International Journal of Computational Intelligence Systems
Vol. **14(1)**, 2021, *pp.* 1438–1450
DOI: https://doi.org/10.2991/ijcis.d.210420.002; ISSN: 1875-6891; eISSN: 1875-6883
https://www.atlantis-press.com/journals/ijcis/

Research Article

Applying Meta-Heuristics Algorithm to Solve Assembly Line Balancing Problem with Labor Skill Level in Garment Industry

Gary Yu-Hsin Chen[1], Ping-Shun Chen[2], Jr-Fong Dang[3,*], Sung-Lien Kang[4], Li-Jen Cheng[2]

[1]*Department of Logistics Management, National Kaohsiung University of Science and Technology, Yanchao District, Kaohsiung City, 824, Taiwan, Republic of China*
[2]*Department of Industrial and Systems Engineering, Chung Yuan Christian University, Chung Li District, Taoyuan City, 320314, Taiwan, Republic of China*
[3]*Department of Industrial Engineering and Systems Management, Feng Chia University, Xitun District, Taichung City, 40724, Taiwan, Republic of China*
[4]*Division of Library and Information Affairs, Chihlee University of Technology, , Banqiao District, New Taipei City, 220305, Taiwan, Republic of China*

ARTICLE INFO

Article History
Received 20 Sep 2020
Accepted 14 Apr 2021

Keywords
Assembly line balancing problem
Toyota sewing system
Tabu search
Simulated annealing

ABSTRACT

This research investigates how to properly place garment industry workers to work stations in the assembly line to achieve a more balanced production and to reduce the production cycle time. We simulate the assembly line balancing problem via staff assignments. In our research, we conduct a comparative case study and implement our own simulation. The experiments are designed with both single- and multitasking modes. Each experiment is carried out for 10 runs. Finally, we compare our results obtained among constructive greedy, tabu search and simulated annealing. We find that tabu search algorithm is better than simulated annealing on the problem of staff assignment. Meanwhile, we also observe that if we adjust 30% labor force from single task into multitasking mode, the assembly line performance deteriorates. This case is accentuated for workers with disparate skill levels for different tasks.

1. INTRODUCTION

Nowadays, the garment manufacturing industries are facing numerous challenges and intense market competition. For example, the industry is struggling with a urgent demand to raise the production efficiency while reduce the costs to gain the competitive edge. Furthermore, garment industries can be characterized with the mass-produced process and labor-intensive production. As the consequence, manufacturers would need to plan well the task placement and staff allocation at the early stage as to meet the customers' requirements. Usually, the production process of garment manufacturing industries is executed in a sequential manner. Figure 1 shows that the general process can consist of the raw material selection, material inspection, sample production, cutting, sewing, ironing, packing, and so on. Please note that the steps from cutting to ironing are mostly labor-intensive. Also, the sequential relationship between the steps enables the production process to be treated as the assembly line model.

In the garment industry, the production quantity per day has to comply with the daily order in the industry. As a result, the assembly line layout can be a major factor and greatly impacts the production output. Additionally, the load distribution at each assembly line workstation may affect productivity. Therefore, the output rate of the assembly line would determine the balancing condition for each workstation and further influence production costs and customer satisfaction. The assembly line balancing (ALB) problem has been studied by enterprises for many decades. The ALB model ensures that the staff assignment balances the whole production process in order to effectively reduce the production time or idle time. To meet the ALB, employees' mastery of skills at each task would be considered as an indicator. However, there are few studies investigating into the multifunctional (multitasking) workers with multiple levels of skills working at work stations. Our research incorporates the concept of Toyota Sewing System (TSS) derived from the Toyota Production System (TPS) for clothing or footwear industry. TSS is credited with less floor space, flexibility and better working environment [1]. TSS is featured with a U-shaped assembly line and teams of workers making garments on a single-piece flow basis. Employees move between tasks and pass pieces of garments to the next employees as soon as they become available to process more products. The Toyota sewing machine system is shown in Figure 2 where sewing machines are represented by rectangle symbols while employees are labeled alphabetically.

As mentioned earlier, the labor-intensive feature of garment industry would be studied. Chan *et al.* [2] state that the skill and experience of supervisors can be treated as the important factor in controlling the production line and therefore improve the performance. Similar, Guo *et al.* [3] point out that the performance of the job shop mainly depends on supervisor's experience and knowledge. Ağrali *et al.* [4] develop their employee scheduling model considering the employee skill levels. To solve the problem within a reasonable amount of time, Ağrali *et al.* [4]

[*] *Corresponding author. Email: jfdang@fcu.edu.tw*

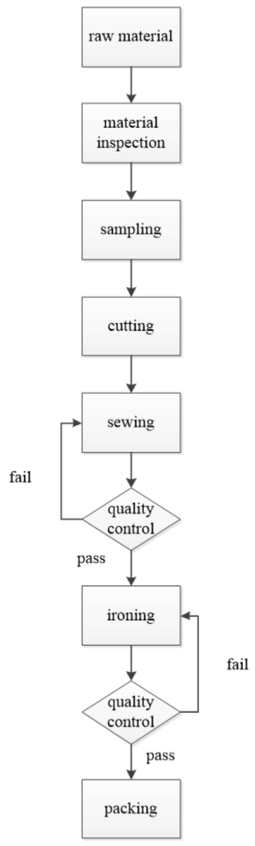


Figure 1 Garment manufacturing production process.

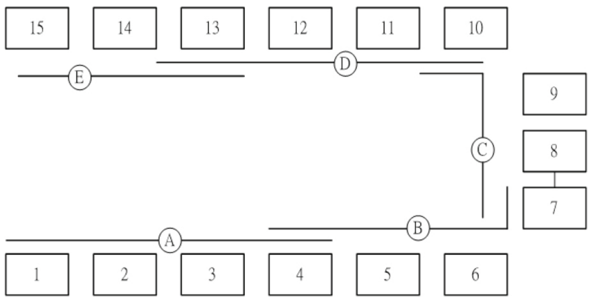


Figure 2 Toyota sewing machine planning system schematic.

considers reformulation strategy. Dekena *et al.* [5] claim that the employees place the great impact on productivity and flexibility of the production system. They develop the production plan adopting the employee competence. Chen *et al.* [6] highlight that there is growing pressure in the skilled labor market. Chen *et al.* [6] proposes the model where the technicians must complete the given tasks within the time period and then gain the experiences. As such, technicians can be assigned to the job in the future. Wang *et al.* [7] study the problem of the employee competence in precast production planning. Furthermore, they propose a competence-based model to optimize the worker allocation. Chen *et al.* [8] address a multi-skill project scheduling problem for IT product development. In their research [8], the project is divided into multiple projects which are completed by a skilled employee. To solve the scheduling problem, they proposed a multi-objective nonlinear mixed integer programming model. This research takes into consideration of employees' skill proficiency at performing tasks, multifunctional employees and cell formation to minimize the production cycle time. Also, we adopt another manner to calculate the cycle time different from the previous studies and further consider the workers' skills to reflect the real-world situation. We find that the production time can be effectively reduced with a better personnel assignment and a preferred mode of production system.

2. RELATED WORKS

In this section, there are two main parts: ALB and TPS together with its extension TSS. The first part would introduce ALB problem and meta-heuristic algorithm. Then the TPS and TSS would provide readers with the production process approaches. Based on the aforementioned research, we develop our solution algorithm and enhance the solution procedure.

2.1. Assembly Line Balancing

Assembly line is an arrangement of workers, machines and equipment in which the product being assembled passes consecutively from operation to operation until completed. Furthermore, the ALB can be considered as one of the classical problems in assembly line and occurs frequently in the manufacturing industry [9,10]. Suer [11] states that ALB operates with precedence relationship to assign a number of operators or machines to each operation of an assembly line to meet the required production rate with a minimum idle time and to keep a leveled workstation time at each operation. In general, ALB includes Simple Assembly Line Balancing Problem (SALBP) and General Assembly Line Balancing Problem (GALBP). The SALBP can be divided into several categories [12]. That is, SALBP – 1, SALBP – 2, SALBP – E and SALBP – F. Note that the objective function of those SALBP is to minimize the idle time at the assembly line. As a result, researchers usually intend to obtain a total minimal idle time at each workstation in the assembly line under different production models or resource allocation. Regarding GALBP, Scholl and Klein [12] categorize it as Mixed ALB Problem (MALBP/MSP) and U-type ALB Problem (UALBP). Furthermore, ALB problem and its variants are considered to be NP-hard. This encourages one to apply meta-heuristic algorithms to derive the approximately optimal solutions due to its feature reducing solution space to accelerate the solution process. There are several meta-heuristic algorithms such as genetic algorithm (GA), tabu search (TS), simulated annealing (SA), and so on. Chan *et al.* [2] solve SALBP-E in garment industry by utilizing GA. In their solution procedure, they consider the skill level of the workers at each machine station and adopt the filtering mechanism of GA to properly assign workers to different station to reduce the total idle time and improve the production efficiency. Lin [13] employs GA to derive the optimal solutions of the shortest movement path of the single-row machine layout problem. The author applies a two-dimensional matrix to mark the movement distance of each machine and its sequence priority. Then the movement distance and suitability degree can be obtained by mathematical calculations of permutations and combinations. Suwannarongsri and Puangdownreong [14] propose a TS method to reduce the work load for the solving of the ALB problem. Given the cycle time and the number of workstations being fixed, the proposed method can not only maximize workstation load reduction but also cut down the idle time if machines are sectorized into different workstations based on the TS method. Dinh and Nguyen [15] develop three different approaches, exhaustive search, SA and SA with greedy, to solve ALB problem and further indicate that SA is the best one in terms of both accuracy and running time. Xu *et al.* [16] adopt an adaptive ant colony (AAC) algorithm with modifications to solve ALB problem. They define an objective function for minimizing the smoothness index measuring the difference among worktime and the targeted cycle time of each workstation on an assembly line. Quyen *et al.* [17] propose a hybrid genetic algorithm (HGA) to solve the resource -constrained assembly line balancing problem (RCALBP) in the sewing line of a footwear manufacturing plant. In their algorithm, the initial solution can be derived by the priority rule-based method (PRBM) in the first stage and then the solution is obtained by GA in the second stage. Gansterer and Hartl [18] consider balancing problems of one- and two-sided assembly lines with real-world constraints. The GA is utilized to solve the problem and then compare the performance of GA with TS by a set of larger test cases. They further show that GA outperforms a specific differential evolution (DE) algorithm and TS. Dang and Pham [19] utilize discrete event simulation (DES) to depict the performance measure and then integrate the adaptive large neighborhood search (ALNS) heuristic into the model to derive the solution. The authors conduct a footwear manufacturing factor with real data as the case study to see the proposed algorithm capability. Pereira [20] enhances the multi-Hoffmann heuristic to solve the ALB problem and further shows that its performance better than previous studies based on the results. He also validates the algorithm by implementing the model in the clothing company. Bautista *et al.* [21] propose their model maximizing the comfort of operators in mixed-model assembly lines. They further utilize mixed integer linear programming (MILP) and greedy randomized adaptive search procedure (GRASP) to derive the solutions. The authors state that even though the MILP provides the best solution, the results obtained by GRASP are comparative. As we can see, there are lots of papers solving the ALB problem by employing the meta-heuristic method. In this paper, we take a different approach of involving employee assignment to calculate cycle time in our research for garment industry, which differs from the existing approach of calculating cycle time. Also, the factor of workers' skills is considered to reflect the real world.

