Date of publication SEP-30, 2021, date of current version MAY-21, 2021.
www.computingonline.net / computing@computingonline.net

Print ISSN 1727-6209
Online ISSN 2312-5381
DOI 10.47839/ijc.20.3.2289

A Hybrid Grasp-genetic Algorithm for Mixed-model Assembly Line Balancing Problem Type 2

LAKHDAR BELKHARROUBI, KHADIDJA YAHYAOUI
University of Mascara Algeria, 29000 Mascara,
(e-mail: lakhdar.belkharroubi@univ-mascara.dz, khadidja.yahyaoui@univ-mascara.dz)

Corresponding author: Lakhdar Belkharroubi (e-mail: lakhdar.belkharroubi@univ-mascara.dz).

ABSTRACT In manufacturing systems, mixed model assembly lines are used to produce different products to deal with the problem of customers' demands variety, and minimizing the cycle time in such assembly line is a critical problem. This paper addresses the mixed model assembly line balancing problem type 2 that consists in finding the optimal cycle time for a given number of workstations. A hybrid Greedy randomized adaptive search procedure-Genetic algorithm is proposed to find the optimal assignment of tasks among workstations that minimize the cycle. A Ranked Positional Weight heuristic is used in the construction phase of the proposed GRASP, and in the local search phase, a neighborhood search procedure is used to ameliorate the constructed solutions in the construction phase. The GRASP is executed many times in order to seed the initial population of the proposed genetic algorithm, and the results of the executions are compared with the final solutions obtained by the hybrid GRASP-GA. In order to test the proposed approaches, a numerical example is used.

KEYWORDS Mixed model assembly line; Cycle time; GRASP; Genetic Algorithm; RPW; Neighborhood search.

I. INTRODUCTION

IN production systems, traditional or simple assembly lines are used to fabricate a single product, and from the literature, SALBP for (Simple assembly line balancing problem) is a famous problem that is related to this type of lines including two different versions SALBP-1 and SALPB-2. The SALBP-1 consists to minimize the number of workstations for a given cycle time, and the SALBP-2 consists to minimize the cycle time for a given number of workstations [1].

Nowadays, the variation of market demands increases more and more, thus, to provide models diversity and meet customers' demands in time, many industrial companies have changed simple assembly lines by another type known by mixed model assembly lines [2]. Unlike simple assembly lines, mixed model assembly lines are used to assemble different models having similar characteristics [3].

The difference between models lies in the number of tasks, processing time of each task, precedence tasks relations and the amount of production [4]. The lot size of each model during the production is equal to one as shown in Fig. 1 [5]. According to [6], mixed assembly lines are better than simple assembly lines in terms of product quality, system flexibility and lead time. Mixed model assembly lines are used in several industries including for example furniture and clothing, automobile industries, aerospace industry, refrigerators and washing machines [7, 8].

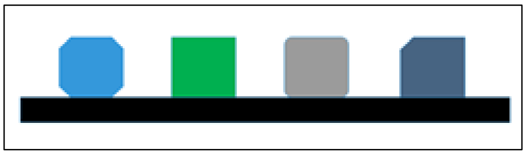
Figure 1. Mixed model assembly line

From the literature, different methods have been developed by researchers to solve different problems related to mixed model assembly lines. The two famous problems are: Mixed Model Assembly Lines Balancing Problem type 1 (MiMALBP-1) and Mixed Model Assembly Lines Balancing Problem type 2 (MiMALBP-2). In MiMALBP-1 the objective is to minimize the number of workstations for a given cycle time, and in MiMALBP-2 the objective is to minimize the cycle time for a given number of workstations [9]. The MiMALBP is more complex than the SALBP because the line must be balanced and in addition, the sequence of models must be determined to maximize the line efficiency [10].

Generally, before solving the MiMALB problem, it can be transformed into the simple version (SALBP) by combining all precedence graphs of models into a single graph as shown in Fig. 2. The final graph is known in the literature by the combined precedence graph [5].

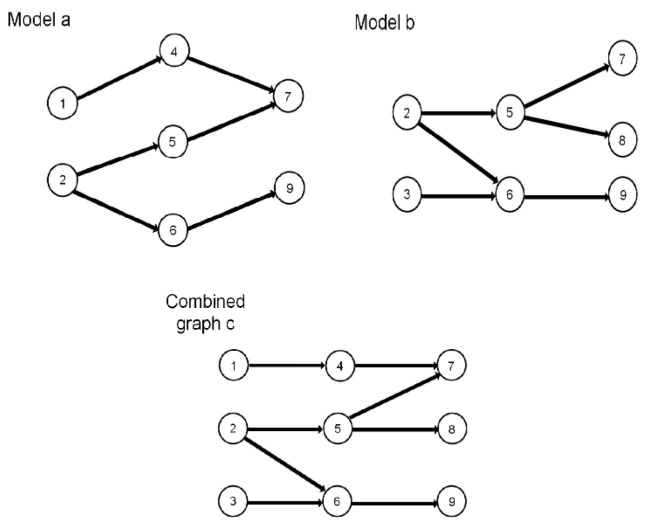

Figure 2. Combined precedence graph

In this work we propose a hybridization of two metaheuristics Greedy randomized adaptive search procedure and Genetic algorithm to solve the MiMALBP-2 and we show the ability of the proposed GA in finding better solutions. The adoption of this approach is motivated by that the greedy algorithm proved its efficiency to solve some optimization problems [11] and from our best knowledge, the proposed hybridization GRASP-GA including the RPW in the construction phase with the random concept to solve the MiMALBP-2 is never used before.

