Artificial bee colony algorithms for two-sided assembly line worker assignment and balancing problem

Mukund Nilakantan Janardhanan [1*], Zixiang Li[2], Peter Nielsen[1], Qiuhua Tang[2]

[1]Department of Mechanical and Manufacturing Engineering, Aalborg University, Denmark

Email: {mnj, peter}@m-tech.aau.dk

[2]Industrial Engineering Department, Wuhan University of Science and Technology, Wuhan 430081, Hubei, China

Email: zixiangli@126.com, tangqiuhua@wust.edu.cn

Abstract. Worker assignment is a new type of problem in assembly line balancing problems, which typically occurs in sheltered work centers for the disabled. However, only a few contributions consider worker assignment in a two-sided assembly line. This research presents three variants of artificial bee colony algorithm to solve worker assignment and line balancing in two-sided assembly lines. The utilization of meta-heuristics is motivated by the NP-hard nature of the problem and the chosen methods utilize different operators for onlooker phase and scout phase. The proposed algorithms are tested on 156 cases generated from benchmark problems. A comparative study is conducted on the results obtained from the three proposed variants and other well-known metaheuristic algorithms, such as simulated annealing, particle swarm optimization and genetic algorithm. The computational study demonstrates that the proposed variants produce more promising results and are able to solve this new problem effectively in an acceptable computational time.

Keywords: Assembly line balancing; two-sided assembly line; worker assignment; artificial bee colony; metaheuristics

1. Introduction

Extensively industry is using assembly lines to assemble different types of products, where a set of tasks is allocated to workstations and each workstation is assigned with one or several workers. In any real application, the problem of worker assignment requires consideration due to the different skills of the workers. This situation has been studied for sheltered work centers for the disabled [1, 2], where disabled workers might need more times to operate certain tasks or even are incapable of operating some tasks. The current state of task allocation mechanisms followed in assembly line balancing problems presently ignores this situation. Worker assignment and task allocation results in a new integrated assembly line worker assignment and balancing problem, which contains two interacted sub-problems. The worker assignment problem determines the

S. Omatu et al. (eds.), *Distributed Computing and Artificial Intelligence, 14th International Conference*, Advances in Intelligent Systems and Computing 620, DOI 10.1007/978-3-319-62410-5_2

assignment of the workers on workstations and the assembly line balancing problem allocates tasks to workstations while satisfying different constraints.

Two-sided assembly lines are widely applied in automobile industries to assemble large-size high-volume products. In this type of line, workers operate tasks on two-faced workstations, referred to as mated-stations, in parallel [3, 4]. In this research worker assignment and balancing of a two-sided assembly line are taken into account simultaneously, a new integrated two-sided assembly line worker assignment and balancing problem (TALWABP) is proposed.

Even though there are plenty of contributions regarding two-sided assembly line balancing problems (TALBP) or worker assignment in one-sided assembly line, to the author's best knowledge there is no reported research on TALWABP where cycle time minimization is considered. Hence, this paper presents a first approach to solve the TALWABP with the objective of minimizing the cycle time. As a common objective function, the cycle time minimization criterion has great applications for the reconfiguration of the installed assembly lines[5]. A simple balancing problem in an assembly line is classified as NP-hard[4]; the considered problem also falls under this category due to the additional complexity incorporated. Hence, there is a need to use optimization techniques such as constraint programming and metaheuristics to solve large problem instances [1, 6, 7]. In this paper, several variants of the artificial bee colony (ABC) algorithm are developed to tackle the problem. A set of benchmarks are generated based on the benchmarks available in [8]. A comparative study on the proposed algorithms and three other well-known metaheuristics is presented and discussed in detail to demonstrate the performance of the proposed algorithms.

The reminder of the paper is organized as follows. Section 2 presents the problem description. Section 3 presents the details of the proposed algorithms. The comparative study and the statistical analysis are presented in Section 4. Finally, Section 5 concludes this research and suggests several future directions.

2. Problem definition

To scope the problem, we consider a set of assumptions as follows:

1) Workers have different skills and the operation times of tasks depend on the assigned workers.

2) All workstation is assigned a worker and the worker number is equal to the workstation number.

3) Some workers are incapable of operating some tasks and the corresponding operation times are set to a very large number when this situation happens.