2.2. Toyota Production System

In our research, we adopt TPS as one of our production process approaches to solve the ALB problem due to its U-shape feature. Pujo *et al.* [22] indicate that U-shaped layout is always at least efficient than an equivalent linear cell. Additionally, the TPS is developed by Taiichi Ohno, Shigeo Shingo and Eiji Toyoda between 1948 and 1975. The main objectives of TPS are to design for reducing cost, improving productivity, and thus eliminating wastes such as excess inventory, redundant human resources, and so on.

2.2.1. Toyota Sewing System

TSS is the extension of the just-in-time (JIT) production concept. TSS assists the garment industry in accepting the cellular manufacturing concept. The worker serving at the U-shape production line can operate multiple machines concurrently. The workers normally move counterclockwise to operate in a cell. When a worker completed his/her own jobs while his/her previous coworker delayed, this worker would adjust the direction and move clockwise to help the previous worker. Therefore, we may need to keep the production in the cells smooth and flexible. TSS production method is shown as Figure 3 [1].

In Figure 3, TSS assigns 3 to 5 workers to operate 13 machine stations. The workers can execute their jobs following the production process introduced above while achieve an effective balanced status. The workers can collaborate each other flexibly to minimize the idle time [1].

3. RESEARCH METHODS

In this section, we would introduce the limitations, formulation and the meta-heuristics algorithms applied in the model.

3.1. Research Definition and Limitations

In this research, we focus on the assignment problem of multiple tasks and different workers' skill proficiency. Based on TPS, we modify one part of production process to the cell layout to solve the minimal cycle time of work.

3.1.1. The basic assumption

This study aims to solve the ALB problem in the garment industry. We make some assumptions about the issues and limitations for our research due to the cellular manufacturing concept. The basic assumptions and limitations are presented in the following bullets:

- Workers have a specified skill proficiency for each task.
- Each task is associated with the standard allowable minutes for the example of male short-sleeved shirt in the garment industry.
- Operating processes observe the precedence relationship.
- The worker's operating routes cannot be crossed in each subcell.
- Some production processes are not necessary classified as multitasking as depicted in the precedence relationship graph.

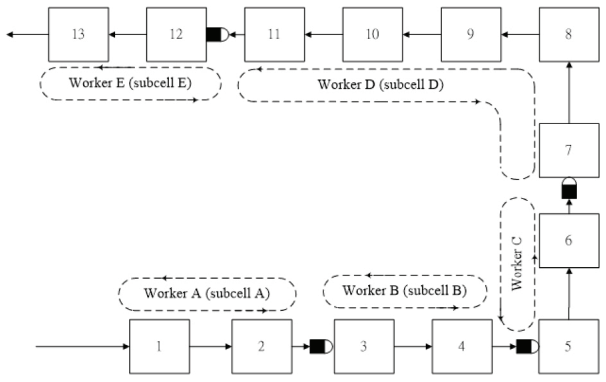


Figure 3 TSS production method [1].

3.2. Mathematical Model

In our mathematical model, the objective function is to minimize the maximum cycle time.

$$\min C = \sum_{k=1}^{n} \max \sum_{j=1}^{m} t_j m_{jk} \tag{1}$$

Subject to

$$\sum_{k=1}^{m} \sum_{l=1}^{n} w_{kl} = 1 \tag{2}$$

$$\sum_{k=1}^{n} w_{ik} \le \sum_{k=1}^{n} w_{jk} \ \left(1 \le i, j \le n\right) \wedge \left(i \in p_j\right) \tag{3}$$

$$m_{jk} \in \{0, 1\} \ \left(1 \le j \le n\right) \wedge (1 \le k \le m) \tag{4}$$

t_j: the completion time of task j

m_{jk}: task j assigned to work station k

C: cycle time

p_j: task i precedes task j

In (1), we calculate the working hours for each workstation and select a largest working time as the cycle time. Then our study selects a minimum working cycle time of the combinations. Equation (2) is the constraint that each work station must be assigned to at least one worker. Equation (3) states the preceding relationship of tasks. For example, task i must be operated before task j; we use work stations to sectorize the positions of tasks that task i is in the work station that must be processed prior to that work station where task j is located. Equation (4) indicates the 0–1 variable; m_{jk} is 1 if task j is assigned to work station k, 0 otherwise.

3.3. Cycle Time Calculation

In this research, the calculation of the cycle time is the main objective. One can derive the cycle time by selecting the longest completion time among the work stations as the completion time of the entire work. When we calculate the working cycle time, we have to make sure the number of tasks, work stations and working time for each task.

3.3.1. Placement of tasks

Based on the assumption as described earlier, we start to place tasks into the stations according the precedence relationship of the tasks. This approach is in accordance with the "most followers" decision rules of assigning the work element or task to a work station to satisfy the precedence requirements [23]. For example, task 1 is performed before task 3, while task 2 before task 4; task 5 is operated later than all tasks. The precedence relationship of tasks is shown as Figure 4.

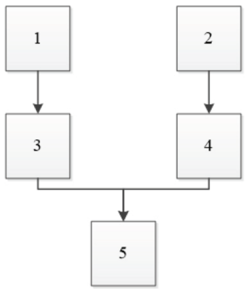


Figure 4 Precedence relationship of task arrangement.

Once we have the above relationship among tasks, we put tasks into each station. We put 5 tasks into three different stations. First, we put the tasks without predecessors into a station. That is, we put task 1 and task 2 into station 1, and then we may put succeeding tasks of 3 and 4 into the next workstation. In the final stage, we put the last task 5 into the third station. Next, we calculate the completion time at each station, and the calculating of the cycle time is to select the longest station completion time as the cycle time. The layout of the tasks may not be optimized initially for cycle time reduction. The tasks at different work stations would then be readjusted to reduce the cycle time. During the process of readjustment, the precedence relationship must be observed among tasks. For example, we utilize Figure 4 to illustrate that task 3 or task 4 may be moved to work station 1. However, it is subject to the constraint that the rearranged tasks must be moved to any station where their predecessors are located or to any station which follows the predecessors' work stations. For example, task 3 may be put at the original station 2 or be placed into station 1, which also includes task 1. However, suppose task 1 is moved station 2, task 3 must be placed at either station 2 or station 3; otherwise, such task movement violates the precedence constraint defined initially among tasks.

After a series of readjustment, we may get a near-optimal arrangement of task placement resulting in the shortest cycle time. However, in our case study, tasks at each station are stationery right from the beginning—the rearrangement occurs with the worker reassignment because the worker's performance impacts the production line very much.

3.3.2. Worker assignment and working hour calculation

Due to the workforce's varied working experiences, their skill proficiency would be affected. It follows that, whenever we assign workers, we should choose employees with high skill proficiency to be assigned in advance. Under the constraint of one worker, one station, we compare all tasks first to find an efficient assignment. After the staff assignment process, we calculate the workers' completion time which is related to their working proficiency. In accordance with the reality, workers' completion times vary; therefore, we divide the standard completion time at each task by workers' skill proficiency in our research. The formula of the adjusted processing time is shown below:

$$a_i = \frac{t_i}{w_{ij}} \tag{5}$$

where $i, j = 1, 2, \ldots, n$

a_i: adjusted processing time

t_i: standard allowable minutes of task i

w_{ij}: skill level of worker j for performing task i

3.4. Toyota Sewing System

TSS also known as modular production system, was proposed to improve highly labor-intensive industries by switching from the single-task to multitask production mode. In this study, we apply the cell manufacturing concept which can help workers to enter the one-to-many, multitasking production mode in the garment industry. We take a page from TSS derived from TPS is developed for garment or footwear industries to eliminate unnecessary waste and improve production efficiency.

Based on the original placement approach of TPS, we devise a new labor staff assignment model with characteristics outlined as the following:

(1) Select work station randomly.

(2) Set the standard completion time of the work station as the standard time of labor staff assignment.

(3) Choose workers based on their skill proficiency of operating tasks according to the standard above.

(4) Once the workers are assigned, begin calculating the completion time of each station.

3.4.1. Computing completion time of workstation

After the workers are assigned into stations, we can calculate the work time of each task according to the workers' skill proficiency of operating the tasks and then calculate the total time of each station. For example, suppose that six workers with various level of skill proficiency for different tasks work at an assembly line where four work stations exist; additionally, workers operate within a standard completion time of tasks and stations. We use the method introduced in this section to calculate the completion time of each station as the following:

(1) Add up the standard allowable minutes or completion time of all tasks, for example, 72 minutes in total, and then divided by the number of work stations to get the number of 18 minutes each.