The rest of the paper is organized as follows. Section 2 presents related work. Section 3 defines the system model and problem formulation. Section 4 details the proposed approach. Section 5 represents the numerical example. Section 6 evaluates and discusses the obtained results. Section 7 concludes the paper and highlight future works.

II. RELATED WORKS

Since the first publication of the mathematical formulation of the assembly line balancing problem, several methods have been developed in order to find feasible solutions. In the literature, these methods are classified into two categories: exact methods and inexact methods known by heuristics [12]. The ALB problem is NP-hard [2], thus exact methods are not useful to solve it because they take a long time to find an optimal solution especially for problems with big sizes. Inexact methods or heuristics are more useful in this situation by finding a feasible solution in a reasonable time, but there is another problem in heuristics methods known by the local optimum which means the solution found may not be the optimal one. Other methods known by metaheuristics can avoid the local optimum, these methods try to improve successively the initial solution towards a global optimum [13].

Researchers have developed different methods to solve the MiMALB problem with aim to minimize the cycle time. For example, an IT tool has been developed [2] to help engineers in controlling manufacturing resources, and in addition to increase the production rate using three heuristics: Largest candidate rule (LCR) method, Kilbridge and Wester Column (KWC) method and Ranked positional weight that combines the strategies of LCR and KWC methods. A hybrid genetic algorithm (HGA) has been proposed in [9] to solve a robust mixed model ALB problem type 2 with uncertain task times. A heuristic method has been used in this algorithm to seed the initial population, in addition, an adaptive local search and a discrete levy flight have been hybridized with this HGA to enhance its performance.

A robotic mixed model assembly line balancing problem type 2 has been studied in [14]. In the robotic line, a set of robots performs different tasks. One of the objectives aimed in this paper is minimizing the cycle time, and to solve the general problem two different multi-objective evolutionary algorithms have been used. The first algorithm is a multi-objective particle swarm MOPSO and the second algorithm is a non-dominated sorting genetic algorithm NSGA-2. A comparison has been carried out shows that the NSGA-2 in most cases is better than the MOPSO in terms of performance. Another iterative procedure has been proposed in [15] for solving mixed model assembly line with parallel workstations using a genetic algorithm. The main objective is to maximize the production rate of the line for a predefined number of operators. In addition, the proposed GA can be used to minimize the number of workstations.

An ant colony optimization approach for solving the mixed model assembly line balancing problem with setup time between operations have been proposed in [16]. This ACO is based in on learning permutations of the operations in contrast to previous ACO used for the assembly balancing problems. Authors found that the proposed ACO leads to excellent results in short times. A reactive generalized simulated annealing (GSA) using type 2 fuzzy controller to tackle with the MiMALBP-2 have been proposed in [17], the fuzzy type 2 controller is used to guide the algorithm to explore the entire search space in early iterations which allows the algorithm to achieve higher performance and accuracy. Result shows that the fuzzy GSA can obtain better solution when compared with the original counterpart and hierarchical approach.

III. PROBLEM DESCRIPTION AND MATHEMATICAL FORMULATION

A. PROBLEM DESCRIPTION

In the mixed model assembly line. Similar models are assembled in the line according to a specific random sequence. Each model has a set of tasks and each task has a determined processing time (or task time). All tasks are regrouped in a graph known as the precedence relations graph. All tasks must be assigned to workstations taking into consideration precedence relations between these tasks, thus, a task cannot be assigned before its predecessors. The sum of assigned tasks times to the same workstation must not exceed the cycle time (workstation time). The mixed model assembly line balancing problem can be transformed into a simple assembly line balancing problem by combining all precedence relations graphs of models into only one precedence graph, and the average task processing time for each task is calculated. So, the aim of solving the MiMALBP-2 is to find the best assignment of tasks with a minimum cycle time for a fixed number of workstations respecting the precedence relations. Finding the optimal sequence of models is not taken into consideration in this paper.

B. MATHEMATICAL FORMULATION

The aim is to maximize the production rate that means minimizing the cycle time. The following mathematical formulation of the problem was proposed by Baybars [18] to solve SALBP-2 and is used to solve MiMALBP-2 to find the minimum cycle time:

n	*number of operations*
i	*task i where i =1,2, …n*
m	*number of stations*
k	*station k where k =1,2, …m*
t_i	*processing time of task i*
p_i	*the set of predecessors of task i*