In TALWABP, there is a special constraint on the preferred directions of tasks, referred to as direction constraint. Direction constraints can be grouped into three general types: L-type tasks, R-type tasks and E-type tasks. L-type tasks are allocated to the left side, R-type tasks are allocated to the right side and E-type tasks can be allocated to the left or right side[9]. In addition, there is a special condition for tasks on a mated-station, which is the existence of sequence-depended idle time[5]. Sequence-depended idle time is due to the precedence constraint and utilization of two sides, and can be reduced by

optimizing the task sequence on each workstation. Taking the sequence-depended idle time into account, our solution to the TALWABP optimizes not only the worker assignment and task allocation to workstations, but also the task sequence on each workstation. An example of worker assignment and task allocation on two-sided assembly line is depicted in Fig.1. In this figure, two facing workstations on the left side and right side comprise a mated-station. For instance, workstation (1, 1) and workstation (1, 2) comprise mated-station 1. Each workstation is assigned a worker and there are four workers on the four workstations. Two workers on a mated-station operate the allocated tasks simultaneously. Two-sided assembly line balancing determines the detailed allocation of tasks on each mated-station while worker assignment assigns the best-fit worker to each workstation.

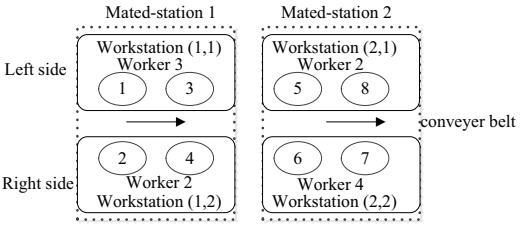

Fig.1 Worker assignment and task allocation on a two-sided assembly line

3. Proposed Methodology

Artificial Bee Colony (ABC) is one of the recently developed metaheuristic algorithm which has shown promising results for solving two sided assembly line balancing problem[5, 10] due to the faster convergence rate and there are only few parameters to be fine-tuned. This section first describes the procedure of the basic ABC algorithm, subsequently describes the proposed encoding and decoding policies along with the details of several variants of ABC algorithm.

3.1 ABC procedure

When solving optimization problems utilizing ABC algorithms, each solution is regarded as a food source and the fitness of an individual is referred to as the nectar amount. A pseudocode example of the ABC algorithm is presented in Fig. 2. Three groups of bees in the swarm search for the best food source, namely employed bees, onlookers and scouts. Employed bees exploit the nectar sources and provide the food sources' information to onlookers. Onlooker select food sources to exploit which emphasize intensification. The scout carries a random search to achieve a new food source to emphasize exploration. The procedure of the ABC algorithm is clarified in the following, where PS is the number of employed bees or onlookers. It should be noted that the number of employed bee is equal to onlookers and an onlooker becomes a scout.

Begin:
Initialize the swarm with *PS* individuals; *% Initialization*
While (termination criterion is satisfied**) do**
For *p*=1 to *PS* **do** *% Employed bee*
Achieve a new individual based on the incumbent individual *p* utilizing neighbor operator;
Replace the incumbent one with new one when better fitness value is achieved;
For *p*=1 to *PS* **do** *% Onlooker*
Select an individual based on probability value for each food source;
Achieve a new individual based on the selected individual utilizing neighbor operator;
Replace the incumbent one with new one when better fitness value is achieved;
If (One individual should be abandoned**)** *% Scout*
Determine the abandoned individual;
Replace the abandoned individual with the new one achieved by a scout
Endif
End

Fig.2 Pseudocode of ABC algorithm

3.2 Encoding and decoding

Since the addressed problem is similar to the robotic TALBP in [11] this paper proposes a similar encoding method which utilizes two vectors: task permutation and worker assignment. It is to be noted that the task permutation vector is selected since it has shown superior performance than the mated-station oriented encoding [5]. In the two vectors, the worker assignment vector determines the detailed worker assignment and task permutation determines the sequence of allocating tasks. The former task in the task permutation is allocated first. An example with 12 tasks and 2 mated-station is illustrated in Fig. 3 to describe the solution presentation. It is observed that the worker assignment vector is 3, 2, 1 and 4, and thus worker 3 and 2 are assigned to the left side and right side of mated-station 1 and worker 1 and 4 are assigned to the left side and right side of mated-station 2. The three former tasks in the task permutation are task 3, 2 and 6. Hence, task 3, 2 and 6 are allocated first and are all allocated to mated-station 1.