(2) The time obtained from step 1 is set as the standard; if the processing time of a task selected is not over 18 minutes, we must add up the processing time of the next tasks. Once the processing time of the task(s) selected are over 18 minutes, we assign a worker to work on those two tasks.

(3) The strategy to select workers to task depends on their skill proficiencies of completing tasks.

(4) A chosen worker can only be assigned to only one workstation.

We summarize the above case study according to the calculation method, three workers' assignments exist: tasks 1 and 2 are assigned to worker 4; tasks 3 and 4 are assigned for worker 1; tasks 5 and task 6 are assigned to worker 2. We can add more work in and we may compute actual completion time of each station after the workers are assigned, and then calculate the objective function value—the cycle time.

3.5. Heuristics and Meta-Heuristics Approaches

In the following sections, three proposed approaches are introduced to solve the ALB problem with workers' performances for each task: constructive greedy (CG) algorithm, TS and SA.

3.5.1. CG algorithm steps

The CG algorithm is developed by selecting the most proficient worker for each task starting from the first task to the last. Once the most skilled worker is determined, the standard allowable minutes are adjusted by formula (5). As all workers have been assigned to those tasks, the tasks then are assigned to work stations with the restrictions of not exceeding the average cycle time of each work station and following the precedence relationship. Later, the cycle time of each station is calculated and the minimum cycle time of the assembly process is determined.

3.5.2. TS steps

In the following, we discuss the TS to solve the ALB problem in the garment industry with the objective of minimizing the cycle time. We describe the algorithmic steps in the following sections.

Step 1. The workers are randomly assigned to tasks, and the adjusted processing time of each worker at each task is computed with respect to each individual's skill proficiency.

Step 2. Initialize the Tabu list. At the beginning stage of the calculation, the Tabu list, the record-keeping matrix for potential candidate solution, is empty. Whenever a new candidate solution exists, it goes into the Tabu list; the solution cannot be reconsidered until it reaches an expiration point. A best solution variable is also initialized to store the current best solution.

Step 3. At each iteration, a number of random swapping is conducted. If a randomly selected worker is better than the current worker assigned to the based on the skill proficiency, the workers' swapping occurs. After the swapping, the new solution will be compared to the current best solution or the initial solution. If the new solution is better, the swapping sequence is recorded. Additionally, the updated current best solution is entered into the Tabu list as a "taboo" and not considered for a certain number of iterations.

Step 4. At the end of each iteration, the expiration time is decremented for each solution in the Tabu list; when the expiration time reaches zero for a particular solution, the solution is formerly considered as a "taboo" and can be revisited.

Step 5. At each iteration, a better solution would be found and recorded. If the newer solution is better than current best solution,

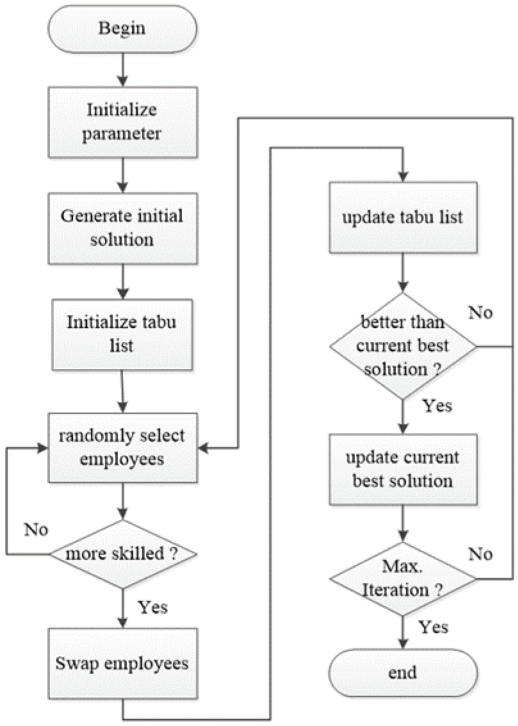


Figure 5 Tabu search for assembly line balancing problem with workers' assignment.

the solution will replace the existing solution. The iterations will be run until a certain number of iterations has been performed.

Step 6. After the best current solution is found for the minimum complete processing time, we assign each station to their respective workstation one-by-one according to the "most-followers" to satisfy the precedence requirement. We then compute the work time at each station and then select the longest work time as the initial solution. For example, suppose that 7 tasks, 7 workers, and 3 work stations are sequenced as {1, 2, 3}, {4, 5}, {6, 7} and the workers' assignment sequence numbers as {2, 5, 3}, {6, 1}, {7, 4}; in another word, task l is assigned to worker 2; task 2 is assigned to worker 5, task 3 worker 3 in work station 1 and so on. The processing time at each station is presented as the following, where t_i is the standard allowable time and w_{ij} is the skill proficiency of workers to tasks.

$$\text{station1} = t_1/l_{21} + t_2/l_{52} + t_3/l_{33}$$

$$\text{station2} = t_4/l_{64} + t_5/l_{15}$$

$$\text{station3} = t_6/l_{76} + t_7/l_{47}$$

The flowchart of TS is shown below in Figure 5.

3.5.3. SA steps

SA is a meta-heuristic, which imitates the annealing process in metallurgy by starting with the initial high temperature and slowly cooling down coupled with the Metroplis random sampling. The aim of the SA is to discover the global near-optimal solution. The algorithmic steps of SA are described as following and shown as Figure 6.

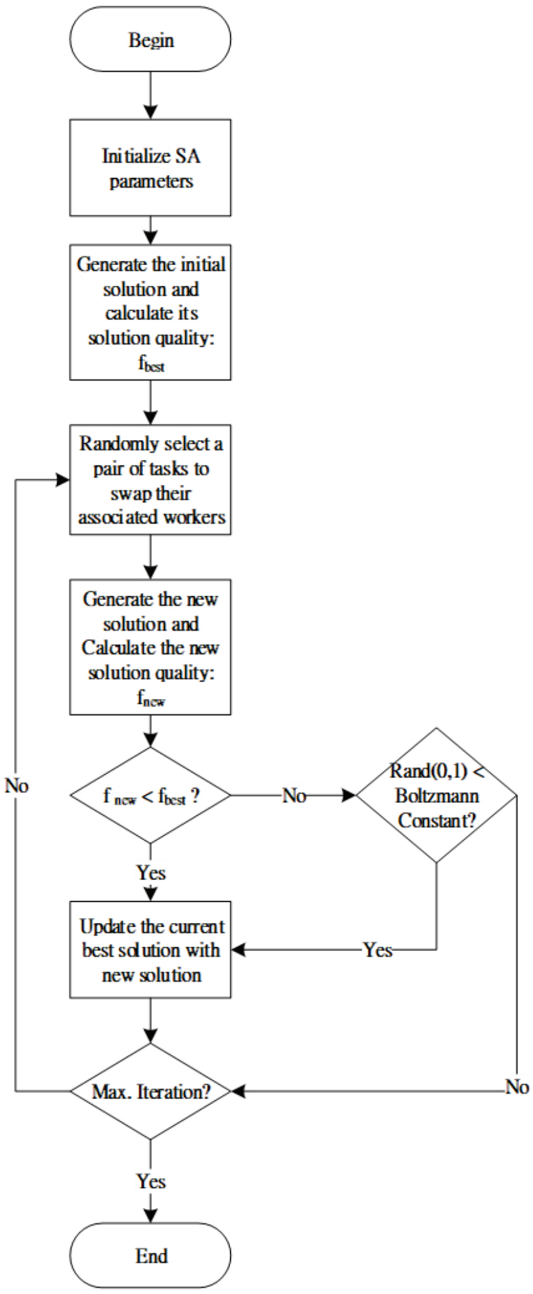

Begin
Initialize SA parameters
Generate the initial solution and calculate its solution quality: f_{best}
Randomly select a pair of tasks to swap their associated workers
Generate the new solution and Calculate the new solution quality: f_{new}
$f_{new} < f_{best}$?
No
Rand(0,1) < Boltzmann Constant?
Yes
Update the current best solution with new solution
Yes
Max. Iteration?
No
No
Yes
End

Figure 6 Simulated annealing for assembly line balancing with worker's assignment.

Step 1. Similar to TS algorithm aforementioned, the workers are randomly assigned to tasks, and the adjusted processing time of each worker at each task is computed with respect to each individual's skill proficiency.

Step 2. We Initialize the SA parameters: λ and T_0. λ corresponds to the proportion of lowering the temperature while T_0 corresponds to the initial temperature.

Step 3. We generate the initial solution, which is assigned as the current best solution, f_{best}, and calculate the solution's fitness value or quality.

Step 4. We randomly select a pair of tasks for exchanging their corresponding workers.

Step 5. After the exchange, the new solution is generated and calculated its fitness or solution quality.

Step 6. Check the new solution, f_{new}, is less than the current best solution, f_{best}. If this is the case, the current best solution is updated with the new solution; otherwise, a random float number between (0, 1) is generated and compared against the Boltzmann constant. If the random value is less than the Boltzmann constant, the current best solution is updated with the new solution even though the new solution is no better than the current best solution. This type of update corresponds to the Metroplis random sampling strategy with the purpose to explore the global search.

Step 7. The algorithm is repeated until the maximum iteration is reached; in this case, the temperature has been lowered below the defined final temperature.

Step 8. Once the near-optimal solution of workers' assignment to tasks is determined, the next step is to assign the tasks to the work stations with the restriction of not exceeding the average cycle time for every work station and following the precedence relationship.

4. RESULTS AND ANALYSIS

We would analyze and discuss the resulting outcomes of our proposed model in this section.

4.1. Research Environment

The research environments are as the following.

- We wrote the algorithms in Java.
- We executed the simulation under the Eclipse integrated development environment.
- The PC where the simulation occurred consists of Intel (R) Core (TM) i3 CPU 2GB RAM PC.

