International Journal of Engineering Innovation & Research
Volume 4, Issue 1, ISSN: 2277 – 5668

A Mixed-Integer Model With Genetic Algorithm for Multi-Objective Assembly Line Balancing Problem in Fuzzy Manufacturing Environment

Ramin Behbamzadeh
Faculty of Industrial Engg, Iran University of Science and Technology, Tehran, Iran
email: RaminBehbamzadeh@gmail.com

Mohammad Alaghebandha
Department of Industrial Engineering, Kharazmi University, Karaj, Iran
email: m.alaghebandha@gmail.com

Amir Azizi
Faculty of Manufacturing Engineering, University Malaysia Pahang, Malaysia
email: amirazizi@ump.edu.my

Abstract: **In this paper, a multi-objective assembly line balancing problem (ALBP) is considered by assuming the fuzzy process time. The objective is minimizing a weighted sum of two contrast and classical criteria in literature as cycle time and the number of stations. A mixed-integer model is proposed for solving this problem. As the ALBP is NP-hard problem, genetic algorithm (GA) with a novel representation is proposed for large-scaled problem. A number of small-sized problems were considered to optimally solve in order to verify the performance of the proposed model. The obtained results present that the efficiency of the proposed model and the ability of developed GA to achieve optimal solution in reasonable time.**

Keywords: **Assembly Line Balancing, Fuzzy, Genetic Algorithm, Mixed-Integer Programming**

1. Introduction

The history of assembly line is for more than 75 years. It means after the advent of the Ford system in assembly line, which is generally discussed in massive production, the necessary steps for production are assigned to work stations according to hypothesis to minimize the cycle time or work stations number. In recent years [1,2], the division has conducted several assembly line balancing problem that can be classified according to the objective function, problem structure and timing of jobs. In categorizing assembly line problems according to objective function, two basic models are mentioned, that different model can be generated by composition of these two models. These models include: Models of Type I and Type II. In Type I the cycle time of assembly line as the input of problem is defined and the steps to assemble the product should be assigned to workstations in such a way to minimize the required work stations. The assembly line balancing problem of Type II is also one of the categorizing of assembly line problems according to objective function, in that problem the number of assembly workstation as is designed the input of problem and the steps to assemble the product should be assigned to workstation in order to minimize the cycle time.The primary solving method of assembly line balancing problem is the linear programming, which is used by Salveson [3] for solving the assembly line balancing problem in simplest situation and after that Bowman [4], White [5] and Baker [6] solve the problem by formulating it in form of an integer linear programming, and also people like Patterson & Albrecht [7] used dynamic programming to solve this problem. Fonseca et al. [8] proposed a work to model and solve the stochastic assembly line balancing problem with a fuzzy representation of the time variables as a viable alternative method.

The assembly line balancing problem is one of the optimization problems for which is not possible to find the optimized result in an acceptable time. So most of the presented solutions are not efficient and proper for small size problems. Therefore to solve these problems creative solutions like Generic Algorithm, Tabu Search and Simulated Annealing are presented. One of the successful metaheuristic methods is Generic Algorithm, which is used in lots of complex optimization problems with acceptable results, high quality and effective time. In recent years, the Genetic Algorithm is used to solve assembly line balancing problem [9]-[14].

This paper proposes a metaheuristic model to solve a combined problem considering fuzzy job processing time in assembly line balancing to reduce the cycle time and the work station number using Genetic Algorithm.

2. Fuzzy Set Theory

Since data in real-world problems are often afflicted with uncertainty, imprecision and vagueness due to both machine and human factors, they can only be estimated as within uncertainty. In an attempt to treat imprecise data, fuzzy numbers are introduced to represent the processing time of each job, where the membership function of a fuzzy data represents the grade of satisfaction of a decision maker [15]-[17]. Therefore Fuzzy Set Theory is a proper tool for modeling the uncertainly equals to Imprecision, ambiguity and loss of information and it is better tool for solving imprecise programming problems as below:

$$\begin{aligned} &Min\ \ \tilde{Z} = \tilde{f}(\tilde{x}) \\ &S.t\ \ \ \ \tilde{x} \in X \end{aligned} \qquad (1)$$

It seems problems with long-term prediction may include incomplete and imprecise information and also decisions are made according to the expert's competence and are subjective, so it is very appropriate to use fuzzy number instead of definite numbers. The triangular numbers are very appropriate for this goal because they are created by defining the smallest, biggest and the more acceptable numbers. The analyses are based on fuzzy average instead of definite average.