X_{ik} = 1 if the task i is assigned to the workstation k, otherwise X_{ik} = 0

The main objective is to minimize the cycle time as shown in the following equation:

$$\mathbf{min\ C} \quad , \qquad (1)$$

Under the following constraints:

$$\sum_{k=1}^{m} \mathbf{X_{ik} = 1} \ , \qquad (2)$$

for *i = 1, 2…n*

$$\sum_{i=1}^{n} \mathbf{t_i * X_{ik} \le C} \ , \qquad (3)$$

for *k = 1, 2...m*

$$\sum_{k=1}^{m} \mathbf{k * X_{hk}} \le \sum_{k=1}^{m} \mathbf{k_{,}* \ X_{ik}} \qquad (4)$$

where $h \in P_i$

$$X_{ik} \in \{0,1\}. \qquad (5)$$

Equation (2) ensures that each task is affected only once. Equation (3) obliges that the sum of process times of tasks assigned to the same workstation must be less than or equal to the cycle time. Equation (4) imposes the precedence relations between tasks. If the task ***h*** must be performed before task ***i***, thus, the index of the station where task ***h*** is affected must be less than or equal to the index of the station where task ***i*** affected. Finally, Equation (5) represents the constraint of decision variables integrity. The objective function (1) can be written as follow:

$$\mathbf{min\ max} \left\{ \sum_{i}^{n} \mathbf{t_i \ * \ X_{ik}} \ \middle| \ \boldsymbol{k = 1, 2, \dots m} \right\}, \qquad (6)$$

Using the equation (6) the formulation no longer represents a linear program in integers and the problem become easier to solve.

IV. PROPOSED METHODS TO SOLVE THE MIXED MODEL ALB PROBLEM TYPE 2

In this paper, a hybridization of two metaheuristics, a genetic algorithm (GA) and the GRASP (Greedy Randomized Adaptive Search Procedure) is proposed to solve the mixed model assembly line problem type 2. The GRASP is used to seed the initial population of the GA, so each individual is a different solution found using GRASP. The proposed hybridization is shown in Fig. 3.

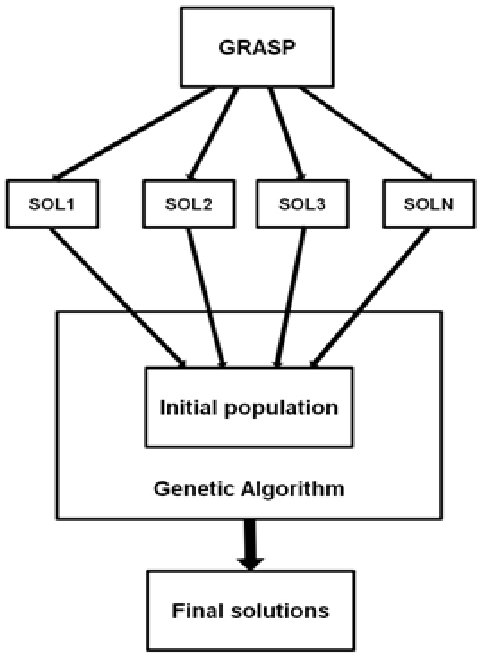

Figure 3. GA-GRASP hybridization

A. PROPOSED GENETIC ALGORITHM

The genetic algorithm is an effective popular metaheuristic, it has been used to solve different assembly line balancing problems [9]. This metaheuristic starts with initial solutions (population), each solution (individual) is evaluated after the calculation of its fitness using a fitness function F. Best solution are selected to undergo operators (crossover and mutation). Finally, generated individuals by the genetic operators are evaluated. All these GA steps are repeated for a chosen number known as the number of generations [15, 10]. In the proposed GA we used the Elitism technique [19] by keeping 25% best individuals from the total population in each generation to undergo GA operators and to guarantee that the best solutions remain until the final population. The GA stages are described as follows:

1- Encoding of solutions:

The sequence of tasks (solution) is created based on the precedence graph, and in this paper, the priority-based encoding method is used. The value of the gene represents the task node, and the position of this node in the sequence represents its priority of assignment.

2- The initial population:

The initial population is a set of individuals, each individual represents the final solution found using the Greedy randomized adaptive search procedure.

3- Fitness function:

To evaluate individuals, the fitness function is calculated for each individual. In this GA the fitness function is based on the cycle time because the aim is to maximize the production rate. The fitness function is given by

$$F = 1/C,$$

where C represents the cycle time.

4- Selection:

In order to select the best individuals, tournament selection is used. In each tournament two individuals are chosen randomly from the remaining 75% of individuals, individual with minimum cycle time is chosen to undergo GA operators (crossover and mutation). The process of selection is repeated until the required number of individuals in the mating pool is reached.

5- Crossover:

To generate a new offspring, the one-point crossover is used:

- Step1: from selected individuals in the selection operation, two parents are chosen.
- Step2: one-point crossover is applied between two parents to create new offspring as shown in Fig. 4.
- Step3: repeated tasks (genes) in the chromosome must be replaced by missing tasks.

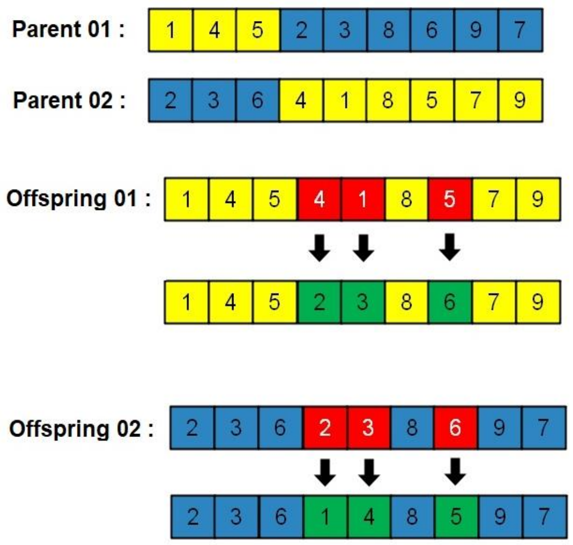

Figure 4. One-point crossover

1- Mutation:

Swap mutation is used in which two genes (tasks) are chosen randomly and their values are interchanged. The process of mutation is applied on some individuals chosen randomly to generate new sequences. In the end, unfeasible solutions must be corrected by reordering tasks that don't respect precedence relations.