4.2. Experiments and Data Analysis

Chan *et al.* [2] discuss the issues of task planning and personnel's skill proficiency for men's short-sleeved T-shirt production in the garment industry. In that paper, each task is associated with a standard allowable minute. However, Chan *et al.* [2] do not take into account of multitasking workers as introduced by the TPS. To better reflect the real-world scenario, thus we design two scenarios—single-task and multitask scenarios—where we assume that workers have arbitrary proficiencies at completing the tasks. The precedence relationship chart, the tables of 41-task processing time and workers' skill tables according to Chan *et al.* [2] are provided as the following.

The precedence relationship of tasks is as shown in Figure 7 [13], which can be further identified as the task-work station assignment in Figure 8 [13]. During the simulations we do not update the task positions; however, we do adjust and decide the personnel assignment to tasks to obtain a better work cycle time.

Lin [13] proposes the modular diagram by grouping tasks in the sub-assembly lines or branches of the precedence relationship [6] into modules or "work stations." For example, Tasks 1–6 are grouped into a work station and so on. This configuration in Figure 7 is used in the simulations of multitasking workers later discussed.

The processing times of tasks in the garment industry are measured in standard allowable minutes, which consists of the basic time of the particular task and slack time (e.g., the allowable time of untying and tying garment bundles). By reducing the slack time, therefore, improve the standard allowable minutes [9]. Table 1 displays tasks and their corresponding standard allowable minutes.

The workers' performances for each task are rated from 0 (no skill at all) to 1.5 (fully skilled) according to the workers' skill table provided by Chan *et al.* [2]. Due to the space limitation, we split up the 41-workers' skill table into several sub-tables (Tables 2a–2d).

4.2.1. Worker's assignment in the single-task mode

We first calculate the total processing time or standard allowable minutes of all tasks to be performed. In the example of the garment industry given by Chan *et al.* [2] the total processing time comes out to be 1,749 minutes. Next, we divide the total processing time by the number of work stations as indicated by the precedence relationship graph where each branch of the graph is an equivalent of a work station according to Chan *et al.* [2] and Lin [13]. The average processing time thus comes out to be roughly 135 minutes per

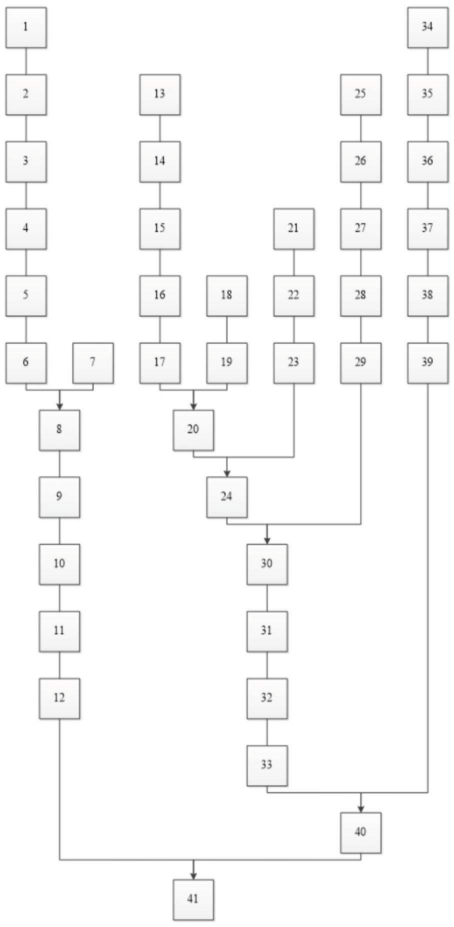
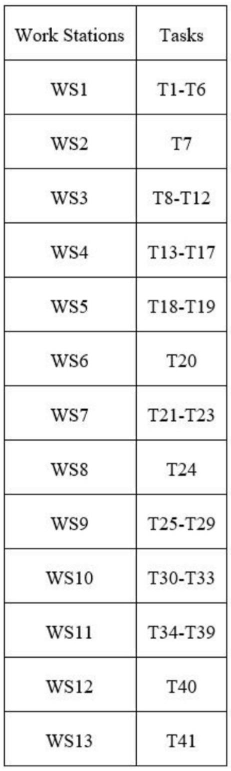


Figure 7 Task precedence relationship [13].

Table 1 41-task and their processing time [2].

Task #	Time$_{(SAM)}$	Task #	Time$_{(SAM)}$	Task #	Time$_{(SAM)}$	Task #	Time$_{(SAM)}$
1	41	12	41	23	34	34	28
2	30	13	37	24	55	35	57
3	25	14	19	25	47	36	49
4	48	15	28	26	67	37	50
5	30	16	30	27	44	38	55
6	65	17	29	28	36	39	32
7	28	18	47	29	65	40	43
8	46	19	66	30	58	41	48
9	57	20	54	31	64		
10	56	21	33	32	23		
11	32	22	20	33	32		



Work Stations	Tasks
WS1	T1-T6
WS2	T7
WS3	T8-T12
WS4	T13-T17
WS5	T18-T19
WS6	T20
WS7	T21-T23
WS8	T24
WS9	T25-T29
WS10	T30-T33
WS11	T34-T39
WS12	T40
WS13	T41

Figure 8 Task-work station assignment [13].

work station. The calculated average processing time is used as the soft time constraint for assigning tasks to work stations to achieve the work balance. However, if the processing times of some tasks go slightly above the average processing time, the solution is still acceptable.

Table 3a provides the statistics on the CG algorithm and 13 work stations (corresponding to 13 branches) with task-worker assignments. The total adjusted SAM corresponding to the adjusted standard allowable minutes obtained from formula (5). The min and max correspond to the minimum and maximum work station processing times, respectively. The Std. Dev. represents the standard deviation of all the work station processing times. Additionally, T*xx*-W*xx* represents the task and assigned worker numbers.

Furthermore, we also explore the possibility of assigning tasks to work stations based on the average processing time for the CG algorithm; that is, the accumulated processing time of tasks at a particular work station is less than the average processing time. Table 3b indicates the statistics of the work station processing time.

In Table 4, the parameter values used in the TS are presented. The values are determined through the empirical approach. The number of iterations is user-defined at 1000 iterations; the length of Tabu list is set at 1681, which corresponds to *number of workers* × *number of workers*; the Tabu move is set at 5—number of iterations or time period within which the same move recorded in the Tabu list are prevented from making.

In Table 5, the parameter values of SA are presented. Three parameters are selected: the maximum iteration, λ (the proportional temperature cooling schedule) and initial temperature.

The results of TS executed for single-task workers are given in Table 6. Note that 24 work stations are required in this instance; the cycle time comes out to be *132* minutes.

The results of SA executed for single-task workers are given in Table 7. Note that 21 work stations are required in this instance; the maximum cycle time comes out to be *183* minutes.

In the single-task worker scenario, we apply the CG algorithm, TS and SA algorithms to find solutions. The results indicate that in the single-task worker scenario, the CG algorithm for non-predetermined work stations performs the best among all the algorithms with shortest maximum cycle time and less work stations. The best solutions are obtained after 10 execution runs. Table 8 presents the comparison results of all algorithms executed in this paper.

4.2.2. Worker's assignment in the multitask scenario

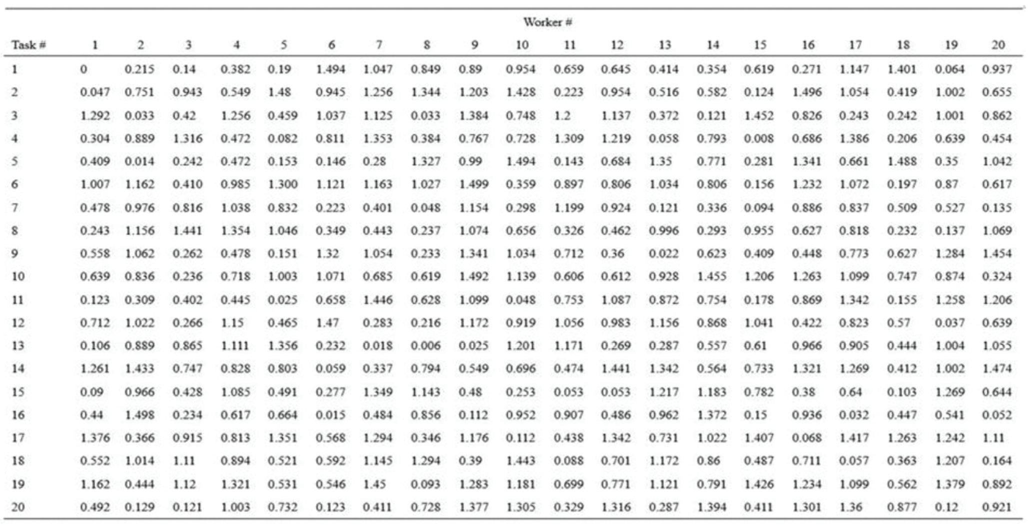
In this section, we adopt the idea of TSS and make 30% of workers operate through multitasking approach (one staff being assigned to multiple tasks). Assume that we have 41 tasks and thus we will have 13 multitasking workstations. The configuration of task and work stations are shown earlier in Figure 7. Assigning one individual worker to a work station is an unique challenging problem since a worker has a portfolio of various skill levels for different tasks.

The CG algorithm operates on the principle of finding the "most fit" worker for a particular task. This principle works well for a single task; however, for a work station with multiple tasks assigned with a single worker, the strategy fails to produce promising results. This is understandable for a worker with wildly fluctuating skill levels for different tasks. Table 9 presents this point. The processing time of 13 work stations varies wildly from minimum of 18.919 minutes to maximum of 6124.305 minutes. On the other hand, with their improving and probabilistic nature of TS and SA in the multitasking mode understandably perform better by rejecting inferior solutions and keeping promising ones.