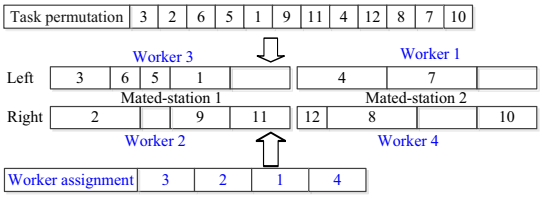

Fig.3 Illustrated solution presentation

However, the task permutation is not a feasible task allocation, and an initial cycle time and an effective decoding scheme are necessary to transfer the task permutation

into a feasible task allocation. Based on the task permutation, the decoding procedure in Li et al. [11] is proposes to transfer the task permutation into a feasible task allocation. In this decoding procedure, tasks are allocated to former workstations as much as possible gradually, and the last mated-station endures all the unallocated tasks. The largest finishing time of all mated-stations is regarded with the cycle time achieved by this task permutation. Regarding the initial cycle time (CT), it is set to be a large number in this paper. The initial cycle time is updated with $\mathrm{CT} \leftarrow \mathrm{CT}_N - 1$ when a smaller new best cycle time CT_N is achieved or $\mathrm{CT} \leftarrow \mathrm{CT} - 1$ when the current best cycle time remains unchanged. The above expression guarantees that the initial cycle time reduces over the algorithm's evolution, and as a consequence, the achieved best cycle time is reduced. When new best cycle time is achieved, all the individuals are re-decoded using the initial cycle time of $\mathrm{CT}_N - 1$ and the new achieved cycle times are set to their objective functions. This method, referred to as the iteration mechanism, ensures that all the individuals are evaluated using the same initial cycle time.

3.3 Proposed ABC methods

This paper develops three variants of ABC algorithms, which are different in onlooker phase and scout phase. The first variant, referred to as ABC1, is the traditional ABC. The other two versions, referred to as ABC2 and ABC3, have some modifications. The features of the three method are listed as follows.

ABC1: In the onlooker phase, the selection probability value for food source p, Pr_p is calculated with $\mathrm{Pr}_p = \left(1/CT_p\right)/\sum_{p'}^{PS}\left(1/CT_{p'}\right)$, where CT_p is the cycle time achieved by individual p. Then an individual is selected using roulette selection. In the scout phase, if the best cycle time is not updated, the duplicated or worst individual with the largest cycle time is replaced with a new randomly generated solution. **ABC2:** ABC2 differs from ABC1 in the way the duplicated or worst individual with the largest cycle time is replaced with a neighbor solution of the randomly selected individual. **ABC3:** ABC3 differs from ABC 1 in both the onlooker and scout phase. In the onlooker phase, ABC3 utilize tournament selection, where two individuals are randomly selected and the individual with the smaller cycle time is finally selected. In the scout phase, the duplicated or worst individual with the largest cycle time is replaced with a neighbor solution of the randomly selected individual.

Compared with the roulette selection in the onlooker phase, tournament selection is much simpler. In the scout phase, the randomly generated solution always has poor performance. Thus, ABC2 and ABC3 utilize the neighbor solution of the randomly selected individual to replace the abandoned one. Apart from the above factor, this neighbor operator is another important factor and all the algorithms share the same neighbor operator. In this neighbor operator, this research employs both insert operation and swap operation to modify either the task permutation vector or worker assignment vector. The insert operation and swap operation have been widely applied in assembly line balancing problems, such as robotic TALBP in Li et al. [11]. Since two vectors are involved, the approach in this research is to first select a vector randomly and later select insert operation or swap operation randomly to modify the selected vector.

4. Computational Study

This section presents the details of the tests conducted on the proposed method and their findings. The utilized benchmark problems are first introduced and the comparative study is presented later. Since there is no available benchmark problem for this new TALWABP, this research generates a set of tested problems on the basis of the original TALBP benchmarks summarized in Tang et al. [5]and Li et al. [9]. The operation times of tasks are produced using the method in Chaves et al. [8], where two variabilities of the operation times are employed: low variability and high variabilities. The first workers have the same operation times published in literature, and the operation times for the remaining workers are generated as follows. Regarding low variability, the operation times of task *i* by workers are randomly generated within a uniform distribution of [1, t_i] where is t_i the original operation times of task *i* in literature. Regarding high variability, the operation times of task *i* by workers are randomly generated within a uniform distribution of [1,3×t_i]. In addition, two ratios of task-worker incompatibilities are utilized: low ratio with 10% and high ratio with 20%. Since the original benchmarks contain 39 cases [12], and there are 156 combinations due to two operation time variabilities and two incompatible percentages. This new benchmark set is summarized in Table 1, where *Nt* is the number of tasks and *Nm* is the number of mated-stations.