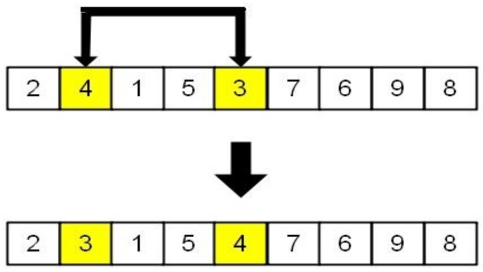

Figure 5. Swap mutation

B. GRASP ALGORITHM

The GRASP for Greedy Randomized Adaptive Local Search is a metaheuristic that has been used to solve different combinatorial problems. In this metaheuristic, each iteration consists of two principal phases: the construction phase and the local search phase. The role of the construction phase is to build a feasible solution, and the local search phase tries to find a better neighborhood solution from the solution found in the construction phase [11].

To find a feasible solution in the construction phase, RPW (ranked positional weight) is used, it is a popular heuristic that has been used to solve the assembly line balancing problem. In this heuristic, a list is made in which tasks are arranged in descending order based on their positional weights. The positional weight is calculated by adding all other durations attributed to the successors to the duration of the chosen task [20].

Each iteration in the construction phase is based on two lists, the Candidate List (CL) and the Restricted Candidate List (RCL) [21]. First, the Candidate List contains all tasks, and from this list, the RCL is created by choosing tasks with the maximum positional weight and respect all constraints (precedence relations and available workstation time). The number of elements of the Restricted Candidate List is limited by the *p* elements with the best positional weight. From the RCL, a task is chosen randomly for the assignment. Chosen task is deleted from the RCL and the CL and the workstation time is updated.

The construction phase stops when all tasks in the CL are assigned and the given number of workstations is respected. Otherwise, the process of the construction phase restarts again until a feasible solution is found with the exact given number of workstations.

The solution found in the construction phase maybe not the optimal one, and for this reason, the local search is used to ameliorate the constructed solution. Neighborhood search is used in the local search phase to find an optimal neighborhood solution by changing randomly the positions of two tasks in the sequence respecting precedence relations. The cycle time of the new sequence is calculated to make a comparison with the sequence found in the construction phase. This process in the local search phase stops when no other best neighborhood solution is found.

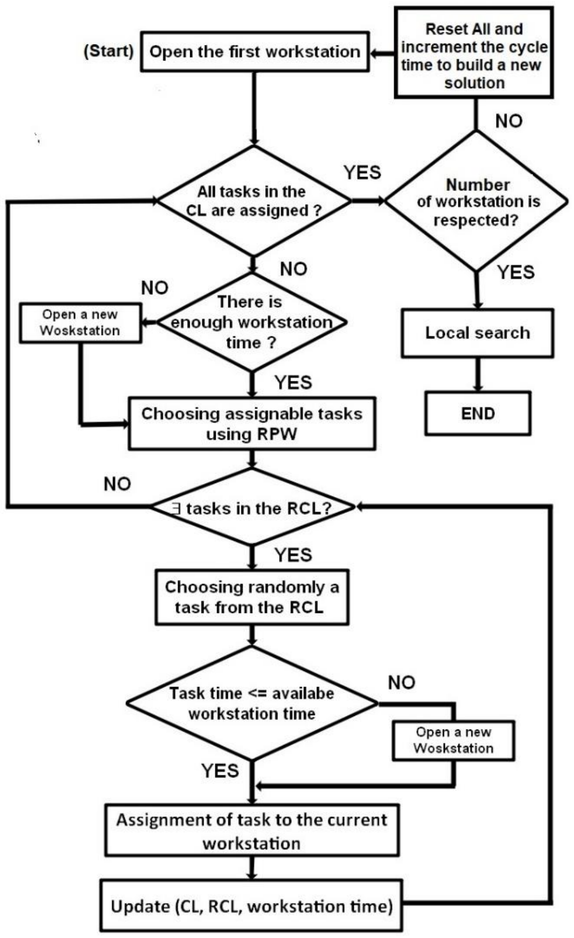
(Start)
Open the first workstation
Reset All and increment the cycle time to build a new solution
NO
YES
All tasks in the CL are assigned ?
Number of workstation is respected?
YES
NO
NO
There is enough workstation time ?
Open a new Woskstation
Local search
YES
END
Choosing assignable tasks using RPW
NO
∃ tasks in the RCL?
YES
Choosing randomly a task from the RCL
Task time <= availabe workstation time
NO
Open a new Woskstation
YES
Assignment of task to the current workstation
Update (CL, RCL, workstation time)

Figure 6. Adopted GRASP approach for MiMALBP resolution

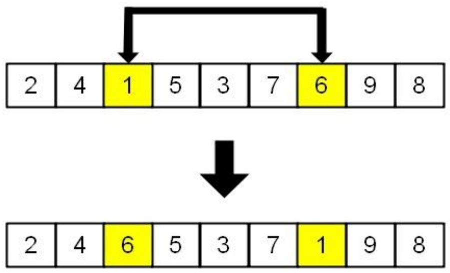

Figure 7. Generation of new neighborhood solution

V. NUMERICAL EXAMPLE

The proposed hybrid method is implemented with Python 3.7.3 on a PC with intel(R) Core (TM) i3-4005U CPU 1.70 GHz, and tested on an example that represents a mixed model problem with two models A and B. Each model has its precedence relations between tasks, and each task may have a different processing time in each model. Table (1) and table (2) represent models A and B data respectively.