Table 10 shows the statistics of work station processing time and task-worker assignment for the TS. The processing time of 13 work stations for TS varies from minimum of 605.017 minutes to maximum of 2769.603 minutes.

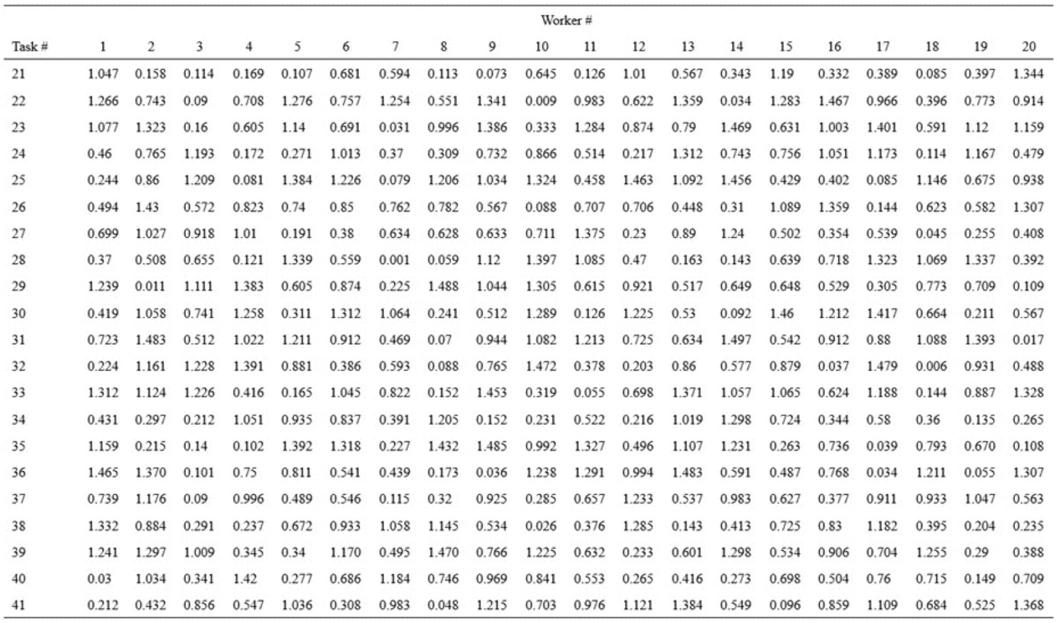
Table 11 shows the statistics of work station processing time and task-worker assignment for the SA search. The processing time of

Table 2a | Worker skill (No. 1–20) vs. Task (No. 1–20) [2].



	Worker #																			
Task #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	0.215	0.14	0.382	0.19	1.494	1.047	0.849	0.89	0.954	0.659	0.645	0.414	0.354	0.619	0.271	1.147	1.401	0.064	0.937
2	0.047	0.751	0.943	0.549	1.48	0.945	1.256	1.344	1.203	1.428	0.223	0.954	0.516	0.582	0.124	1.496	1.054	0.419	1.002	0.655
3	1.292	0.033	0.42	1.256	0.459	1.037	1.125	0.033	1.384	0.748	1.2	1.137	0.372	0.121	1.452	0.826	0.243	0.242	1.001	0.862
4	0.304	0.889	1.316	0.472	0.082	0.811	1.353	0.384	0.767	0.728	1.309	1.219	0.058	0.793	0.008	0.686	1.386	0.206	0.639	0.454
5	0.409	0.014	0.242	0.472	0.153	0.146	0.28	1.327	0.99	1.494	0.143	0.684	1.35	0.771	0.281	1.341	0.661	1.488	0.35	1.042
6	1.007	1.162	0.410	0.985	1.300	1.121	1.163	1.027	1.499	0.359	0.897	0.806	1.034	0.806	0.156	1.232	1.072	0.197	0.87	0.617
7	0.478	0.976	0.816	1.038	0.832	0.223	0.401	0.048	1.154	0.298	1.199	0.924	0.121	0.336	0.094	0.886	0.837	0.509	0.527	0.135
8	0.243	1.156	1.441	1.354	1.046	0.349	0.443	0.237	1.074	0.656	0.326	0.462	0.996	0.293	0.955	0.627	0.818	0.232	0.137	1.069
9	0.558	1.062	0.262	0.478	0.151	1.32	1.054	0.233	1.341	1.034	0.712	0.36	0.022	0.623	0.409	0.448	0.773	0.627	1.284	1.454
10	0.639	0.836	0.236	0.718	1.003	1.071	0.685	0.619	1.492	1.139	0.606	0.612	0.928	1.455	1.206	1.263	1.099	0.747	0.874	0.324
11	0.123	0.309	0.402	0.445	0.025	0.658	1.446	0.628	1.099	0.048	0.753	1.087	0.872	0.754	0.178	0.869	1.342	0.155	1.258	1.206
12	0.712	1.022	0.266	1.15	0.465	1.47	0.283	0.216	1.172	0.919	1.056	0.983	1.156	0.868	1.041	0.422	0.823	0.57	0.037	0.639
13	0.106	0.889	0.865	1.111	1.356	0.232	0.018	0.006	0.025	1.201	1.171	0.269	0.287	0.557	0.61	0.966	0.905	0.444	1.004	1.055
14	1.261	1.433	0.747	0.828	0.803	0.059	0.337	0.794	0.549	0.696	0.474	1.441	1.342	0.564	0.733	1.321	1.269	0.412	1.002	1.474
15	0.09	0.966	0.428	1.085	0.491	0.277	1.349	1.143	0.48	0.253	0.053	0.053	1.217	1.183	0.782	0.38	0.64	0.103	1.269	0.644
16	0.44	1.498	0.234	0.617	0.664	0.015	0.484	0.856	0.112	0.952	0.907	0.486	0.962	1.372	0.15	0.936	0.032	0.447	0.541	0.052
17	1.376	0.366	0.915	0.813	1.351	0.568	1.294	0.346	1.176	0.112	0.438	1.342	0.731	1.022	1.407	0.068	1.417	1.263	1.242	1.11
18	0.552	1.014	1.11	0.894	0.521	0.592	1.145	1.294	0.39	1.443	0.088	0.701	1.172	0.86	0.487	0.711	0.057	0.363	1.207	0.164
19	1.162	0.444	1.12	1.321	0.531	0.546	1.45	0.093	1.283	1.181	0.699	0.771	1.121	0.791	1.426	1.234	1.099	0.562	1.379	0.892
20	0.492	0.129	0.121	1.003	0.732	0.123	0.411	0.728	1.377	1.305	0.329	1.316	0.287	1.394	0.411	1.301	1.36	0.877	0.12	0.921

Table 2b | Worker skill (No. 1–20) vs. Task (No. 21–41) [2].



	Worker #																			
Task #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
21	1.047	0.158	0.114	0.169	0.107	0.681	0.594	0.113	0.073	0.645	0.126	1.01	0.567	0.343	1.19	0.332	0.389	0.085	0.397	1.344
22	1.266	0.743	0.09	0.708	1.276	0.757	1.254	0.551	1.341	0.009	0.983	0.622	1.359	0.034	1.283	1.467	0.966	0.396	0.773	0.914
23	1.077	1.323	0.16	0.605	1.14	0.691	0.031	0.996	1.386	0.333	1.284	0.874	0.79	1.469	0.631	1.003	1.401	0.591	1.12	1.159
24	0.46	0.765	1.193	0.172	0.271	1.013	0.37	0.309	0.732	0.866	0.514	0.217	1.312	0.743	0.756	1.051	1.173	0.114	1.167	0.479
25	0.244	0.86	1.209	0.081	1.384	1.226	0.079	1.206	1.034	1.324	0.458	1.463	1.092	1.456	0.429	0.402	0.085	1.146	0.675	0.938
26	0.494	1.43	0.572	0.823	0.74	0.85	0.762	0.782	0.567	0.088	0.707	0.706	0.448	0.31	1.089	1.359	0.144	0.623	0.582	1.307
27	0.699	1.027	0.918	1.01	0.191	0.38	0.634	0.628	0.633	0.711	1.375	0.23	0.89	1.24	0.502	0.354	0.539	0.045	0.255	0.408
28	0.37	0.508	0.655	0.121	1.339	0.559	0.001	0.059	1.12	1.397	1.085	0.47	0.163	0.143	0.639	0.718	1.323	1.069	1.337	0.392
29	1.239	0.011	1.111	1.383	0.605	0.874	0.225	1.488	1.044	1.305	0.615	0.921	0.517	0.649	0.648	0.529	0.305	0.773	0.709	0.109
30	0.419	1.058	0.741	1.258	0.311	1.312	1.064	0.241	0.512	1.289	0.126	1.225	0.53	0.092	1.46	1.212	1.417	0.664	0.211	0.567
31	0.723	1.483	0.512	1.022	1.211	0.912	0.469	0.07	0.944	1.082	1.213	0.725	0.634	1.497	0.542	0.912	0.88	1.088	1.393	0.017
32	0.224	1.161	1.228	1.391	0.881	0.386	0.593	0.088	0.765	1.472	0.378	0.203	0.86	0.577	0.879	0.037	1.479	0.006	0.931	0.488
33	1.312	1.124	1.226	0.416	0.165	1.045	0.822	0.152	1.453	0.319	0.055	0.698	1.371	1.057	1.065	0.624	1.188	0.144	0.887	1.328
34	0.431	0.297	0.212	1.051	0.935	0.837	0.391	1.205	0.152	0.231	0.522	0.216	1.019	1.298	0.724	0.344	0.58	0.36	0.135	0.265
35	1.159	0.215	0.14	0.102	1.392	1.318	0.227	1.432	1.485	0.992	1.327	0.496	1.107	1.231	0.263	0.736	0.039	0.793	0.670	0.108
36	1.465	1.370	0.101	0.75	0.811	0.541	0.439	0.173	0.036	1.238	1.291	0.994	1.483	0.591	0.487	0.768	0.034	1.211	0.055	1.307
37	0.739	1.176	0.09	0.996	0.489	0.546	0.115	0.32	0.925	0.285	0.657	1.233	0.537	0.983	0.627	0.377	0.911	0.933	1.047	0.563
38	1.332	0.884	0.291	0.237	0.672	0.933	1.058	1.145	0.534	0.026	0.376	1.285	0.143	0.413	0.725	0.83	1.182	0.395	0.204	0.235
39	1.241	1.297	1.009	0.345	0.34	1.170	0.495	1.470	0.766	1.225	0.632	0.233	0.601	1.298	0.534	0.906	0.704	1.255	0.29	0.388
40	0.03	1.034	0.341	1.42	0.277	0.686	1.184	0.746	0.969	0.841	0.553	0.265	0.416	0.273	0.698	0.504	0.76	0.715	0.149	0.709
41	0.212	0.432	0.856	0.547	1.036	0.308	0.983	0.048	1.215	0.703	0.976	1.121	1.384	0.549	0.096	0.859	1.109	0.684	0.525	1.368

