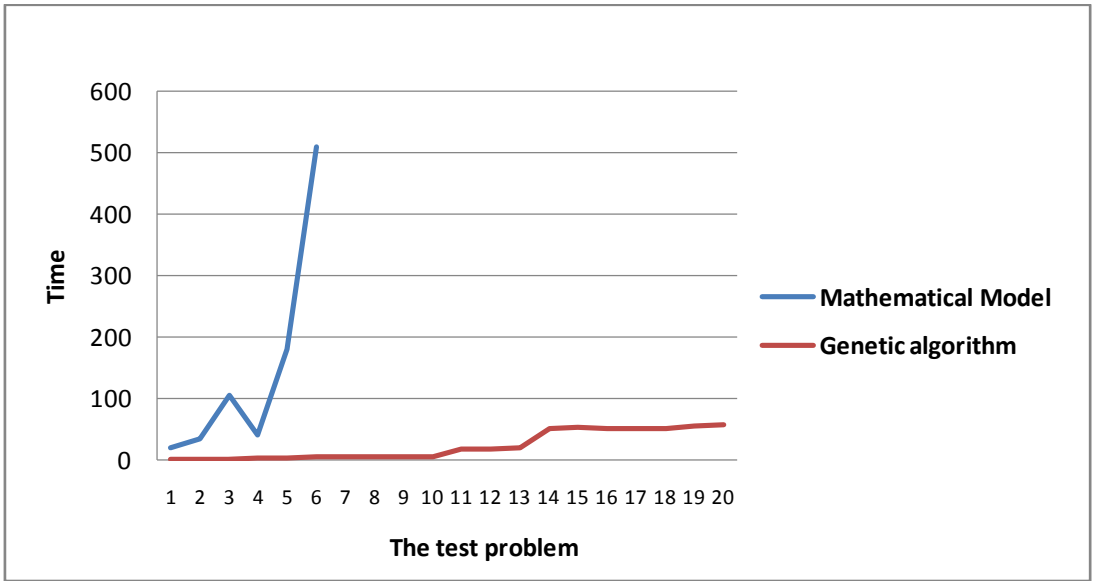


Fig.11. CPU time of GA and INLP

In the Fig.12, the best and average objective value of GA are presented with green line and red line for samples respectively and blue line shows objective value of

INLP. it is clear that GA obtains identical solution with INLP for small size test problem. Distance between green and red lines shows gap of GA which is very small.

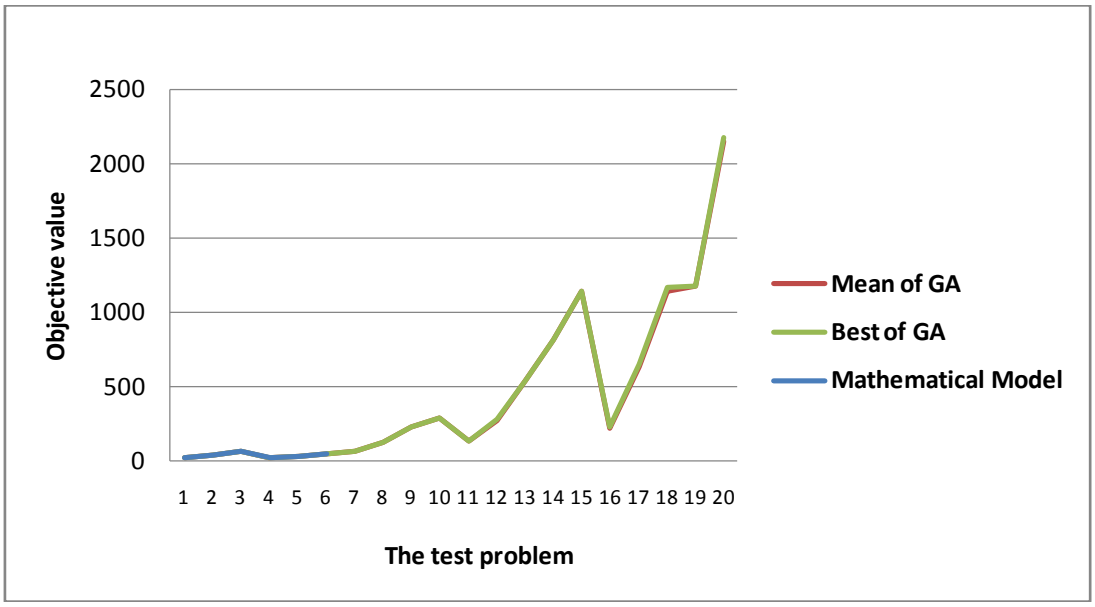


Fig.12. Objective value GA and INLP

VI. CONCLUSIONS AND FUTURE DIRECTIONS

In this study, minimization of total customer's waiting time cost and setup cost of sequence dependent jobs is presented on a single processor, Mathematical modeling (INLP) is considered to solve small and medium size problems with lingo 9. Since, this problem is NP-hard and the size of a real problem is very large, a genetic algorithm is proposed for solving large-size problems in a reasonable computational time and the efficiency in the performance of genetic algorithm is shown with small, medium and large size problems randomly. For future investigation, we look forward to develop an approach to the case of scheduling with dynamic job arrivals with preemption and probability of machine failure will be allowed in other shop systems e.g. flow shop and parallel machines. Develop heuristics for minimizing customer's waiting time on a machine, effective lower bounds for comparison of algorithms and Multi- objective evaluation in this area.

