ICIC Express Letters
Part B: Applications
Volume **7**, Number **12**, December **2016**
 ISSN 2185-2766
pp. **2633–2640**

A HYBRID GRAVITATIONAL SEARCH ALGORITHM FOR TWO-SIDED ASSEMBLY LINE BALANCING PROBLEM WITH ZONING CONSTRAINTS

Yi Wu, Qiuhua Tang and Liping Zhang

Industrial Engineering Department
Wuhan University of Science and Technology
No. 947, Heping Avenue, Qingshan District, Wuhan 430081, P. R. China
tangqiuhua@wust.edu.cn

Received June 2016; accepted September 2016

Abstract. *Two-sided assembly lines are widely applied to plants which produce large-sized high-volume products. The zoning constraints are considered in this paper besides the traditional constraints of two-sided assembly line balancing problem (TALBP). Assembly line balancing problem is NP-Hard and the additional constraints make it more complicated. Therefore, a novel hybrid gravitational search algorithm (GSA) combined with variable neighborhood search (VNS) is proposed to solve TALBP with zoning constraints. The GSA seeks the global optimal and the VNS enhances the capability of local search. Moreover, a novel decoding scheme is designed to balance the workload between workstations and decrease the sequence-dependent finish time of tasks. The computational results demonstrate the effectiveness of the proposed algorithm.*
Keywords: Two-sided assembly line balancing, Zoning constraints, Gravitational search algorithm, Variable neighborhood search

1. **Introduction.** Two-sided assembly lines are a kind of assembly lines in which tasks are carried out on the same product in parallel at both sides of the line. Due to the use of both sides of a two-sided assembly line, task operation directions can be classified into three types: left side (L), right side (R) and either side (E). A two-sided assembly line is depicted in Figure 1. Two directly facing workstations (such as workstations 1 and 2) are called a mated-station, and one of them calls the other a companion. The sequence-dependent finish time of tasks should be taken into consideration when balancing two-sided assembly lines [1]. The sequence-dependent finish time of tasks is caused by the task waiting for its predecessor which has been assigned to the opposite side of the current mated-station. Considering tasks j and h are assigned to workstation 4, and task i which is the predecessor of task h is assigned to its companion workstation, so the idle time between tasks j and h (shaded rectangles in Figure 1) is unavoidable.

In some real applications, zoning constraints must be imposed on the two-sided assembly lines. Zoning constraints are divided into two types: positive zoning and negative zoning constraints [2,3]. Positive zoning constraint indicates a set of tasks must be operated at the same workstation and negative zoning constraint means tasks cannot be performed at the same mated-station.

Since [4] firstly proposed the TALBP in 1993, many approaches have been reported in the literature. However, less attention has been given to the TALBP with zoning constraints. [2] firstly solved the TALBP with zoning constraints using an ant-colony-based heuristic (ACO). [5] proposed the goal programming models for the TALBP with zoning constraints and multiple objectives were considered in their models. [6] proposed the bees algorithm (BA) to solve the TALBP without zoning constraints and with zoning

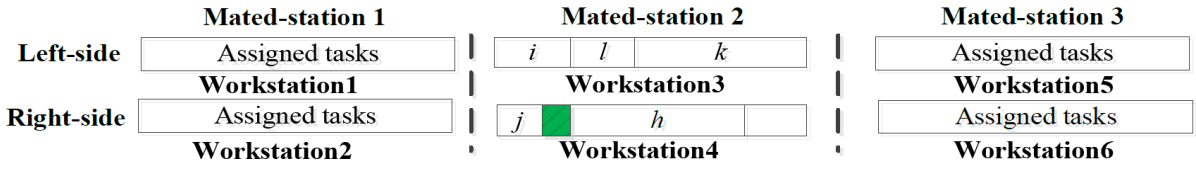

FIGURE 1. A two-sided assembly line

constraints. [7] proposed the harmony search to solve the TALBP with zoning constraints with the objective of minimizing the cycle time.

Gravitational search algorithm (GSA) is a new-born meta-heuristic algorithm which shows great performance and global search ability in solving various nonlinear functions [8]. As far as we know, there are no published papers using GSA to solve TALBP with zoning constraints. Taking into consideration of its novelty, effectiveness and good performance, we try to avoid its shortcoming of being trapped into local optimal and adjust it to solve TALBP effectively in this paper. Hence, a novel hybrid GSA is proposed to solve TALBP with zoning constraints for the objective of minimizing the number of mated-stations and the number of stations. In the hybrid GSA, variable neighborhood search (VNS) is integrated into GSA so as to enhance the local search ability of the standard GSA. What is more, a new decoding scheme based on task selection rule and side selection rule is proposed to avoid the sequence-dependent finish time of tasks and balance the workload between workstations. The rest of this paper is organized as follows. Section 2 illustrates the hybrid GSA and its application to solving TALBP with zoning constraints in detail. Section 3 gives the computational results. Some conclusions are given in the last section.