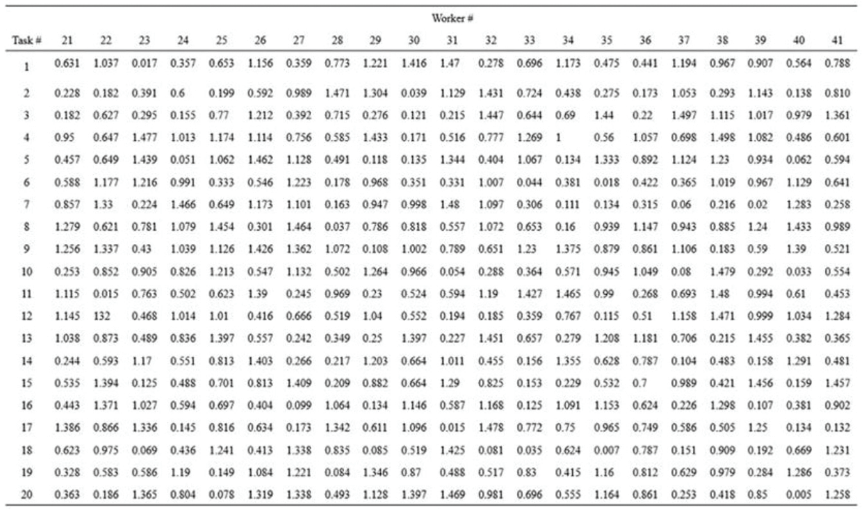
13 work stations for SA varies from minimum of 762.782 minutes to maximum of 4129.875 minutes.

The following Table 12, displays the statistics of execution results by three algorithms. The results indicate that the TS is the best among those three when it comes to multitasking. The reason behind this may be the random skills of workers make it harder to apply greedy approach of assigning a particular best worker to a particular work station since no worker has the best of all skills (skills are varied) for a particular work station. Since the TS and SA are both improvement algorithms, they fares better than the CG algorithm which operates only on finding the best worker for a single task. Nevertheless, because a single worker may not perform top-notched for all tasks for a work station with multiple tasks, the processing time suffers for the CG algorithm.

4.2.3. Discussion

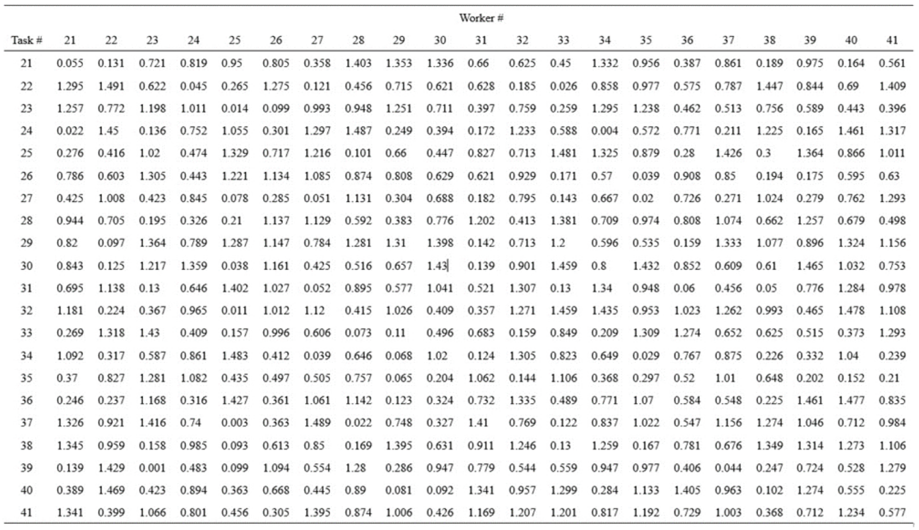
In this section, we summarize and compare our research analysis. As mentioned earlier, there are few studies investigating into the issue of "labor skill"— a resource assignment problem with task precedence. This encourages us to adopt the concept of the labor skill in our model and further apply meta-heuristic algorithms to solve for the optimal solutions. With the data from Chan *et al.* [2], we executed the single-tasking and multitasking experiments against the CG algorithm, TS algorithm and SA approach. In the single-task scenario, the CG method turns out to be the best approach, with less required number of work stations and lowest cycle time. On the other hand, the TS outperforms both CG algorithm and SA in terms of the lowest maximum cycle time of the entire assembly line. It is worthy of mentioning that our research

Table 2c | Worker skill (No. 21–41) vs. Task (No. 1–20) [2].



	Worker #																				
Task #	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41
1	0.631	1.037	0.017	0.357	0.653	1.156	0.359	0.773	1.221	1.416	1.47	0.278	0.696	1.173	0.475	0.441	1.194	0.967	0.907	0.564	0.788
2	0.228	0.182	0.391	0.6	0.199	0.592	0.989	1.471	1.304	0.039	1.129	1.431	0.724	0.438	0.275	0.173	1.053	0.293	1.143	0.138	0.810
3	0.182	0.627	0.295	0.155	0.77	1.212	0.392	0.715	0.276	0.121	0.215	1.447	0.644	0.69	1.44	0.22	1.497	1.115	1.017	0.979	1.361
4	0.95	0.647	1.477	1.013	1.174	1.114	0.756	0.585	1.433	0.171	0.516	0.777	1.269	1	0.56	1.057	0.698	1.498	1.082	0.486	0.601
5	0.457	0.649	1.439	0.051	1.062	1.462	1.128	0.491	0.118	0.135	1.344	0.404	1.067	0.134	1.333	0.892	1.124	1.23	0.934	0.062	0.594
6	0.588	1.177	1.216	0.991	0.333	0.546	1.223	0.178	0.968	0.351	0.331	1.007	0.044	0.381	0.018	0.422	0.365	1.019	0.967	1.129	0.641
7	0.857	1.33	0.224	1.466	0.649	1.173	1.101	0.163	0.947	0.998	1.48	1.097	0.306	0.111	0.134	0.315	0.06	0.216	0.02	1.283	0.258
8	1.279	0.621	0.781	1.079	1.454	0.301	1.464	0.037	0.786	0.818	0.557	1.072	0.653	0.16	0.939	1.147	0.943	0.885	1.24	1.433	0.989
9	1.256	1.337	0.43	1.039	1.126	1.426	1.362	1.072	0.108	1.002	0.789	0.651	1.23	1.375	0.879	0.861	1.106	0.183	0.59	1.39	0.521
10	0.253	0.852	0.905	0.826	1.213	0.547	1.132	0.502	1.264	0.966	0.054	0.288	0.364	0.571	0.945	1.049	0.08	1.479	0.292	0.033	0.554
11	1.115	0.015	0.763	0.502	0.623	1.39	0.245	0.969	0.23	0.524	0.594	1.19	1.427	1.465	0.99	0.268	0.693	1.48	0.994	0.61	0.453
12	1.145	132	0.468	1.014	1.01	0.416	0.666	0.519	1.04	0.552	0.194	0.185	0.359	0.767	0.115	0.51	1.158	1.471	0.999	1.034	1.284
13	1.038	0.873	0.489	0.836	1.397	0.557	0.242	0.349	0.25	1.397	0.227	1.451	0.657	0.279	1.208	1.181	0.706	0.215	1.455	0.382	0.365
14	0.244	0.593	1.17	0.551	0.813	1.403	0.266	0.217	1.203	0.664	1.011	0.455	0.156	1.355	0.628	0.787	0.104	0.483	0.158	1.291	0.481
15	0.535	1.394	0.125	0.488	0.701	0.813	1.409	0.209	0.882	0.664	1.29	0.825	0.153	0.229	0.532	0.7	0.989	0.421	1.456	0.159	1.457
16	0.443	1.371	1.027	0.594	0.697	0.404	0.099	1.064	0.134	1.146	0.587	1.168	0.125	1.091	1.153	0.624	0.226	1.298	0.107	0.381	0.902
17	1.386	0.866	1.336	0.145	0.816	0.634	0.173	1.342	0.611	1.096	0.015	1.478	0.772	0.75	0.965	0.749	0.586	0.505	1.25	0.134	0.132
18	0.623	0.975	0.069	0.436	1.241	0.413	1.338	0.835	0.085	0.519	1.425	0.081	0.035	0.624	0.007	0.787	0.151	0.909	0.192	0.669	1.231
19	0.328	0.583	0.586	1.19	0.149	1.084	1.221	0.084	1.346	0.87	0.488	0.517	0.83	0.415	1.16	0.812	0.629	0.979	0.284	1.286	0.373
20	0.363	0.186	1.365	0.804	0.078	1.319	1.338	0.493	1.128	1.397	1.469	0.981	0.696	0.555	1.164	0.861	0.253	0.418	0.85	0.005	1.258

Table 2d | Worker skill (No. 21–41) vs. Task (No. 21–41) [2].