REFERENCES

[1] A. G. Merten and M. E. Muller, Variance minimization in single machine sequencing problems., *Management Science*, vol. 18, pp. 518–528, 1972.

[2] C. James, and A. Bean, *Genetic Algorithms and Random Keys for Sequencing and Optimization*, ORSA Journal on Computing Spring, 6:154-160, 1994.

[3] G. Vikas and P. Anshu, Comparison of processor scheduling algorithms using Genetic Approach. International, *Journal of Advanced Research in Computer Science and Software Engineering*, vol.2, no. 8, 2012.

[4] G. Viswanathkumar, G. Srinivasan, A branch and bound algorithm to minimize completion time variance on a single processor, *Computers and Operations Research*, vol. 8,no.30, pp.1135–50, 2003.

[5] J. Holland, Adaptation in Natural and Artificial Systems',book, seconded, *University of Michigan Press*, 1992.

[6] J. Mittenthal, M. Raghavachari and A. I. Rana, V- and ^-shaped properties for optimal single machine schedules for a class of non-seqarable penalty functions, *European Journal of Operations Research*, vol. 86, pp. 262–269, 1995.

[7] K. Gowrishankar, R. Chandrasekharan and G. Srinvivasan, Flow shop scheduling algorithms for minimizing the completion time variance and the sum of squares of completion time deviations from a common due date, *European journal of operational research*, vol. 132, pp. 643-665, 2001.

[8] K. R. Baker and G. D. Scudder, Sequencing with earliness and tardiness penalties: *A review. Operations Research*, vol. 38, pp. 22–36, 1990.

[9] M. Zandieh, E. Rashidi, An Effective Hybrid Genetic Algorithm for Hybrid Flow Shops with Sequence Dependent Setup Times and Processor Blocking, *Journal of Industrial Engineering*, vol. 4, pp. 51- 58, 2009.

[10] N. Ye and X. Xu, Minimization of job waiting time variance on identical parallel machines. *IEEE Transactions on Systems*, Man, and Cybernetics, Part C: Applications and Reviews, vol. 37, pp. 917–927, 2007.

[11] N. Ye, X. Li, T. Farley and X. Xu, Job scheduling methods for reducing waiting time variance, *Computers and Operations Research*, vol. 34, pp. 3069–3083, 2007.

[12] N. Tyagi. R. G. Varshney and A. B. Chanramouli, Six Decades of Flowshop Scheduling Research, International Journal of Scientific & Engineering Research, vol. 4, issue 9, September-2013

[13] N. Tyagi. R. G. Varshney and A. B. Chanramouli, A Comprehensive Study of Genetic Algorithm for flowshop scheduling research, International Journal of Scientific & Engineering Research, vol. 4, Issue 6, June 2013.

[14] N. Tyagi, M. Abedi and R. G. Varshney, Single Machine Scheduling in a Batch Delivery System with Fixed Delivery Dates, International Journal of Engineering and Technology (IJET), Vol 5 No 4 Aug-Sep 2013

[15] N. Tyagi and R. G. Varsheny, A Model To Study Genetic Algorithm For The Flowshop Scheduling Problems, Journal of Information and Operations Management, vol. 3, issue 1, pp.38-42, 2012.

[16] S. A. Torabi, S. M. T. Fatemi Ghomi and B. Karimi, *A hybrid genetic algorithm for the finite horizon economic lot and delivery scheduling in supply chains'* European Journal of Operational Research, 173:173–189, 2006.

[17] S. Eilon and I. G.Chowdhury, Minimizing waiting time variance in the single machine problem. *Management Science*, vol. 32, pp. 65-80, 1977.

[18] U. Bagchi, Simultaneous minimization of mean and variation of flow time and waiting time in single machine systems. *Operations Research*, vol. 37, pp. 118–125, 1989.

[19] X. Li, N. Ye and T. Liu,Y, Sun, Job scheduling to minimize the weighted waiting time variance of jobs, *Computers & Industrial Engineering*, vol. 52, pp. 41–56, 2007.

[20] Z. Michalewicz, Genetic algorithm + data structure = evolution programs (3rd ed.), *New York*, 1996.