A. Triangular fuzzy numbers

Triangular fuzzy number A with membership function $\mu_A(x)$ is defined as follow:

$$A \triangleq \mu_A(x) = \begin{cases} \dfrac{x-a_1}{a_M-a_1} & a_1 \le x \le a_M \\ \dfrac{x-a_2}{a_M-a_2} & a_M \le x \le a_2 \\ 0 & \text{otherwise} \end{cases} \qquad (2)$$

B. Defuzzification of fuzzy mean

Defuzzification is the process of producing a quantifiable result in fuzzy logic, given fuzzy sets and corresponding membership degrees. It is typically needed in fuzzy control systems.

In this paper, "Greatest Associate Ordinary Number" method is used for defuzzification as follow:

$$A_{ave} = (a_1, a_M, a_2)$$
$$x_{\max}^{(2)} = \frac{a_1 + 2a_M + a_2}{4} \qquad (3)$$

3. Problem Definition and Formulation

The considered problem is a two objective assembly line balancing problem for minimizing the cycle time and workstations simultaneously while jobs processing time are fuzzy. The goal is to present a Fuzzy linear Programming (FLP) in such a way that the objective function is in form of crisp (none fuzzy) but the Feasible space of the problem is fuzzy, so the final result is fuzzy.

The model parametersare definedas follows:

N= number of jobs

M= maximum number of work station

$\tilde{P}_i$=fuzzy processing times of jobi

p_{ij}=1 If job i is processed before job j, otherwise is 0;

$\tilde{C}$= cycle time

$\tilde{t}_k$= Workloadimposed onthestation k

$F(\tilde{X})$= crispvalue of thefuzzy number$\tilde{X}$

w_1 = Coefficient ofthe first componentof the objective function

w_2= Coefficient ofthe first componentof the objective function; $w_1+w_2 = 1$

x_{ki} = 1 if job i allocate to station k otherwise is 0

x_{lj} = 1 if job j allocate to station l otherwise is 0

The mathematical formulation of the fuzzy ALBP is given as follows:

$$\text{Min } Z = w_1 \times F(\tilde{C}) + w_2 \times F\left(\sum_{k=1}^{M}\sum_{i=1}^{N}\left(\tilde{C} - \tilde{t}_k\}\right)x_{ki}\right) \qquad (4)$$

s.t:

$$\sum_{k=1}^{M} x_{ki} = 1 \qquad \forall i \qquad (5)$$

$$\sum_{k=1}^{M} kx_{ki} - \sum_{l=1}^{M} lx_{lj} \ge 0 \qquad \forall i,j \;;\; p_{ij=1} \qquad (6)$$

$$\tilde{t}_k = \sum_{i=1}^{N} \tilde{p}_i x_{ki} \qquad \forall k \qquad (7)$$

$$\tilde{C} \gtrsim \tilde{t}_k \qquad \forall k \qquad (8)$$

$$x_{ki}, x_{lj} \in \{0,1\} \qquad (9)$$

The objective function(4) is the minimization of the total weighted defuzzification cycle time and cycle time standard deviation workloads stations. The second component, the model would have to use a smaller number of stations so that the stations at least once a process is homogeneous and close to the optimum cycle time.

Constraint(5) ensures that each task is assigned to only one station. Constraints(6) respecting the precedence relations between the tasks are guaranteed. Constraints (7) and (8) are the problem of fuzzy constraints. Constraint(7) Fuzzy processing load imposed by each station determines and constraint(8) the cycle time specifies the optimal fuzzy.

4. Genetic Algorithm

GA is a search metaheuristic that mimics the process of natural evolution was introduced by Holland [18]. Many researchers applied and expanded this algorithm in different fields of study [19, 20]. The steps that involved in the GA algorithm are briefly presented as follows:

(1) Setting GA parameters including the crossover probability, the mutation probability, population size and number of generation,

(2) Initializing the population with the size of population size randomly;

(3) Evaluating the objective function;

(4) Choosing individual for mating pool by roulette wheel selection;

(5) Implementing the continues crossover operation for each pair of chromosomes based on crossover probability;

(6) Implementing mutation operation for each chromo some based on mutation probability;

(7) Replacing the current population by the resulting new population;

(8) If stopping criteria is met, then stop. Otherwise, go to Step 3.

A. Proposed Algorithm

The proposed algorithm with aninitial population(generation zero), which contains various solutions of the problem begins. Then each population by applying genetic operations, changes and other solutions are produced. In each generation, the fitness value of the solutions calculated in terms of the selection process, individuals are chosen for the next generation.