2. The Proposed Algorithm for TALBP with Zoning Constraints.

GSA is a newly developed stochastic population based search algorithm motivated by the law of gravity and mass interactions [8]. In GSA, agents attract each other by gravitational force which causes a global movement of all agents toward heavier masses. Each agent represents a solution and the heavier one corresponds to a better solution. By lapse of time, the agents will be attracted by the heaviest mass which represents an optimal solution in the search space. In order to enhance the local search ability of GSA, VNS is combined with GSA to improve its performance. The proposed hybrid GSA has five phases including initialization, fitness evaluation, calculation of the total force exerted on each agent, updating agents' acceleration, velocity and position and improving each agent using VNS.

2.1. Encoding scheme and initialization.

Consider a system with N agents. The position of the ith agent is defined as follows:

$$K_i = \left(k_i^1, \ldots, k_i^d, \ldots, k_i^n\right) \text{ for } i = 1, 2 \ldots, N \tag{1}$$

where k_i^d represents the position of the ith agent in the dth dimension and n is the dimension of the search space.

GSA is a continuous algorithm in nature, and the standard encoding scheme of GSA cannot be applied directly to TALBP. So in order to qualify GSA to solve the TALBP, random-keys method [9] is used to represent task sequence. Take the TALBP with 12 tasks as an example. We randomly generate 12 floating-point numbers between 0 and 1, and the position with lower value should be assigned earlier in the task sequence.

2.2. Decoding to get a feasible solution.

The task sequence obtained by random-keys method is not a feasible solution. So the decoding process is designed to get a feasible solution. Compared with traditional TALBP, this process has to deal with zoning

constraints and a heuristic side selection and task selection rule is developed to balance the workload between workstations and decrease the sequence-dependent finish time of tasks. The main idea of this process is to generate candidate tasks for both sides of the current mated-station and then select a side and task by the side and task selection rule, respectively. Then check whether the task belongs to the zoning constraints. If the task satisfies the zoning constraints or the task does not belong to the zoning constraints, assign the task to the corresponding workstation. Figure 2 is the main procedure of the decoding scheme.