The first column represents the task number, the second column represents the task time and in the last column, represents immediate predecessors. For each model, some tasks are not included in the assemblage.

Table 1. Model A data

Task number	Task Time	Immediate predecessors
1	9	-
2	21	-
3	25	-
4	14	1
5	23	2
6	12	3
7	11	4, 5
8	7	5
9	20	5, 6
10	4	7, 8

Table 2. Model B data

Task number	Task Time	Immediate predecessors
1	3	-
2	25	-
4	19	1
5	17	2
7	7	4, 5
8	15	5
9	8	5
10	13	7, 8
11	17	9
12	13	11

To solve this mixed model assembly line problem, we transformed it into a simple problem by combining graphs of models A and B in one graph as shown in Fig. 8. For each common task between models, the average processing time is calculated and unnecessary relations are deleted. The problem contains 12 tasks and 12 precedence relations that must be respected during the assignment of tasks into workstations. As shown in Fig. 8, Values inside circles are tasks number and T is the average processing time. So, the aim is to find the minimum cycle time based on the number of workstations. In this example, the number of workstations is 4.

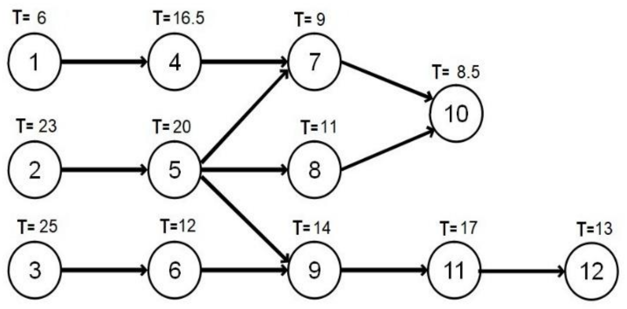

Figure 8. Combined precedence graph

Table 3. GRASP-GA parameters

	Parameters	Values
Grasp	RCL (number of elements)	3
	Number of iterations	10
GA	Number of generations	100
	Population size	20
	Crossover probability	0.5
	Mutation Probability	0.15
	Elitism	25%

VI. RESULTS AND DISCUSSION

Fig. 9 represents the application of the GRASP on the proposed example. In order to create the initial population, the GRASP has been executed 20 times, and the result of each execution is a different feasible solution.

As shown each solution includes the solution found in the construction phase and its best neighborhood generated by the local search procedure. 8 best solutions (S1, S5, S7, S9, S10, S12, S14, S17) have been found with cycle time (c = 47.5). In some cases (S7, S11, S13, S18), no better neighborhood solutions were detected in the local search phase.

Fig. 10 shows the result obtained by the hybridization of the two proposed metaheuristics (GRASP and Genetic algorithm). All neighborhood solutions found in the 20 executions using GRASP were used as an initial population in the genetic algorithm, and after 100 generations, 8 better solutions (S1, S2, S3, S4, S5, S11, S15, S17) have been found with cycle time (c= 45). Best found solutions differ in the sequence of tasks as shown in Table 4.