The task of creating new solutions and investigate new areas of the solution space of genetic operators. These operators include cross over operator and mutation operator. The task of these operators produce off spring(new parts) of the parent(the previously area studied) so that children can inherit parental characteristics. The transfer characteristics(inheritance) are such that the average quality of the population(Answer) generation increases. Similarly the evolution process continues until as topping criterion is satisfied and the algorithm will terminate.

Performance of a genetic algorithm depends on the following factors: how to show problem's solutions, the initial population, selection, selection of operators and parameters. Then genetic algorithm is proposed to solve the problem of explaining how the various parts of its features will be explained.

A.1 Chromosome Structure

To display the structure of the N gene, which is used to answer a series of N represents the number of jobs.

The expression of gene i, shows the work station that job i is assigned to it. In other words, each gene can receive values in the interval [1, M].

To observe precedence relationships and feasible chromosome is needed to keep the constraint(6), if job j precedes job i is on, then the value of gene i, is smaller than the value of the j-th gene. In other words, job i is processed before job j, making the assumption that the work stations are numbered in ascending order.

Figure 1 is a feasible solution with 6 jobs and 3 work stations that shows the precedence relations matching network shown in Figure 2 follows. Figure 3 as an infeasible solution network Figure 2 shows. As 2 above 5, but 5 should be processed before 2.

1	1	2	2	2	3

Fig.1. Feasible Solution

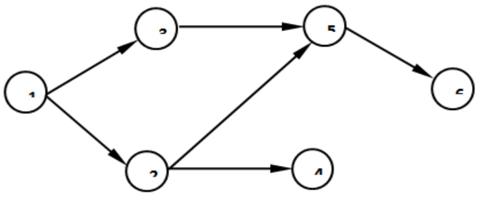


Fig.2. Network Precedence

1	2	2	2	1	3

Fig.3.Infeasible Solution

A.2 Produce feasible solution for initial population

Required to produce the initial population of initial solutions is generated. The innovative approach is used to generate feasible solutions.

The main idea of this method is that each job must be assigned to the station with the largest number of allocated among the stations that jobs prior to his are the stations that or, at most to the next station.

The approach used is as follows:

Step 1 - The first job to assign to the first station.

Step 2 - Place a work counter, i = 2.

Step 3 - Let A_i be the set of all the jobs precedes job i.

Then Max A_i calculated according to:

$$MaxA_i = \max_{j \in A_i}\{s_j\} \tag{10}$$

Step 4 - If Max A_i = M then assign job i to the final station M, otherwise assign job i with a certain probability to Max A_i or Max A_{i+1} station.

Step 5 - Set i = i +1. If i = N +1 stop else go to step3.

A. 3 Mutation operator

To maintain genetic diversity in the population using the mutation operator is required. Throughout the design of genetic algorithm, we try to maintain feasibility and not using the repair operator in the genetic operators employed in such a way that the resulting solution will remain feasible. For this, at first, a randomly selected to job like i. Let A_i as the set of all of the jobs prior to the job i and B_i is all the jobs that job i have preceded them, and s_j is equal to the work station that job j in the current chromosome is allocated; now we define the following values:

$$MaxA_i = \max_{j \in A_i}\{s_j\} \tag{11}$$

$$MinB_i = \min_{j \in B_i}\{s_j\} \tag{12}$$

For doing mutation, we choose job i that Min $B_i \neq$ Max A_i. Then the work station assigned to job i in the range[Max A_i, Min B_i] will change. It ensures feasibility of solution. If you do not find job with the above conditions, mutation will not carry out.

A. 4 Crossover Operator

Crossover operator is fundamental GA operator which its design and application is very important. In order to effective use of Crossover operator, and having feasible solution, max and min operation has been used here. These operation has been done on selected parents from current population, and cause to generate a new feasible offspring. Instruction of how to use these operator Ss is described as follows.

Assume that chromosomes in Fig. 4 are two selected parents from current population according to previous example.

1	2	1	3	3	4

1	1	2	2	2	3

Fig.4. Chromosomes of two Selected Parents

After application of max and min operators, new offspring will be produced as follows (Fig. 5 and 6, respectively).

1	2	2	3	3	4

Fig.5. New Offspring AfterMax Operator

1	1	1	2	2	3

Fig.6. New Offspring After Min Operator

Application rate of max and min operator have to be almost equal, as the usage of only one of the above mentioned operators leads to divergence of future generation, because in this case all the jobs will assign only to station 1 or only to last station (m).