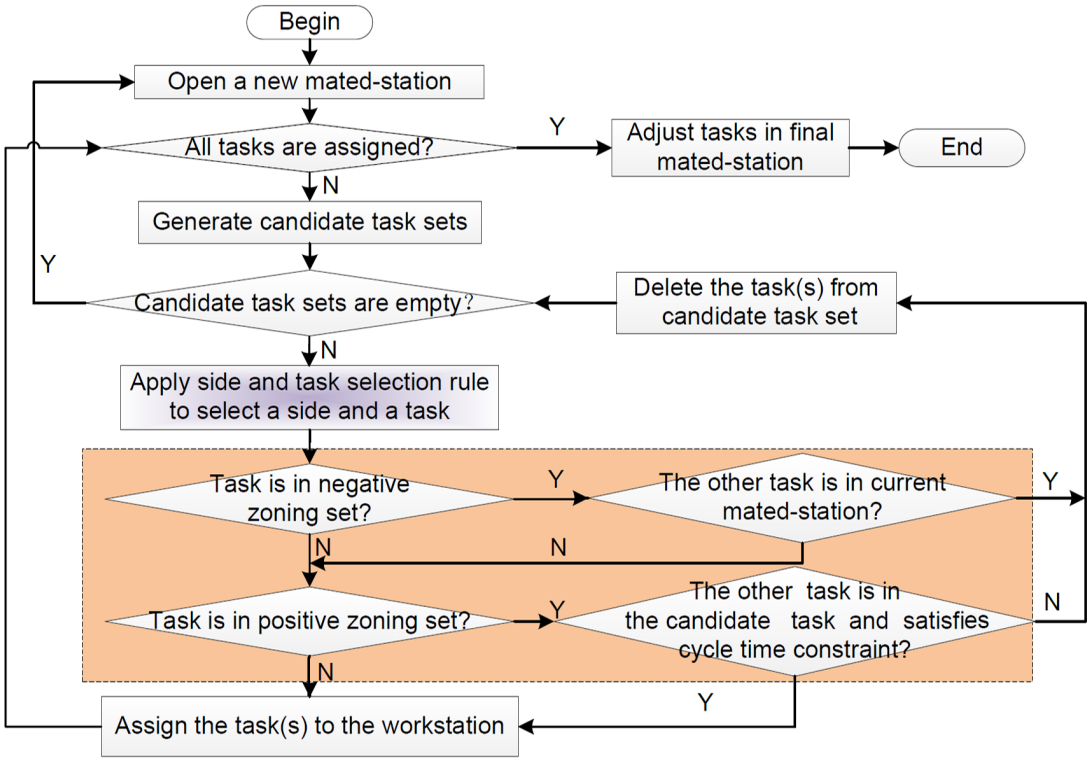

FIGURE 2. Main procedure of the decoding scheme

2.2.1. *Side selection rule.* Side selection rule is used to select the side for the current mated-station. If both candidate task sets are not empty, choose one side which can perform the task earlier or choose one randomly when the start times of both sides are equal; else choose the side which has candidate tasks.

2.2.2. *Task selection rule.* Task selection rule is applied to choosing one task for the selected side of the current mated-station. Firstly, we should decide whether tasks which can be operated at the earliest possible time for the selected side of the current mated-station exist. If the tasks exist, select a task according to the priority of tasks; if not, select a task from the candidate task set according to the priority of tasks.

2.2.3. *Adjusting tasks in final mated-station.* Tasks in the final mated-station can be reassigned to one workstation when it satisfies the following conditions: ① the total operation time of the tasks in the final mated-station is not greater than the cycle time; ② the directions of the tasks are L and E, R and E or E and E.

2.3. **Fitness function.** After decoding we can get a solution and calculate its fitness. Since two objectives are involved in this paper: minimizing the number of mated-stations and the number of workstations simultaneously, we use the linear weighting method to combine the two objectives into one

$$fit_i(t) = \beta \times NS + \gamma \times NM \tag{2}$$

Given the condition that two different solutions have the same number of workstations and mated-stations, one may be better balanced than the other. So when two solutions with the same fitness are obtained, Equation (3) is utilized to distinguish these two solutions. The solution with smaller SI is selected since it has a smaller difference among workstations.

$$SI_i(t) = \sqrt{\sum_{k=1}^{NS} (CT - S_k) \Big/ NS} \tag{3}$$

where NS, NM, CT and S_k are the number of workstations, mated-stations, cycle time and workload of workstation k, respectively. β and γ are the weighted coefficients of the number of workstations and mated-stations. In this study, β and γ are set as equal to 1.0 and 2.0, respectively.

2.4. **Calculation of the total force.** The force between two agents is directly proportional to the product of their masses and inversely proportional to the square of the distance between them. At a specific iteration t, total force exerted on agent i in d dimension is defined by:

$$F_i^d(t) = G(t) \times M_i(t) \sum_{j=1, j\neq i}^{N} rand_j \times \frac{M_j(t)}{R_{ij}(t) + \xi} \left(k_j^d(t) - k_i^d(t)\right) \tag{4}$$

where $R_{ij}(t)$ is the Euclidian distance ($R_{ij}(t) = \|k_i(t) - k_j(t)\|_2$) between agents i and j, ξ is a small constant, $G(t)$ is the gravitational constant at iteration t, and $rand_j$ is a random number in the interval $[0, 1]$. $M_i(t)$ and $M_j(t)$ are the masses of agent i and j.

The mass of each agent is computed according to the relative efficiency of the fitness function which is as follows:

$$M_i(t) = \frac{fit_i(t) - worst(t)}{\sum_{j=1}^{N} (fit_j(t) - worst(t))} \tag{5}$$

where $fit_i(t)$ represents the fitness value of the agent i at t, and $worst(t)$ is the worst fitness of all agents.

The gravitational constant G is initialized at the beginning and will be decreased with the iteration to control the search accuracy.

$$G(t) = G_0 \times \exp\left(-\alpha \times \frac{t}{T}\right) \tag{6}$$

where G_0 is the initial gravitational constant, α is a control constant, t is the current iteration and T is the max iteration number.