	Worker #																				
Task #	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41
21	0.055	0.131	0.721	0.819	0.95	0.805	0.358	1.403	1.353	1.336	0.66	0.625	0.45	1.332	0.956	0.387	0.861	0.189	0.975	0.164	0.561
22	1.295	1.491	0.622	0.045	0.265	1.275	0.121	0.456	0.715	0.621	0.628	0.185	0.026	0.858	0.977	0.575	0.787	1.447	0.844	0.69	1.409
23	1.257	0.772	1.198	1.011	0.014	0.099	0.993	0.948	1.251	0.711	0.397	0.759	0.259	1.295	1.238	0.462	0.513	0.756	0.589	0.443	0.396
24	0.022	1.45	0.136	0.752	1.055	0.301	1.297	1.487	0.249	0.394	0.172	1.233	0.588	0.004	0.572	0.771	0.211	1.225	0.165	1.461	1.317
25	0.276	0.416	1.02	0.474	1.329	0.717	1.216	0.101	0.66	0.447	0.827	0.713	1.481	1.325	0.879	0.28	1.426	0.3	1.364	0.866	1.011
26	0.786	0.603	1.305	0.443	1.221	1.134	1.085	0.874	0.808	0.629	0.621	0.929	0.171	0.57	0.039	0.908	0.85	0.194	0.175	0.595	0.63
27	0.425	1.008	0.423	0.845	0.078	0.285	0.051	1.131	0.304	0.688	0.182	0.795	0.143	0.667	0.02	0.726	0.271	1.024	0.279	0.762	1.293
28	0.944	0.705	0.195	0.326	0.21	1.137	1.129	0.592	0.383	0.776	1.202	0.413	1.381	0.709	0.974	0.808	1.074	0.662	1.257	0.679	0.498
29	0.82	0.097	1.364	0.789	1.287	1.147	0.784	1.281	1.31	1.398	0.142	0.713	1.2	0.596	0.535	0.159	1.333	1.077	0.896	1.324	1.156
30	0.843	0.125	1.217	1.359	0.038	1.161	0.425	0.516	0.657	1.43	0.139	0.901	1.459	0.8	1.432	0.852	0.609	0.61	1.465	1.032	0.753
31	0.695	1.138	0.13	0.646	1.402	1.027	0.052	0.895	0.577	1.041	0.521	1.307	0.13	1.34	0.948	0.06	0.456	0.05	0.776	1.284	0.978
32	1.181	0.224	0.367	0.965	0.011	1.012	1.12	0.415	1.026	0.409	0.357	1.271	1.459	1.435	0.953	1.023	1.262	0.993	0.465	1.478	1.108
33	0.269	1.318	1.43	0.409	0.157	0.996	0.606	0.073	0.11	0.496	0.683	0.159	0.849	0.209	1.309	1.274	0.652	0.625	0.515	0.373	1.293
34	1.092	0.317	0.587	0.861	1.483	0.412	0.039	0.646	0.068	1.02	0.124	1.305	0.823	0.649	0.029	0.767	0.875	0.226	0.332	1.04	0.239
35	0.37	0.827	1.281	1.082	0.435	0.497	0.505	0.757	0.065	0.204	1.062	0.144	1.106	0.368	0.297	0.52	1.01	0.648	0.202	0.152	0.21
36	0.246	0.237	1.168	0.316	1.427	0.361	1.061	1.142	0.123	0.324	0.732	1.335	0.489	0.771	1.07	0.584	0.548	0.225	1.461	1.477	0.835
37	1.326	0.921	1.416	0.74	0.003	0.363	1.489	0.022	0.748	0.327	1.41	0.769	0.122	0.837	1.022	0.547	1.156	1.274	1.046	0.712	0.984
38	1.345	0.959	0.158	0.985	0.093	0.613	0.85	0.169	1.395	0.631	0.911	1.246	0.13	1.259	0.167	0.781	0.676	1.349	1.314	1.273	1.106
39	0.139	1.429	0.001	0.483	0.099	1.094	0.554	1.28	0.286	0.947	0.779	0.544	0.559	0.947	0.977	0.406	0.044	0.247	0.724	0.528	1.279
40	0.389	1.469	0.423	0.894	0.363	0.668	0.445	0.89	0.081	0.092	1.341	0.957	1.299	0.284	1.133	1.405	0.963	0.102	1.274	0.555	0.225
41	1.341	0.399	1.066	0.801	0.456	0.305	1.395	0.874	1.006	0.426	1.169	1.207	1.201	0.817	1.192	0.729	1.003	0.368	0.712	1.234	0.577

utilizes the existing meta-heuristic algorithms to derive the near-optimal solutions within a limited time. This would assist garment industry in implementing the scheduling model with consideration of the labor skill in an efficient manner. Furthermore, because of the Internet of Things (IoT) trend, one can expect that the data at the shop floor can be automatically collected. We can apply the labor performance data captured by tasks and then employ the artificial intelligence (AI) model to predict the behaviors of labors. This would be more accurate to formulate the labor skills in our model.

5. CONCLUSIONS AND RECOMMENDATIONS

In this research, we apply the TS and SA algorithms to solve the ALB problem considering the labor-intensive nature of the garment industry. The objective of the proposed model is to minimize the cycle time in the assembly line performed through task placement. Nevertheless, in the garment industry, since tasks or machines usually are placed at fixed positions at the early stage of facility layout planning, the cycle time can be minimized via the staff assignment. Based on the resulting outcomes for the single-task scenario, the CG method outperforms TS and SA in two aspects: work stations numbers and cycle time. Furthermore, this paper also adopts the multitasking concept of TSS derived from TPS to solve the ALB problem in the garment industry by assigning 30% of workers to work stations in the multitasking mode. The TS comes out to be the winner in terms of the lowest cycle time. We observe that with a worker with randomized skill levels for different tasks as given by [6], it is less likely that a worker, exceptional for a single task, may perform equally well against multiple tasks. Therefore, in regards to the managerial implication, a team of workers with specific talents (high skill proficiency) for particular tasks performs better than multitasking workers with disparate skill levels do.

Table 3a Statistics of constructive greedy algorithm for 13 work stations.

Total adjusted SAM		**1,282.26 minutes**
Work station processing time statistics		**Mean: 98.635; Min: 18.919; Max: 235.258; Std. Dev.: 69.964**
Work station assignment	WS1	{T1-W6, T2-W16, T3-W37, T4-W38, T5-W10, T6-W9}
	WS2	{T7-W31}
	WS3	{T8-W27, T9-W20, T10-W14, T11-W34, T12-W22}
	WS4	{T13-W39, T14-W12, T15-W41, T16-W2, T17-W32}
	WS5	{T18-W8, T19-W7}
	WS6	{T20-W30}
	WS7	{T21-W28, T22-W13, T23-W17}
	WS8	{T24-W40}
	WS9	{T25-W33, T26-W23, T27-W11, T28-W5, T29-W4}
	WS10	{T30-W15, T31-W25, T32-W3, T33-W1}
	WS11	{T34-W21, T35-W24, T36-W18, T37-W19, T38-W29, T39-W26}
	WS12	{T40-W36}
	WS13	{T41-W35}

Table 3b Statistics of constructive greedy algorithm for nonpredetermined work stations.

Total adjusted SAM		**1,282.26 minutes**
Work station processing time statistics		**Mean: 116.5689; Min: 93.649; Max: 132.904; Std. Dev: 14.73408**
Work station assignment	WS1	{T1-W6, T2-W16, T3-W37, T4-W38, T5-W10}
	WS2	{T6-W9, T7-W31, T8-W27, T9-W20}
	WS3	{T10-W14, T11-W34, T12-W22, T13-W39, T14-W12}
	WS4	{T15-W41, T16-W2, T17-W32, T18-W8}
	WS5	{T19-W7, T20-W30, T21-W28, T22-W13}
	WS6	{T23-W17, T24-W40, T25-W33}
	WS7	{T26-W23, T27-W11, T28-W5}
	WS8	{T29-W4, T30-W15, T31-W25}
	WS9	{T32-W3, T33-W1, T34-W21, T35-W24}
	WS10	{T36-W18, T37-W19, T38-W29}
	WS11	{T39-W26, T40-W36, T41-W35}

Table 4 Parameter values for Tabu search.

Number of Iterations	**Tabu List**	**Tabu Move**
1000	1681	5

We outline the contributions of this research as following.

(1) Three algorithms—CG, TS and SA algorithms—are proposed and discussed.

Table 5 Parameters for simulated annealing.

Maximum Iteration	λ	**Initial Temperature**
1000	0.70	100

Table 6 Statistics of Tabu search for single-task workers for nonpredetermined work stations.

Total adjusted SAM		**2,325.415 minutes**
Work station processing time statistics		**Mean: 96.892; Min: 47.235; Max: 131.998; std. Dev: 23.085**
Work station assignment	WS1	{T1-W25, T2-W6, T3-W39}
	WS2	{T4-W21, T5-W8}
	WS3	{T6-W38, T7-W22}
	WS4	{T8-W17, T9-W28}
	WS5	{T10-W27}
	WS6	{T11-W2}
	WS7	{T12-W14}
	WS8	{T13-W40, T14-W16, T15-W7}
	WS9	{T16-W5, T17-W4, T18-W3}
	WS10	{T19-W24, T20-W35}
	WS11	{T21-W10, T22-W37}
	WS12	{T23-W41}
	WS13	{T24-W33}
	WS14	{T25-W26, T26-W23}
	WS15	{T27-W34}
	WS16	{T28-W29}
	WS17	{T29-W15}
	WS18	{T30-W13}
	WS19	{T31-W19, T32-W20}
	WS20	{T33-W30, T34-W1}
	WS21	{T35-W12}
	WS22	{T36-W18, T37-W9}
	WS23	{T38-W36, T39-W11}
	WS24	{T40-W32, T41-W31}

(2) The original single-task mode of production line is revised; 30% labor force are assigned to work stations in the multitasking scenario.