A.5 How to select parent population

For selection of parent population from current generation, common method of roulette Wheel has been used. In this method probability of selection of one chromosome depends on its fitness magnitude. Also, in order to assess quality of generation's solution in terms of diversity and evolution (continues movement toward optimal solution) mean and variance of each generation will be calculated as the following relations:

$$z_i^g = \frac{f_i^g - \mu_g}{\delta_g} \in [-1,1] \tag{13}$$

$$\delta_g = \left(\frac{\sum_{i=1}^{K} \left(f_i^{\,g} - \mu_g \right)}{K} \right)^{\frac{1}{2}} \quad (14)$$

Mean of each generation could be a proper criterion to assess degree of evolution (continuous movement toward optimal solution) of current generation rather than previous generation. Variance of each generation demonstrates degree of diversity (heterogeneity of chromosomes). In addition, Variance of generation has been used as a criterion for stopping algorithm.

A.6 Stopping Criteria

The GA will terminate when either one of the following stopping criteria is met:

-Maximum generations

-Minimum allowable variance of generation

-Maximum time of Algorithm

B. Steps of proposed genetic algorithm

Step0- Initialize first parameters as follows

K: Population quantity in each generation

G: Maximum allowable number of generations

δ: Minimum allowable variance for each generation

r_i: Selection rate of operator type i

G: counter generation g = 1,2,…,G

Step1- set g=1 and generate Initial population

Step2-Select parent population using roulette Wheel method

Step3-Generate k offspring's and substitute in new generation by applying crossover and mutation operators on selected parent's population chromosomes.

Equal g=g+1 and calculate mean and variance of new generation. If one of the criteria of part 5-6 has been satisfied, stop and send the best last generation solution to the output, otherwise go to step2.

5. Results and Discussions

In this section comparison between results of designed genetic algorithm, and the results of Lingo 8 software solution are carried out. At the end complete results of solving one example is represented.

For obtaining best existing weighting in model (w_1, w_2), first one problem with 12 jobs and 4 stations has been solved with LINGO 8 and then compared with designed genetic algorithm that problem 20 times is performed.

After recording the result in Table 1 and their comparison, best weight has been found in order to use in next implementations. It is necessary to mention that weight is severely affected by management idea, because designed model is a type of multi-objective models, and by implementation of different weights we can actuate the results toward reducing cycle time, or number of stations, or simultaneously either number of stations or cycle time.

Here is also there is much attention to rationality of results for weights as for some results, irrationality of them in term of assigning number of stations is completely obvious and has not been considered in decisions.

Best value for objective function with implementing of Lingo is 22.125. According to outcomes (Table 1) it is observed that best value for objective function with weights (w_1=0.3, w_2=0.7) is 18.525, but assigning 2 stations for 12 jobs has made this weight to be unacceptable.

Acceptable weights are (w_1=0.6, w_2=0.4) and this weight has been considered as a base weight for implementing of algorithm in the next steps.After weighing determination, designed algorithm has been implemented for different problems with different jobs and different stations, and the results have been analyzed.

At first for considered problems, number of stations is assumed constant and in every step number of jobs is increased and then the results of their solution is compared with the results of Lingo solution.

For the rest of problems in number of stations and different jobs are analyzed. In some cases, CPU time for solution by Lingo is too much; so an agreement has been considered as follows.

For problems in which their CPU time last more than one hour, a lower bound is calculated as follow:

$$LB = \frac{\sum_{i=1}^{N} \tilde{t}_i}{\tilde{C}} \qquad M \quad (15)$$

M: maximum allowable number of station

N: number of jobs

It is necessary to mention that probability of implementing of crossover operation (Pc) and mutation operation (Pm) in designed genetic algorithm is considered 0.5 and 0.5 respectively after different implementation of algorithm.

The results of different implementation algorithm are summarized in Table 1. Considered example in different primary populations is solved 10 times with genetic algorithm and the best answers are mentioned in Table 2.

For more complete explanation, the result of last problem, which consists of 80 jobs and 10 stations, is presented completely in Table 5.

Fuzzy process time of jobs and prerequisites for the considered problem are presented in Table 3 and Table 4.

Best answer for the problem in primary population of 1000 and with objective function of 90, with the time of 30 minutes and 57 seconds is obtained. Finally, assigning jobs to 4 stations is described as follows and presented in Table 5.