2.5. **Updating acceleration, velocity and position.** All agents move along the direction of the total force based on the velocity and acceleration. The acceleration of the agent i at iteration t in the dth dimension is calculated using Equation (7):

$$\alpha_i^d(t) = \frac{F_i^d(t)}{M_i(t)} \tag{7}$$

And then the next velocity and its next position are calculated as Equations (8) and (9):

$$V_i^d(t+1) = rand_i \times V_i^d(t) + \alpha_i^d(t) \tag{8}$$

$$k_i^d(t+1) = k_i^d(t) + V_i^d(t+1) \tag{9}$$

2.6. **Improving agents by VNS.** VNS is a meta-heuristic algorithm which uses more than one neighborhood structure during the search process [10]. When a systematic switch from one type of neighborhood search structure (NSS) is performed, it has more possibilities to find better solutions. In this paper four types of different NSSs are designed to produce new neighborhood solutions. Four neighborhoods $N_1(s)$, $N_2(s)$, $N_3(s)$ and $N_4(s)$ are characterized as follows:

1) $N_1(s)$ (swap): Select two tasks i_1 and i_2 ($i_1 \neq i_2$) randomly and then swap tasks i_1 and i_2.
2) $N_2(s)$ (multi swap): Repeat the first NSS twice.
3) $N_3(s)$ (backward insert): Select two tasks i_1 and i_2 randomly and then insert i_1 to the back of i_2 (Assume that i_1 is before i_2 in the sequence).
4) $N_4(s)$ (forward insert): Select two tasks i_1 and i_2 randomly and then insert i_2 to the fiont of i_1 (Assume that i_1 is before i_2 in the sequence).

Among the four NSSs, the first NSS is done first when the VNS is carried out. If no improvement is made, the next NSS is utilized, and when a better solution is achieved, the first NSS is used again. Figure 3 is the pseudo code of the proposed VNS.

```
Determine the neighborhood structure N_k,
k=1,2,3,4
Input the initial solution s
While(k<=4) do
    shake procedure: find a new solution s'
    Perform N_k(s') to find a solution s";
     if fit(s")<fit(s) then
        s=s", k=1;
     else
        k=k+1;
     end if
end while
```

FIGURE 3. Pseudo code for the VNS

2.7. **Main body of the hybrid GSA.** The main body of the hybrid GSA is described in Figure 4. VNS is combined with the GSA so as to improve the local search ability of GSA and to strike a balance between the diversification and the intensification.

3. **Experimental Design and Results.** The proposed hybrid GSA for TALBP with zoning constraints is programmed in Visual C++2010 and runs on Intel (R) Core5 (TM) CPU 3.1 GHz, 8 GB memory PC with Microsoft Windows7. In order to demonstrate the effectiveness and efficiency of the proposed hybrid algorithms, three large-sized problems P65, P148, P205 are solved. P148 is taken from [4], and P65 and P205 are taken from [1]. In P148, the operation times of tasks 79 and 108 are changed from 281 to 111 and from 383 to 43 according to [1]. The zoning constraints of all these problems are taken from [2]. In this section, we first calibrate the proposed decoding scheme with four heuristic based decoding methods. And then the performance of the proposed hybrid GSA is compared with those of ACO [2], BA [6], and GSA.

3.1. **Comparison of different decoding schemes.** The proposed decoding scheme uses the task selection rule that selects the task which can be operated at the earliest possible time (EBT). There are other heuristic task selection rules which can be used in the proposed decoding scheme including: (1) Max-IFOL: select the task having the maximum number of immediate follower tasks; (2) MAX-TTST: select the task having

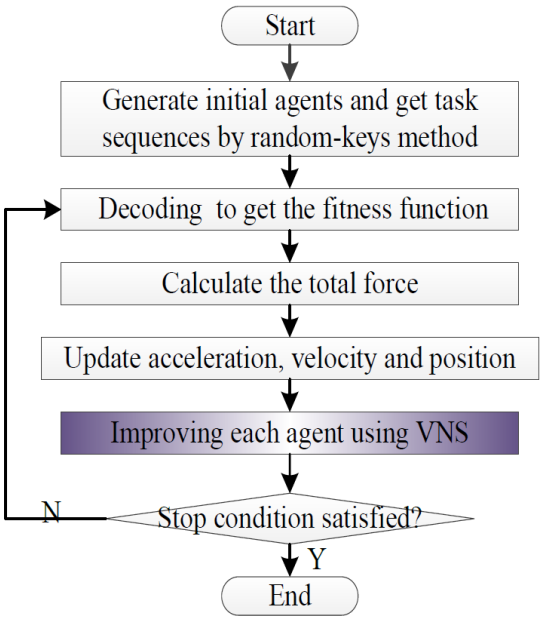

FIGURE 4. Main body of the hybrid GSA

maximum total processing time of successor tasks; (3) Max-TFOL: select the task having the maximum total number of follower tasks; (4) Max-RPW: select the task having maximum ranked positional weight, where the ranked positional weight of a task is the sum of the processing time of the task and all its follower tasks. So these four task selection rules can be applied to selecting a task and hence generate four new decoding schemes.