In our research, we adjust the cycle time based on employee's proficiency assignment to calculate the cycle time. To reflect the reality of the world, we consider employee's skill to update the standard task working time. Although, TSS promote the concept of multitasking, it should be noted that the existing limitations of production processes prevent the entire update of all or most single production into multitasking mode. For example, the turnover rate of skilled or cross-trained workers can be quite high. This may discourage employers from cross-training the entire workforce. Additionally, the ramp-up time for multitasking may dissuade the management to put new workers immediately in a cell to handle multiple tasks. Therefore, in our research we only consider 30% of workforce as multitasking workers in terms of predetermined number of work stations. For future work, we suggest a number of issues for future researchers.

Table 7 Statistics of simulated annealing for single-task workers.

Total adjusted SAM		**2,129.846 minutes**
Work station processing time statistics		**Min: 47.714; Max: 183.007; Std Dev: 31.796**
Work station assignment	WS1	{T1-W18, T2-W37, T3-W38}
	WS2	{T4-W40}
	WS3	{T5-W12, T6-W5, T7-W2}
	WS4	{T8-W41, T9-W24}
	WS5	{T10-W30, T11-W20, T12-W4}
	WS6	{T13-W19, T14-W25}
	WS7	{T15-W33}
	WS8	{T16-W16, T17-W28}
	WS9	{T18-W9}
	WS10	{T19-W6}
	WS11	{T20-W27, T21-W1, T22-W15}
	WS12	{T23-W36, T24-W32}
	WS13	{T25-W35}
	WS14	{T26-W3}
	WS15	{T27-W22, T28-W21}
	WS16	{T29-W34}
	WS17	{T30-W7, T31-W26}
	WS18	{T32-W13, T33-W23, T34-W14, T35-W10}
	WS19	{T36-W11, T37-W17}
	WS20	{T38-W31, T39-W8, T40-W39}
	WS21	{T41-W29}

Table 8 Comparison of solution results in the single-task scenario.

	Fixed Size (CG)*	**CG**	**TS**	**SA**
Best Soln. (maximum cycle time)	235.258	132.904	131.998	183.007
Number of work stations	13	11	24	21

*13 pre-determined work stations.

Table 9 Statistics of constructive greedy algorithm for multitasking.

Work station processing time statistics		**Mean: 965.792; Min: 18.919; Max: 6124.305; Std Dev: 1861.080**
Work station assignment	WS1	{T[1-6]-W9}
	WS2	{T7-W31}
	WS3	{T[8-12]-W20}
	WS4	{T[13-17]-W39}
	WS5	{T[18-19]-W7}
	WS6	{T20-W30}
	WS7	{T[21-23]-W14}
	WS8	{T24-W28}
	WS9	{T[25-29]-W2}
	WS10	{T[30-33]-W25}
	WS11	{T[34-39]-W8}
	WS12	{T40-W22}
	WS13	{T41-W27}

Table 10 Statistics of tabu search for multitasking.

Work station processing time statistics		**Mean: 605.017; Min: 91.503; Max: 2769.603; Std Dev: 755.573**
Work station assignment	WS1	{T[1-6]-W12}
	WS2	{T7-W33}
	WS3	{T[8-12]-W13}
	WS4	{T[13-17]-W11}
	WS5	{T[18-19]-W28}
	WS6	{T20-W15}
	WS7	{T[21-23]-W6}
	WS8	{T24-W7}
	WS9	{T[25-29]-W29}
	WS10	{T[30-33]-W10}
	WS11	{T[34-39]-W2}
	WS12	{T40-W1}
	WS13	{T41-W26}

Table 11 Statistics of simulated annealing for multitasking.

Work station processing time statistics		**Mean: 762.782; Min: 38.654; Max: 4129.875; Std Dev: 1211.692**
Work station assignment	WS1	{T[1-6]-W3}
	WS2	{T7-W6}
	WS3	{T[8-12]-W25}
	WS4	{T[13-17]-W33}
	WS5	{T[18-19]-W8}
	WS6	{T20-W30}
	WS7	{T[21-23]-W2}
	WS8	{T24-W16}
	WS9	{T[25-29]-W35}
	WS10	{T[30-33]-W21}
	WS11	{T[34-39]-W10}
	WS12	{T40-W11}
	WS13	{T41-W4}

Table 12 Best solutions of three algorithms for multitasking.

	CG	**TS**	**SA**
Best Soln. (maximum cycle time)	6124.305	2769.603	4129.875

(1) Our research focuses on the scenario of tasks placed at fixed positions at the construction stage—we can only change labor assignment. For further investigation, we may take into consideration of both employee and task swapping.

(2) In the research, we implement CG, TS and SA algorithms in this research. We can employ other approaches in the future research.

(3) The study is concerned with a single objective—the cycle time; multiple objectives may be investigated in the future.

(4) Due to the trend of IoT and applications of AI, researchers may further incorporate the machine learning algorithms into our model to better predict the performance of labor.

AUTHORS' CONTRIBUTIONS

Gary Yu-Hsin Chen: Conceptualization, Software, Investigation, Writing - Original Draft, Funding acquisition. **Ping-Shun Chen:** Methodology, Supervision, Software. **Jr-Fong Dang:** Formal analysis, Writing - Review & Editing, Project administration. **Sung-Lien Kang:** Investigation, Resources. **Li-Jen Cheng:** Software, Writing - Original Draft.

ACKNOWLEDGMENTS

This work was supported by Ministry of Science and Technology of Taiwan under Grant MOST 108-2221-E-992-019-.

REFERENCES

[1] E.P. Degarmo, J.T. Black, R.A. Kohser, B.E. Klamecki, Materials and Process in Manufacturing, John Wiley and Sons, Hoboken, NJ, USA, 2003.
[2] K.C.C. Chan, P.C.L. Hui, K.W. Yeung, F.S.F. Ng, Handling the assembly line balancing problem in the clothing industry using a genetic algorithm, Int. J. Cloth. Sci. Technol. 10 (1997), 21–37.
[3] Z.X. Guo, W.K. Wong, S.Y.S. Leung, J.T. Fan, S.F. Chan, Mathematical model and genetic optimization for the job shop scheduling problem in a mixed- and multi-product assembly environment: a case study based on the apparel industry, Comput. Ind. Eng. 50 (2006), 202–219.
[4] S. Ağrali, Z.C. Taşkin, A.T. Ünal, Employee scheduling in service industries with flexible employee availability and demand, Omega. 66 (2017), 159–169.
[5] B. Denkena, M.A. Dittrich, F. Winter, C. Wagener, Simulation-based planning and evaluation of personnel scheduling in knowledge-intensive production systems, Prod. Eng. Res. Dev. 10 (2016), 489–496.
[6] X. Chen, B.W. Thomas, M. Hewitt, Multi-period technician scheduling with experience-based service, Comput. Oper. Res. 82 (2017), 1–14.
[7] Z. Wang, H. Hu, J. Gong, Competence based worker assignment and impacts on production scheduling in precast construction, in IEEE Technology and Engineering Management Conference (TEMSCON), Evanston, IL, USA, 2018.
[8] R. Chen, C. Liang, D. Gu, J.Y.-T. Leung, A multi-objective model for multi-project scheduling and multi-skilled staff assignment for IT product development considering competency evolution, Int. J. Prod. Res. 55 (2017), 6207–6234.
[9] M.R. Garey, D.S. Johnson, Computers and Intractability: a Guide to the Theory of NP-Completeness, W.H. Freeman, New York, NY, USA 1979.
[10] A.L. Gutjahr, G.L. Nemhauser, An algorithm for the line balancing problem, Manag. Sci. 11 (1964), 308–315.
[11] G.A. Suer, Designing parallel assembly lines, Comput. Ind. Eng. 35 (1998), 467–470.
[12] A. Scholl, R. Klein, ULINO: optimally balancing U-shaped JIT assembly lines, Int. J. Prod. Res. 37 (1999), 721–736.
[13] M.-T. Lin, The single-row machine layout problem in apparel manufacturing by hierarchical order-based genetic algorithm, Int. J. Cloth. Sci. Technol. 20 (2009), 258–270.
[14] S. Suwannarongsri, D. Puangdownreong, Optimal assembly line balancing using Tabu search with partial random permutation technique, Int. J. Manag. Sci. Eng. Manag. 3 (2008), 3–18.
[15] M.H. Dinh, V.D. Nguyen, Cycle time enhancement by simulated annealing for a practical assembly line balancing problem, Informatica. 44 (2020), 127–138.
[16] H. Xu, B. Xu, J. Yan, Balancing apparel assembly lines through adaptive ant colony optimization, Text. Res. J. 89 (2019), 3677–3691.
[17] N.T.P. Quyen, J.C. Chen, C.-L. Yang, Hybrid genetic algorithm to solve resource constrained assembly line balancing problem in footwear manufacturing, Soft Comput. 21 (2017), 6279–6295.
[18] M. Gansterer, R.F. Hartl, One- and two-sided assembly line balancing problems with real-world constraints, Int. J. Prod. Res. 56 (2018), 3025–3042.
[19] Q.-V. Dang, K. Pham, Design of a footwear assembly line using simulated-based ALNS, Procedia CIRP. 40 (2016), 596–601.
[20] J. Pereira, Modelling and solving a cost-oriented resource-constrained multi-model assembly line balancing problem, Int. J. Prod. Res. 56 (2018), 3994–4016.
[21] J. Bautista, R. Alfaro-Pozo, C. Batalla-Garcia, Maximizing comfort in assembly lines with temporal, spatial and ergonomic attributes, Int. J. Comput. Int. Syst. 9 (2016), 788–799.
[22] P. Pujo, I.E. Khabous, F. Ounnar, Experimental assessment of the productivity improvement when using U-shaped production cells with variable takt time, Int. J. Lean Six Sigma. 6 (2015), 17–38.
[23] L.J. Krajewski, M. Malhotra, L.P. Malhotra, Operations Management: Processes and Supply Chains, eleventh ed., Pearson, London, England, 2016.