Station 1:
1,2,4,6,9,10,12,13,17,23,27,30,32,33,37,42,58,59,77

Station 2:
3,5,8,14,15,16,18,19,21,24,25,26,29,34,35,40,41,45,48,49,60

Station 3:
7,11,20,22,28,31,36,38,39,43,47,50,52,56,62,64,67,68,72

Station 4:
44,46,51,53,54,55,57,61,63,65,66,69,70,71,73,74,75,76,78,79,80

It is necessary to mention that the obtained means and variances for the problem and also their charts are presented in Table 6, Figure 7 and 8 respectively.

International Journal of Engineering Innovation & Research
Volume 4, Issue 1, ISSN: 2277 – 5668

Table 1. Results of Each Iteration in GA Related to Different Weights

Weight (W_1, W_2)		0.5,0.5	0.8,0.2	0.6,0.4	0.4,0.6	0.3,0.7
Best objective function in each iteration	1	25.12	29.3	26.1	23.6	18.52
	2	24.5	29.3	26.1	22.9	18.65
	3	24.5	29.3	26.1	22.9	18.52
	4	27.62	29.3	26.6	22.9	18.52
	5	24.5	29.3	26.1	22.9	18.52
	6	24.5	30.8	26.1	22.9	18.52
	7	24.5	29.3	26.1	22.9	18.52
	8	24.5	29.3	26.1	22.9	18.52
	9	24.5	29.3	26.1	22.9	18.52
	10	25.12	30.1	26.1	22.9	18.52
Number of allocate stations		4	4	4	4	2

Table 2. Results of Different Problems

Pro-blem	Number of Jobs	Number ofstations	Best O.F.V		CPU Time (hh:mm:ss)		Station No.	
			Lingo	GA	Lingo	GA	Lingo	GA
1	17	5	20.75	21.9	00:03:38	00:01:40	5	3
2	18	5	22.3	22.7	00:10:32	00:02:16	5	3
3	19	5	19.55	20.1	00:11:11	00:05:01	5	4
4	20	5	21	21.85	00:13:25	00:09:15	5	5
5	21	5	18.35	19.65	00:40:02	00:11:17	5	5
6	20	7	10(LB)	18.98	-----	00:15:16	---	4
7	30	10	11(LB)	20.15	-----	00:17:43	---	4
8	40	10	14(LB)	24.3	-----	00:18:36	---	4
9	50	7	24(LB)	33.6	-----	00:28:16	---	5
10	80	10	66(LB)	90	-----	00:30:57	---	4

According to Figure 7 and 8, it can be inferred that descending trend is a characteristic for approving proper function of defined operators in model. As the objective of the mode is minimization of cycle time and station number, descending trend of average presents that algorithm is approaching the optimum point.

Table 3. Fuzzy Process Time

job No.	fuzzy process time	job No.	fuzzy process time	job No.	fuzzy process time	job No.	fuzzy process time
1	(7,10,13)	21	(5,8,12)	41	(2,3,5)	61	(9,12,14)
2	(4,6,10)	22	(4,8,10)	42	(2,3,6)	62	(2,5,7)
3	(1,5,9)	23	(3,6,7)	43	(7,10,14)	63	(10,12,13)
4	(7,8,11)	24	(5,10,12)	44	(2,4,8)	64	(4,7,12)
5	(9,13,15)	25	(8,11,13)	45	(5,7,9)	65	(6,9,13)
6	(10,14,15)	26	(9,13,17)	46	(7,12,15)	66	(8,12,16)
7	(10,12,15)	27	(5,6,9)	47	(2,6,9)	67	(10,12,15)
8	(8,9,12)	28	(7,12,16)	48	(5,8,10)	68	(5,9,11)
9	(5,7,10)	29	(1,4,9)	49	(4,5,8)	69	(5,8,12)
10	(7,9,11)	30	(5,9,12)	50	(3,8,9)	70	(6,9,12)
11	(8,12,15)	31	(6,9,11)	51	(5,7,11)	71	(3,6,9)
12	(10,15,17)	32	(5,7,8)	52	(8,10,14)	72	(7,12,14)
13	(7,12,14)	33	(3,8,9)	53	(3,4,5)	73	(4,6,8)
14	(6,7,12)	34	(5,7,10)	54	(4,8,10)	74	(6,8,11)
15	(7,8,11)	35	(2,5,7)	55	(3,6,8)	75	(4,9,12)
16	(2,3,7)	36	(7,10,12)	56	(4,5,8)	76	(3,7,11)
17	(4,5,7)	37	(9,13,16)	57	(3,7,10)	77	(7,11,13)
18	(4,6,11)	38	(5,6,10)	58	(8,13,15)	78	(8,10,15)
19	(10,13,15)	39	(5,9,12)	59	(6,7,11)	79	(2,3,7)
20	(2,4,8)	40	(8,9,11)	60	(5,10,11)	80	(4,7,8)