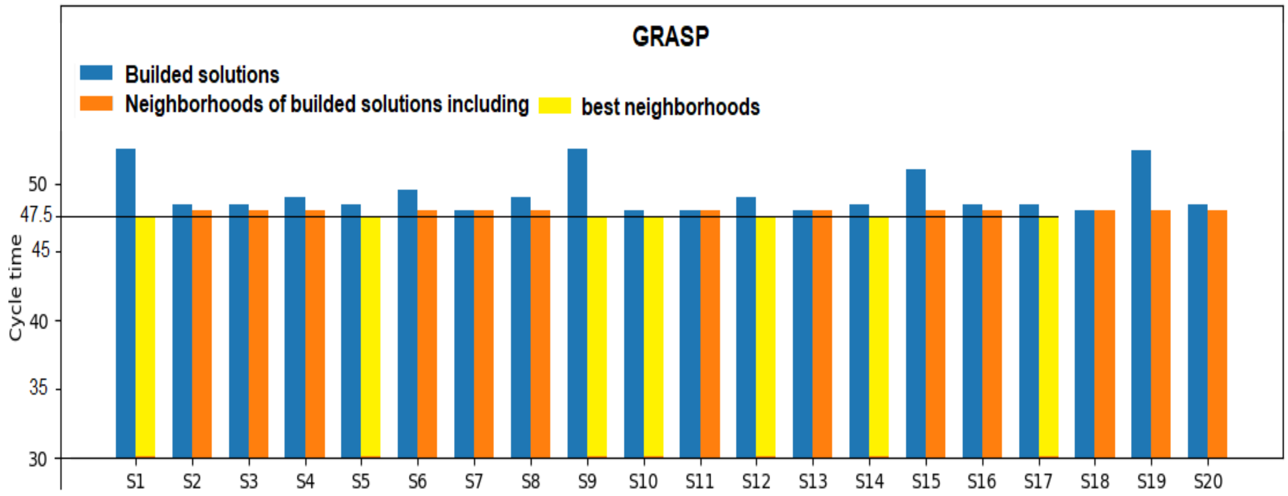

Figure 9. Obtained solutions using GRASP

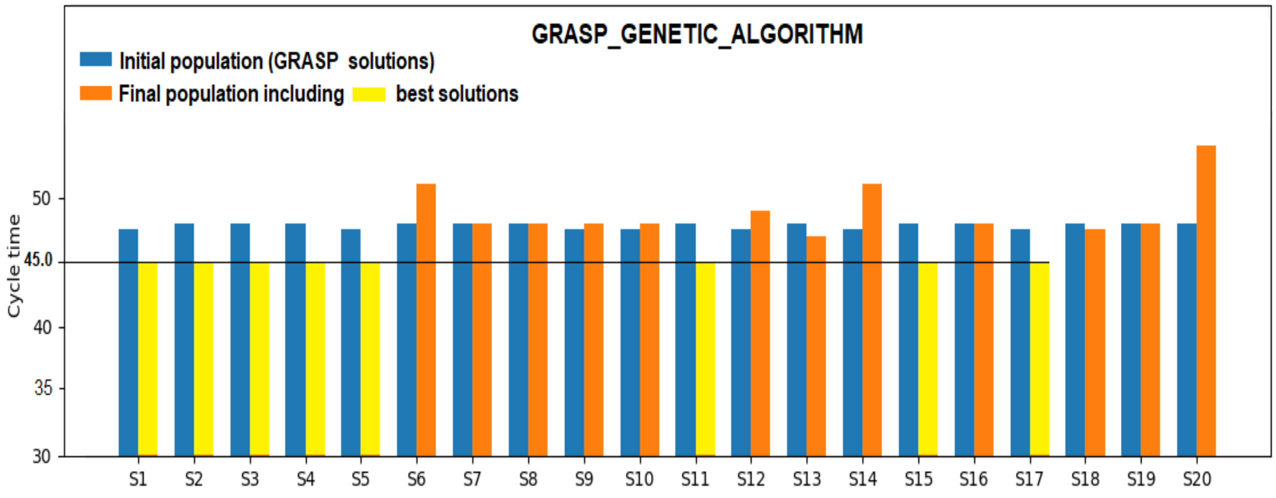

Figure 10. Obtained Solutions using GRASP-GA

Table 4. Best sequences found in the final population

Solutions	Sequences
S1	2 5 3 6 1 9 11 12 8 4 7 10
S2	2 5 1 3 6 9 11 12 4 7 8 10
S3	3 6 1 2 5 9 11 12 4 8 7 10
S4	2 5 1 3 6 9 11 12 8 4 7 10
S5	2 5 1 3 6 9 11 12 4 8 7 10
S11	2 5 3 1 6 9 11 12 4 7 8 10
S15	3 6 1 2 5 9 11 12 8 4 7 10
S17	2 5 3 1 6 9 11 12 8 4 7 10

Any solution from the 8 best solutions found in the final population can be chosen as a final solution because there is no difference between them only in the order of tasks. The first solution is chosen randomly as the final solution, and the table below shows the assignment of tasks to workstations according to the chosen solution.

Table 5. Assignment of tasks to workstations

Workstation	Task
1	2, 5
2	3, 6, 1
3	9, 11, 12
4	8, 4, 7, 10

Based on the chosen assignment, the workload can be calculated for each model, thus, for each workstation the sum of processing times of assigned tasks is calculated to find the workload, and like there is difference in processing times of common tasks between models, the workloads will be different, so the utilization of workstations during the production is not stable due to the variety of products.

Fig. 11 and Fig. 12 shows the workload for model A and B respectively, and Fig. 13 represents avreage workload. Also overload time and lead time were calculated.