Table 1 Description of the tested benchmarks

Problem	*Nt*	*Nm*	Time variabilities	Incompatibilities
P9	9	2,3	Low, high	Low, high
P12	12	2,3,4,5	Low, high	Low, high
P16	16	2,3,4,5	Low, high	Low, high
P24	24	2,3,4,5	Low, high	Low, high
P65	65	4,5,6,7,8	Low, high	Low, high
P148	148	4,5,6,7,8,9,10,11,12	Low, high	Low, high
P205	205	4,5,6,7,8,9,10,11,12,13,14	Low, high	Low, high

4.1 Computational evaluation

This section presents the comparative study and three other algorithms are adopted: simulated annealing algorithm[12], particle swarm optimization algorithm [11] and genetic algorithm[13]. All the tested algorithms are calibrated using full factor design following Tang et al. [5] and [11]. All cases are solved by all the combinations of the parameters and they terminate when CPU time reaches $\mathrm{Nt} \times \mathrm{Nt} \times \tau$ milliseconds, where τ is set to 10. After executing all the experiments, the relative percentage deviation (RPD) is proposed to transfer the achieved cycle times using $\mathrm{RPD} = 100 \times (CT_{some} - CT_{Best})/CT_{Best}$, where CT_{some} is the cycle time yield by one combination and CT_{Best} is the cycle time achieved by all the combinations. The multifactor analysis of variance (ANOVA) technique is applied to analyze the achieved RPD values as done in Li et al. [11]. Based on the determined parameters, all algorithms are solved using 156-benchmark problems under the two-termination criterion of $Nt \times Nt \times 10$ milliseconds and $Nt \times Nt \times 20$ milliseconds. Table 2 presents the average RPD values of tested algorithms, where the average RPD value in each cell is the average value of several cases. For instance, the value for P205 is the average value of 11

cases. It can be observed that ABC3 is the best performer with an average RPD value of 3.90 under the first termination criterion. ABC2 is the second best performer with an average RPD value of 4.15. PSO and GA algorithms showed worst performance when compared. When the computational time reaches $Nt \times Nt \times 20$ milliseconds, ABC3 still ranks the first and ABC ranks the second according to the increasing order of the average RPD values. Among the remaining algorithms, SA ranks third, ABC1 ranks fourth, GA ranks fifth and finally PSO ranks sixth. In addition, ABC3 is the best performer among the three ABC variant for P24 and P148 using the second termination criterion. ABC3 is the best performer for P65 and P205. These computational results suggest that the ABC2 and ABC3 are quite effective when solving TALWABP due to its ability to achieve good solution in short computation time.

Table 2 Average RPD values of six tested algorithms

Problem	Tested cases	Average relative percentage deviation						CPU time(s)
		SA	PSO	GA	ABC1	ABC2	ABC3	
Nt × Nt × 10								
P9	2	0.00	0.00	0.00	0.00	0.00	0.00	0.81
P12	4	0.00	2.08	1.56	0.00	1.56	0.00	1.44
P16	4	0.42	0.00	0.00	0.00	0.00	0.00	2.56
P24	4	0.48	4.01	2.09	0.00	0.42	0.51	5.76
P65	5	6.05	32.54	11.40	7.28	4.90	5.09	42.25
P148	9	11.43	39.78	16.04	15.78	7.40	6.88	219.04
P205	11	8.45	43.16	13.64	13.35	5.70	5.71	420.25
Average RPD		5.89	26.15	9.39	8.34	4.15	**3.90**	-
Nt × Nt × 20								
P9	2	0	0.00	0.00	0.00	0.00	0.00	1.62
P12	4	0.00	2.08	1.56	0.00	0.00	0.00	2.88
P16	4	0.42	0.00	0.00	0.00	0.00	0.00	5.12
P24	4	0.48	2.93	1.11	0.00	0.42	0.15	11.52
P65	5	5.54	28.97	10.12	4.29	3.49	3.66	84.50
P148	9	8.01	36.57	14.17	11.74	3.69	2.75	438.08
P205	11	5.16	40.35	11.47	9.10	2.64	3.24	840.50
Average RPD		4.10	24.05	8.08	5.83	2.09	**2.03**	-