Table 4. Prerequisites

job No.	Prereq-uisites	job No.	Prereq-uisites	job No.	Prereq-uisites	job No.	Prereq-uisites
1	----	21	12	41	10	61	13,36
2	1	22	16	42	33	62	52
3	2	23	2	43	22	63	59,61
4	4	24	21	44	38	64	27,62
5	4	25	12	45	24,40	65	31,49
6	1	26	3	46	39	66	35,54
7	3	27	4	47	7,45	67	29
8	3	28	22	48	25,34	68	58
9	4	29	18	49	29,48	69	20,56,64
10	9	30	13	50	23	70	51
11	7	31	24	51	46	71	8,44,66
12	9	32	9	52	14	72	43
13	6	33	17	53	16,21,37,47	73	68
14	12	34	19	54	12,53	74	50,60,63,65
15	8	35	5	55	54	75	70,73
16	2	36	15	56	20	76	33,69,75
17	1	37	6	57	55	77	13
18	10	38	36	58	13	78	41,42,74,77
19	10	39	11	59	32	79	28,57,67,71,72
20	7	40	2	60	10,26,30	80	76,78,79

Table 5. Results of Different Primary Populations with GA and Lingo

	Solving method	O.F.V	CPU Time (hh:mm:ss)	Station No.	Generation No.
	Lingo(LB)	66	-------	----	----
GA	P=100	266.15	00:02:55	4	32
	P=200	215.3	00:06:27	4	32
	P=400	126.5	00:10:49	4	32
	P=600	101.8	00:15:31	4	32
	P=800	99.85	00:22:52	4	32
	P=1000	90	00:30:57	4	32

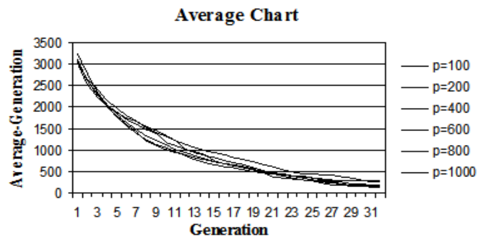


Fig.7. Means with Different Primary Population

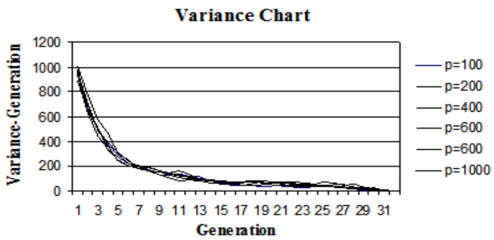


Fig.8. Variances with Different Primary Population

Figure 9 shows that for a problem with numerous jobs and stations, CPU time for Lingo software is relatively high and this trend increases exponentially while the developed genetic algorithm has low CPU time and this trend is almost linear. Therefore, the result reveals that the proposed algorithm and model has a proper function

Table 6. Means and Variances for Generations in Different Primary Populations

P \ G	A-G V-G	P=100	P=200	P=400	P=600	P=800	P=1000	P \ G	A-G V-G	P=100	P=200	P=400	p=600	P=800	P=1000
1	A-G	3249	3064	3067	3035	3119	3111	17	A-G	537	835	654	764	629	642
	V-G	936	956	981	895	1008	938		V-G	48.5	75.4	39	64.2	74.3	40.4
2	A-G	2809	2674	2674	2551	2654	2651	18	A-G	505	782	606	709	582	588
	V-G	703.1	719.9	719.9	656.3	787.4	671.4		V-G	45.4	77.5	50.1	61.9	78.2	58.9
3	A-G	2356	2444	2297	2238	2321	2329	19	A-G	476	723	564	651	544	544
	V-G	487	487	449	503	572	511		V-G	35	78	52	74	66	71
4	A-G	2070	2179	2062	2039	2032	2073	20	A-G	449	664	462	582	504	521
	V-G	375	385	332	355	464	328		V-G	38.8	76.9	61.2	78.6	61.5	52.6
5	A-G	1766	1980	1886	1896	1807	1858	21	A-G	424	605	386	513	465	484
	V-G	289	308	253	242	303	277		V-G	32.8	73.2	54.5	72	64	47.5
6	A-G	1581	1815	1734	1754	1650	1609	22	A-G	412	550	353	458	435	443
	V-G	219	244	206	204	249	222		V-G	24	72	29	63	60	42
7	A-G	1390	1672	1616	1670	1501	1448	23	A-G	400	500	330	402	408	406
	V-G	193	197	181	178	182	200		V-G	28.6	64.8	34	64	56.1	47.1
8	A-G	1251	1530	1486	1559	1359	1237	24	A-G	355	459	307	350	375	347
	V-G	160	184	192	161	162	171		V-G	45	57	35	51	52	44