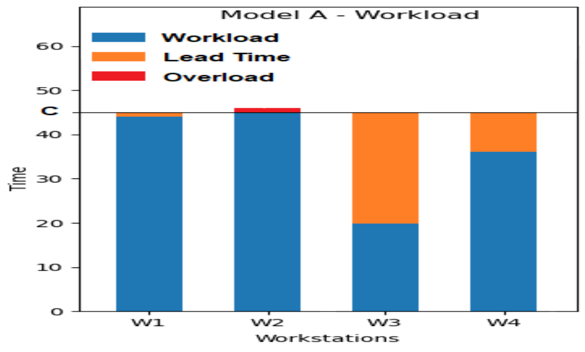

Figure 11. Model A workload

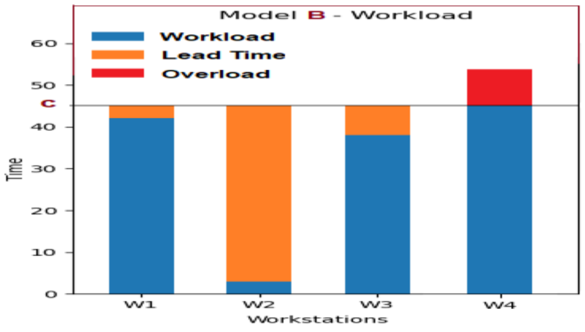

Figure 12. Model B workload

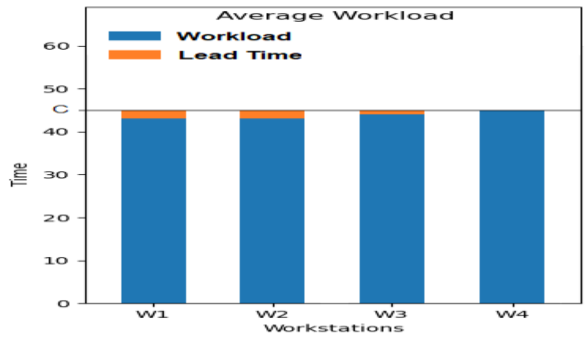

Figure 13. Average workload

The reader can observe that the cycle time is exceeded in workstation 2 for model A and in workstation 4 for model B, this problem results due to the variety of models and can influence the line efficiency, so to deal with this problem in mixed model assembly lines, the best sequence of product that can minimize the work overload must be determined. This problem is known in the literature by mixed model sequencing problem [22].

From the results, it is clear that the proposed greedy randomize adaptive search procedure was trapped in the local optima in different executions with (c = 47.5) in this example, and in some cases the local search procedure did not found a neighborhood solution that minimizes the cycle time. After the hybridization of the GRASP with the proposed GA by starting with all neighborhood solutions found using GRASP as the initial population, better solutions were found with a minimum cycle time (c = 45).

In solving the proposed numerical example, the genetic algorithm played an important role in finding new solutions that cannot be found using the proposed GRASP, and due to the existence of precedence relations constraints, the neighborhood search method was restricted because for each solution found by the construction phase, a neighborhood solution that differs only on positions of two tasks must be found, but with genetic algorithm operators (crossover and mutation) during generations, the possibility of finding new solutions becomes greater.

VII. CONCLUSION

In this paper, the mixed model assembly line balancing problem type 2 is addressed, and the objective is to find the best assignment of tasks among workstations to minimize the cycle time. A Greedy randomized adaptive search procedure based Ranked positional weight heuristic is proposed in order to seed the initial population of the genetic algorithm, and a numerical example that represents a mixed model assembly line that assembles two different products is used to test the proposed hybridization. Results show the efficiency of the proposed hybridization by improving the solutions found by GRASP using the GA. In the first stage, the proposed GRASP was trapped in the local optimal due to the usage of a fixed alpha value which cannot help the GRASP to expand the search space, but in the second stage, the genetic algorithm starts with the solutions found by the GRASP as an initial population to tackle with the GRASP drawback and as results avoid the local optimal problem.

In future work, the proposed approach can be developed using for example one from the GRASP enhancements such as the Reactive version, cost perturbations, bias functions, memory and learning, and local search on partially constructed solutions or using other local search methods in the local search phase of the GRASP that can avoid the local optimal problem. Also, using the main idea of the proposed approach, other versions of the Mixed-model assembly line balancing problem can be solved, for example, Mixed-Model Two-sided ALBP, Mixed-Model U-shaped ALBP, Mixed-Model Robotic ALBP and so on.

References

[1] P. Sivasankaran and P. Shahabudeen, “Literature review of assembly line balancing problems,” *Int J Adv Manuf Technol*, vol. 73, no. 9–12, pp. 1665–1694, 2014. https://doi.org/10.1007/s00170-014-5944-y.

[2] D. Krenczyk, B. Skolud, and A. Herok, “A heuristic and simulation hybrid approach for mixed and multi model assembly line balancing,” *Intelligent Systems in Production Engineering and Maintenance*, vol. 637, pp. 99–108, 2018. https://doi.org/10.1007/978-3-319-64465-3_10.

[3] A. Yadav, P. Verma, and S. Agrawal, “Mixed model two-sided assembly line balancing problem: an exact solution approach,” *Int J Syst Assur Eng Manag*, vol. 11, no. S2, pp. 335–348, 2020. https://doi.org/10.1007/s13198-020-00956-1.

[4] M. M. Razali, M. F. F. Ab. Rashid, and M. R. A. Make, “Mathematical modelling of mixed-model assembly line balancing problem with

resources constraints," *IOP Conf. Ser.: Mater. Sci. Eng.*, vol. 160, p. 012002, 2016. https://doi.org/10.1088/1757-899X/160/1/012002.

[5] J. I. van Zante-de Fokkert and A. G. de Kok, "The mixed and multi model line balancing problem: a comparison," *European Journal of Operational Research*, vol. 100, no. 3, pp. 399-412, 1997. https://doi.org/10.1016/S0377-2217(96)00162-2.

[6] O. Banjo, D. Stewart, and M. Fasli, "Optimising mixed model assembly lines for mass customisation: A multi agent systems approach," *Advances in Manufacturing Technology*, vol. 3, pp. 337-342, 2016.

[7] T. Ziarnetzky, L. Mönch, and A. Biele, "Simulation of low-volume mixed model assembly lines: Modeling aspects and case study," *Proceedings of the 2014 Winter Simulation Conference*, 2014, pp. 2101-2112. https://doi.org/10.1109/WSC.2014.7020055.