For the evaluation of these decoding schemes, P148 is selected to test the performance of these five decoding schemes. 15 different instances are randomly generated and each instance is run 10 times. The stopping criterion is fixed to a maximum CPU time. The average relative percentage deviation (RPD) for each instance is utilized for comparison and RPD can be calculated using the following expression.

$$\text{Relative Percentage Deviation (RPD)} = \frac{Some_{sol} - Best}{Best} \times 100 \tag{10}$$

where $Some_{sol}$ is the solution obtained by one running and $Best$ is the optimal solution obtained by any of these decoding schemes. Analysis of variance (ANOVA) is used to check whether these five decoding schemes are different in the RPD value. The response variable is the RPD value of each decoding scheme. The means plot for the single factor is depicted in Figure 5. From the results we can see that our proposed decoding scheme outperforms the other four methods.

3.2. **Computational results and comparisons.** In this subsection, three large-sized test problems with various cycle times are examined. The stopping criterion is set as $n \times n \times 10$ ms. The parameters G_0, α and population size are set as 10, 1, 50 respectively, according to the preliminary parameter adjustment experiments.

The computational results are summarized in Table 1. Two evaluation criteria i.e. the number of workstations (NS) and the number of mated-stations (NM) are used in this experiment. Maximum, minimum and average numbers of workstations and the best number of mated-stations among 10 times are reported in Table 1.

As can be seen in Table 1, GSA and hybrid GSA find all the best solutions. The bold characters indicate the NS values which are better than ACO and BA. GSA and the proposed hybrid GSA obtained the best solutions for all the 22 large-sized instances. Among the 22 large-sized instances in Table 1, the proposed hybrid GSA outperforms 16 cases over ACO and 11 cases over BA, respectively. For the largest test problem P205, the

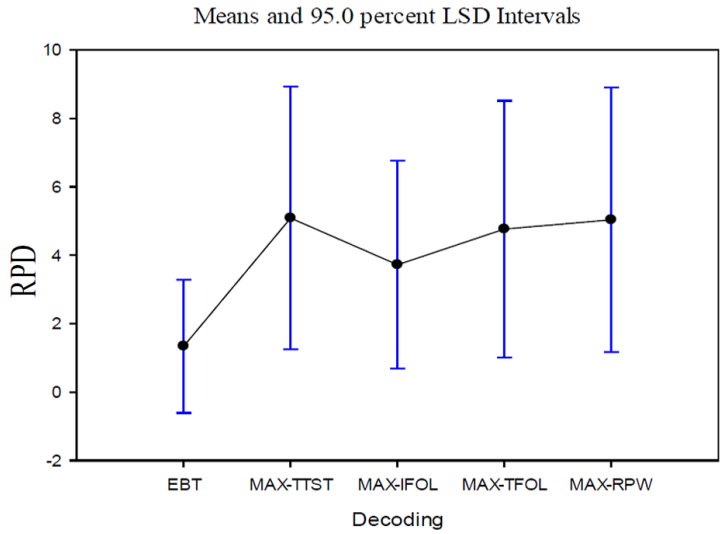

FIGURE 5. Means plot for decoding

TABLE 1. Computational results

Problem	Cycle Time	LB	ACO	BA	GSA				hybrid GSA			
					NS			NM	NS			NM
					Min	Avg	Max		Min	Avg	Max	
P65	326	16	17	17	17	17.3	18	9	17	17	17	9
	381	14	15	14	14	14.8	15	7	14	14	14	7
	435	12	13	13	13	13	13	7	13	13	13	7
	490	11	12	11	11	11	11	6	11	11	11	6
	544	10	10	10	10	10	10	5	10	10	10	5
P148	204	26	26	26	26	26.2	27	13	26	26	26	13
	255	21	21	21	21	21	21	11	21	21	21	11
	306	17	18	18	**17**	17.8	18	9	**17**	17.6	18	9
	357	15	18	15	15	15	15	8	15	15	15	8
	408	13	15	14	**13**	13	13	7	**13**	13	13	7
	459	12	13	12	12	12	12	6	12	12	12	6
	510	11	11	11	11	11	11	6	11	11	11	6
P205	1133	21	25	23	23	23	23	12	23	23	23	12
	1322	18	22	21	**20**	20	20	10	**19**	19.2	20	10
	1510	16	19	18	**17**	17.9	18	9	**17**	17.1	18	9
	1699	14	18	17	**15**	15.9	16	8	**15**	15.6	16	8
	1888	13	16	16	**14**	14.4	15	7	**14**	14	14	7
	2077	12	16	14	14	14	14	7	**13**	13.2	14	7
	2266	11	14	14	**13**	13	13	7	**12**	12.4	13	6
	2454	10	14	14	**12**	12	12	6	**11**	11.8	12	6
	2643	9	13	13	**11**	11.7	12	6	**11**	11	11	6
	2832	9	12	12	**11**	11	11	6	**10**	10.3	11	5