9	A-G	1068	1412	1402	1468	1225	1117	25	A-G	322	443	286	317	347	306
	V-G	152	163	165	156	133	131		V-G	31	42	29	47	46	38
10	A-G	963	1315	1193	1352	1124	1124	26	A-G	302	437	253	280	314	290
	V-G	131	145	134	146	126	126		V-G	23	36	30	71	51	30
11	A-G	866.2	1220	1103	1228	1024	956.4	27	A-G	290	420	234	238	287	259
	V-G	135.6	130.8	123.5	159.3	105.4	81.27		V-G	13	25	29	67	54	44
12	A-G	776	1132	1014	1132	944	890	28	A-G	283	370	207	197	253	224
	V-G	117	106	116	173	95.4	87.6		V-G	5.4	16	22	45	27	53
13	A-G	718	1054	964	1020	876	820	29	A-G	281	348	194	183	224	186
	V-G	113	98.8	95.9	129	86.7	79.3		V-G	4.2	13	16	26	20	35
14	A-G	653	987	892	955	814	775	30	A-G	278	303	183	176	212	154
	V-G	80.9	91	92.6	98.4	82.4	62		V-G	4.1	9.3	9.3	9.1	10	19
15	A-G	624	936	750	880	741	733	31	A-G	276	270	178	171	189	144
	V-G	61.6	82	60.7	82.7	78.3	56.4		V-G	3.8	4.6	5.2	3.7	5.2	5.6
16	A-G	584	888	693	827	679	689	32	A-G	274	258	166	160	187	131
	V-G	58.4	73.7	46.6	70.3	75	51.1		V-G	3.1	3.1	2.9	2.5	1.9	1.8

P: Population, G: Generation
A-G: Average-Generation, V-G: Variance-Generation

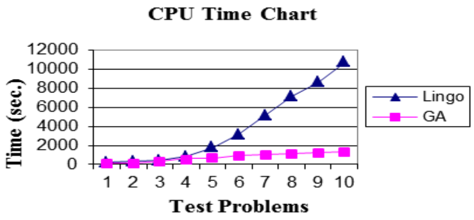


Fig.9. Comparison BetweenDesigned GA and Lingo According to CPU time

6. Conclusion

This study has focused on assembly line balancing problem considering two objectives that were minimizing cycle time and the number of work stations. A mixed-integer model that is combined with GA is proposed to solve the problem of line balancing. Processing time was not considered a constant in this study. We defined the processing time with fuzzy sets to represent a real manufacturing environment better. The proposed model was tested with small-sized problems. The results show that the proposed model and algorithm are able to reach optimum solution in short period of time.

Acknowledgements

The authors are thankful for constructive comments of respected reviewers. Taking care of the comments certainly improved the presentation of the paper

References

[1] M. Peeters and Z. Degraeve, *An linear programming based lower bound for the simple assembly line balancing problem*, European Journal of Operational Research, vol. 168, pp. 716-731, Feb 1, 2006.

[2] J. Bautista and J. Pereira, *A dynamic programming based heuristic for the assembly line balancing problem*, European Journal of Operational Research, vol. 194, pp. 787-794, May1,2009

[3] Salveson , M. E.; "*The Assembly Line Balancing Problem*", Journal Of Industrial Engineering , vol.6, p.p.18-25, 1995.

[4] Bowman, E. H.; "*Assembly line Balancing by Linear programming*", Operations Research, vol.8, p.p.385-389, 1960. [5] White, W. W.; "*Comments on a paper by Bowman*", Operation Research, vol.9, p.p.274-276, 1961.

[6] Baker, E.; "*An Additive alg. For Solving Linear programming with 0-1variables*", Management Science, vol.32, p.p.909-932, 1965.

[7] Patterson, J. H.; Albracht, J. J.; "*Assembly Line Balancing: 0-1 programming with fibonacci search*", Operation Research, vol.23, p.p.166-174, 1975.