[8] B. Yuan, C. Zhang, X. Shao, and Z. Jiang, "An effective hybrid honey bee mating optimization algorithm for balancing mixed-model two-sided assembly lines," *Computers & Operations Research*, vol. 53, pp. 32–41, 2015. https://doi.org/10.1016/j.cor.2014.07.011.

[9] J.-H. Zhang, A.-P. Li, and X.-M. Liu, "Hybrid genetic algorithm for a type-II robust mixed-model assembly line balancing problem with interval task times," *Adv. Manuf*, vol. 7, pp. 117-132, 2019. https://doi.org/10.1007/s40436-019-00256-3.

[10] P. Su and Y. Lu, "Combining genetic algorithm and simulation for the mixed model assembly line balancing problem," *Proceedings of the Third International Conference on Natural Computation*, 2007, pp. 314-318. https://doi.org/10.1109/ICNC.2007.306.

[11] M. G. C. Resende and C. C. Ribeiro, "Greedy randomized adaptive search procedures," *HandBook of Metaheuristics*, vol. 272, pp. 169-220, 2003. https://doi.org/10.1007/978-3-319-91086-4_6.

[12] G. Abdul-Nour, H. Beaudoin., P. Ouellet, R. Rochette, S. Lambert, "A reliability based maintenance policy: a case study," *Computers and Industrial Engineering*, vol. 35, no. 3- 4, pp. 591-594, 1998. https://doi.org/10.1016/S0360-8352(98)00166-1.

[13] P. R. McMullen, & G. V. Frazier, "A heuristic for solving mixed-modelline balancing problems with stochastic task durations and parallel stations," *International Journal Production Economics*, vol. 51, pp. 177-190. 1997. https://doi.org/10.1016/S0925-5273(97)00048-0.

[14] M. Rabbania, Z. Mousavia, and H. Farrokhi-Aslb, "Multi-objective metaheuristics for solving a type II robotic mixed-mode assembly line balancing problem," *Journal of Industrial and Production Engineering*, vol. 33, no. 7, pp. 472-482, 2016. https://doi.org/10.1080/21681015.2015.1126656.

[15] A. S. V. Raj, J. Mathew, P. Jose, and G. Sivan, "Optimization of cycle time in an assembly line balancing problem," *Procedia Technology*, vol. 25, pp. 1146–1153, 2016. https://doi.org/10.1016/j.protcy.2016.08.231.

[16] D. Thiruvady, A. Nazari, and A. Elmi, "An ant colony optimisation based heuristic for mixed-model assembly line balancing with setups," *Proceedings of the 2020 IEEE Congress on Evolutionary Computation (CEC)*, Glasgow, United Kingdom, 2020, pp. 1–8. https://doi.org/10.1109/CEC48606.2020.9185757.

[17] M. Lalaoui and A. E. Afia, "A versatile generalized simulated annealing using type-2 fuzzy controller for the mixed-model assembly line balancing problem," *IFAC-PapersOnLine*, vol. 52, no. 13, pp. 2804–2809, 2019. https://doi.org/10.1016/j.ifacol.2019.11.633.

[18] I. Baybars, "A survey of exact algorithms for the simple assembly line balancing problem," *Management Science*, vol. 32, no. 8, pp 909-932, 1986. https://doi.org/10.1287/mnsc.32.8.909.

[19] H. Du, Z. Wang, W. Zhan and J. Guo, "Elitism and distance strategy for selection of evolutionary algorithms," *IEEE Access*, vol. 6, pp. 44531-44541, 2018. https://doi.org/10.1109/ACCESS.2018.2861760.

[20] R. O. Edokpia and F. U. Owu, "Assembly line re-balancing using ranked positional weight technique and longest operating time technique: A comparative analysis," *AMR*, vol. 824, pp. 568–578, 2013. https://doi.org/10.4028/www.scientific.net/AMR.824.568.

[21] R. Martí, M. G. C. Resende, and P. M. Pardalos, *Handbook of Heuristics*, Springer, pp. 741-758, 2018. https://doi.org/10.1007/978-3-319-07124-4.

[22] N. Boysen, M. Kiel, and A. Scholl, "Sequencing mixed-model assembly lines to minimise the number of work overload situations," *International Journal of Production Research*, vol. 49, no. 16, pp. 4735–4760, 2011. https://doi.org/10.1080/00207543.2010.507607.

__Belkharroubi Lakhdar__ Is pursuing his Ph.D. in Information and communication technologies at University Mustapha Stambouli of Mascara, Algeria. Previously, he received a Master's degree in Networks and systems from University Abd Elhamid Ibn Badis of Mosatagnem, Algeria, in 2019. His research interests include Optimization, Artificial Intelligence, complex Assembly lines and Manufacturing Systems, Industry 4.0.

__Yahyaoui Khadidja__ is an Associate Professor at University Mustapha Stambouli, Mascara, Algeria, she is currently a research member of the computer science laboratory at Mascara University. She received an engineering degree in industrial computing from Oran university computer science department, Algeria, in 2000 and 2006, respectively. She received a Ph.D. degree in computer science in 2013 from Oran University, Algeria. Her main research interests include optimization, artificial intelligence, manufacturing System, Next-generation networking, 5G networks, and Information-Centric Networking.