proposed algorithm obtained 10 of 11 better solutions than ACO and BA. The proposed hybrid GSA gets five new better solutions than GSA. The average number of workstations obtained by the proposed algorithm is fewer than that of GSA which shows good stability. The majority differences between ACO or BA and the results by hybrid GSA are three, such as P205 ($CT = 2454$), which is a great improvement. The proposed hybrid GSA obtained the lower bounds (LB) values of workstations for 10 of the 22 test instances.

In summary, the proposed hybrid GSA demonstrates that it is efficient and effective for solving TALBP with zoning constraints.

4. **Conclusions.** In this paper, a new hybrid GSA is proposed to solve TALBP with zoning constraints which aims to minimize the number of mated-stations and the number of workstations. The random-keys method is applied to making the GSA for TALBP. And a decoding scheme which consists of side selection and task selection rules is proposed to get a feasible solution. The workload between workstations can be balanced through the side selection rule and sequence-dependent finish time of tasks can be reduced by using task selection rule. And the performance of the proposed decoding scheme is demonstrated by comparison with the other four decoding schemes through ANOVA. Moreover, VNS is employed to improve solutions generated by GSA so as to avoid being trapped into local optimal. The computational results showed that the hybrid GSA can solve the problem effectively especially for large-sized problems. In the future, the proposed algorithm can be applied to solving other assembly line balancing problems such as mixed-model assembly balancing problem and parallel assembly line balancing problem.

Acknowledgment. The work is supported by the National Science Foundation of China under grants 50875190, 51275366 and 51305311. The authors also thank suggestions and comments from the anonymous referees to improve this paper.

REFERENCES

[1] T. O. Lee, Y. Kim and Y. K. Kim, Two-sided assembly line balancing to maximize work relatedness and slackness, *Computers & Industrial Engineering*, vol.40, no.3, pp.273-292, 2001.

[2] A. Baykasoglu and T. Dereli, Two-sided assembly line balancing using an ant-colony-based heuristic, *International Journal of Advanced Manufacturing Technology*, vol.36, pp.582-588, 2008.

[3] B. Yuan, C. Y. Zhang and X. Y. Shao, A late acceptance hill-climbing algorithm for balancing two-sided assembly lines with multiple constraints, *Journal of Intelligent Manufacturing*, vol.26, no.1, pp.159-168, 2015.

[4] J. J. Bartholdi, Balancing two-sided assembly lines: A case study, *International Journal of Production Research*, vol.31, no.10, pp.2447-2461, 1993.

[5] U. Özcan and B. Toklu, Multiple-criteria decision-making in two-sided assembly line balancing: A goal programming and a fuzzy goal programming models, *Computers & Operations Research*, vol.36, no.6, pp.1955-1965, 2009.

[6] L. Özbakır and P. Tapkan, Bee colony intelligence in zone constrained two-sided assembly line balancing problem, *Expert Systems with Applications*, vol.38, no.9, pp.11947-11957, 2011.

[7] H. D. Purnomo and H. M. Wee, Maximizing production rate and workload balancing in a two-sided assembly line using harmony search, *Computers & Industrial Engineering*, vol.76, pp.222-230, 2014.

[8] E. Rashedi, H. Nezamabadi-Pour and S. Saryazdi, GSA: A gravitational search algorithm, *Inform Sciences*, vol.179, no.13, pp.2232-2248, 2009.

[9] J. C. Bean, Genetics and random keys for sequencing and optimization, *ORSA Journal on Computing*, vol.6, no.2, pp.154-160, 1994.

[10] N. Mladenović and P. Hansen, Variable neighborhood search, *Computers & Operations Research*, vol.24, no.11, pp.1097-1100, 1997.