[8] Fonseca, D.J., C.L. Guest, M. Elam and C.L. Karr."*A fuzzy logic approach to assembly line balancing*", Mathware Soft Comput., 12: 57-74, 2005.

[9] Anderson, E. J.;Ferris, M. C.; "*Genetic Algorithms for Combinational Optimization the Assembly Line Problem* ", ORSA J.on Computing, vol.6, p.p.161-173, 1994.

[10] Leu, y. y.; Matheson, L. A.;Ress, L. P.; "*Assembly Line Balancing Using Genetic Algorithms With Heuristic Initial Population and Multiple Evaluation* ", Decision Science, vol.4(25), p.p.581-606, 1994.

[11] Rubinoviz, J.;Levitin, G.; "*Genetics Algorithm for Assembly Line Balancing*", International Journal of Production Economics, vol.41, p.p.343-354, 1994.

[12] Tsujimura,Y.;Gen, M.; Kubta,E.; " *Solving Fuzzy Assembly Line Balancing Problem with Genetic Algorithms* ", Computers and Industrial Engineering, vol.1-4(29), p.p.543-547, 1995.

[13] Kim,Y. k.;Kim,Y. J.;Kim, Y.; "*Geneti Algorithms for Assembly Line Balancing with Various Objectives* ", Computers and Industrial Engineering,vol.3(30),p.p367-385.1996.

[14] Gen, M.;Tsujimura, Y.;Li, Y.; "*Fuzzy Assembly Line Balancing Using Genetic Algorithms* ", Computers and Industrial Engineering,vol.3-4(31), p.p.631-634, 1996.

[15] Ross, T. J.; "*Fuzzy Logic with Engineering Application*, McGraw-Hill", New York, 1995.

[16] Becker, C., & Scholl, A.;"*A survey on problems and methods in generalized assembly line balancing*", European Journal of Operational Research, 168(3), 694–715, 2006.

[17] Tasan, S. O., &Tunali, S. ;"*A review of the current applications of genetic algorithms in assembly line balancing*", Journal of Intelligent Manufacturing, 19(1), 49–69, 2008.

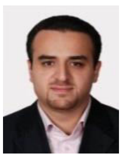
[18] Holland, J.H,; "*Adaptive in Natural and Artificial Systems*", Ann Arbor: University of Michigan Press, 1975.

[19] Alaghebandha, M. and Hajipour, V.;"*A soft computing-based approach to optimise queuing inventory control problem*", International Journal of Systems Science, 23 June, 2013.

[20] Zhang, W., Gen, M.; "*An efficient multiobjective genetic algorithm for mixed-model assembly line balancing problem considering demand ratio-based cycle time*", Journal of Intelligent Manufacturing, 2009.

[21] P. Th. Zacharia· Andreas C. Nearchou ; " *Multi-objective fuzzy assembly line balancing using genetic algorithms* ", Journal of Intelligent Manufacturing, 23:615-627,2012.

Author's Profile



Ramin Behbamzadeh received his BSc in Industrial Engineering from Islamic Azad University (IAU), Qazvin Branch in 2003and MSc.degrees from Iran University of Science and Technology in 2005.His research interests are in Reliability, Multi-Objective Optimization and Line Balancing.

Mohammad Alaghebandha is a doctoral student in the Department of Industrial Engineering at the Kharazmi University, Karaj, Iran. He obtained his BSc and MSc degrees in Industrial Engineering from Islamic Azad University (IAU), Qazvin Branch in 2008 and 2010, respectively. His research interests are in Inventory Control, Multi-Objective Optimization, Supply Chains,

Queuing Theory, Scheduling and Fuzzy sets. Mohammad Alaghebandha has published a number of papers and a book in journals such as *International Journal of Systems Science, Middle-East Journal of Scientific Research, Applied Mathematical Sciences* and others.

Amir Azizi is a senior lecturer at University Malaysia Pahang. He received BSc. degree from Iran in 2005, MSc. from University Putra Malaysia in 2007, and Ph.D. in industrial engineering from University Sains Malaysia. He has published journal and conference papers more than 40. He has the industrial working experiences since 2002. He is a member of editorial and advisory board of five international journals and he also is a reviewer of some international journals. He was the chairperson of several international conferences too. His research interests are on lean production system, total quality management, supply chain management, operation management, project management and manufacturing systems for modeling under uncertainties of production environments. He gained ASME award for the best paper at international student conference in 2006, and doctorate fellowship award in 2008. He is a member of ASME, IIE, and IAENG, IE-OR, and international society on multiple criteria decision making.