5. Conclusion and future research

Worker assignment is a new problem in assembly line balancing problems, which arises due to different skills of the worker. This research considers the integrated worker assignment and balancing problem in two-sided assembly line systems with the objective of optimizing cycle time. Three variants of the artificial bee colony (ABC) algorithm are developed to solve this new problem, where the first variant is the original ABC algorithm and two latter variants are improved editions. Since no benchmark problems are available, a total number of 156 cases are generated ranging from P9 with 9 tasks to P205 with 205 tasks. To evaluate the performance of the proposed algorithms, three other algorithms are implemented and compared. Computational results show that the two improved ABC algorithms are the two best performers and produce promising results for this new problem. Future researchers might consider more constraints in real application, such as positional constraint and zoning constraint. It is also interesting to

develop more advanced algorithm such co-evolutionary algorithms to solve the problem.

References

1. Miralles, C., Garcia-Sabater, J.P., Andres, C., Cardos, M.: Advantages of assembly lines in sheltered work centres for disabled. A case study. International Journal of Production Economics 110, 187-197 (2007)
2. Miralles, C., García-Sabater, J.P., Andrés, C., Cardós, M.: Branch and bound procedures for solving the assembly line worker assignment and balancing problem: Application to sheltered work centres for disabled. Discrete Applied Mathematics 156, 352-367 (2008)
3. Bartholdi, J.: Balancing two-sided assembly lines: a case study. The International Journal Of Production Research 31, 2447-2461 (1993)
4. Li, Z., Tang, Q., Zhang, L.: Two-sided assembly line balancing problem of type I: Improvements, a simple algorithm and a comprehensive study. Computers & Operations Research 79, 78-93 (2017)
5. Tang, Q., Li, Z., Zhang, L.: An effective discrete artificial bee colony algorithm with idle time reduction techniques for two-sided assembly line balancing problem of type-II. Computers & Industrial Engineering 97, 146-156 (2016)
6. Relich, M., Świc, A., Gola, A.: A knowledge-based approach to product concept screening. In: Distributed Computing and Artificial Intelligence, 12th International Conference, pp. 341-348. Springer, (Year)
7. Relich, M., Bzdyra, K.: Knowledge discovery in enterprise databases for forecasting new product success. In: International Conference on Intelligent Data Engineering and Automated Learning, pp. 121-129. Springer, (Year)
8. Chaves, A.A., Miralles, C., Lorena, L.A.N.: Clustering search approach for the assembly line worker assignment and balancing problem. In: Proceedings of the 37th international conference on computers and industrial engineering, Alexandria, Egypt, pp. 1469-1478. (Year)
9. Li, Z., Tang, Q., Zhang, L.: Minimizing the cycle time in two-sided assembly lines with assignment restrictions: improvements and a simple algorithm. Mathematical Problems in Engineering (Article ID 4536426) :1-15,
10. Tapkan, P., Ozbakir, L., Baykasoglu, A.: Modeling and solving constrained two-sided assembly line balancing problem via bee algorithms. Applied Soft Computing 12, 3343-3355 (2012)
11. Li, Z., Janardhanan, M.N., Tang, Q., Nielsen, P.: Co-evolutionary particle swarm optimization algorithm for two-sided robotic assembly line balancing problem. Advances in Mechanical Engineering 8, 1687814016667907 (2016)
12. Özcan, U., Toklu, B.: Balancing of mixed-model two-sided assembly lines. Computers & Industrial Engineering 57, 217-227 (2009)
13. Kim, Y.K., Song, W.S., Kim, J.H.: A mathematical model and a genetic algorithm for two-sided assembly line balancing. Computers & Operations Research 36, 853-865 (2009)

http://www.springer.com/978-3-319-62409-9

Distributed Computing and Artificial Intelligence, 14th International Conference

Omatu, S.; Rodríguez, S.; Villarrubia, G.; Faria, P.; Sitek, P.; Prieto, J. (Eds.)

2018, XVIII, 344 p. 137 illus., Softcover
ISBN: 978-3-319-62409-9