applied sciences

MDPI

Article

An Algorithm for Arranging Operators to Balance Assembly Lines and Reduce Operator Training Time

Ming-Liang Li

Department of Industrial Management, Asia Eastern University of Science and Technology, Banqiao Dist., New Taipei City 22061, Taiwan; fg017@mail.aeust.edu.tw; Tel.: +886-(2)-7738-0145 (ext. 5113)

Abstract: Industry 4.0 is transforming how costs, including labor costs, are managed in manufacturing and remanufacturing systems. Managers must balance assembly lines and reduce the training time of workstation operators to achieve sustainable operations. This study's originality lies in its use of an algorithm to balance an assembly line by matching operators to workstations so that the line's workstations achieve the same targeted output rates. First, the maximum output rate of the assembly line is found, and then the number of operators needed at each workstation is determined. Training time is reduced by matching operators' training and skills to workstations' skill requirements. The study obtains a robust, cluster algorithm based on the concept of group technology, then forms operator skill cells and determines operator families. Four numerical examples are presented to demonstrate the algorithm's implementation. The proposed algorithm can solve the problem of arranging operators to balance assembly lines. Managers can also solve the problem of worker absences by assigning more than one operator with the required skillset to each workstation and rearranging them as needed.

Keywords: assembly line balancing; group technology; cluster algorithm; bottleneck station; output rate

Citation: Li, M.-L. An Algorithm for Arranging Operators to Balance Assembly Lines and Reduce Operator Training Time. *Appl. Sci.* **2021**, *11*, 8544. https://doi.org/10.3390/app11188544

Academic Editors: Vladimir Modrak and Zuzana Soltysova

Received: 26 July 2021
Accepted: 12 September 2021
Published: 14 September 2021

1. Introduction

Netessine and Taylor [1] demonstrated that expensive production technology usually leads to low product prices and may simultaneously lead to high-quality products. However, the product price is usually proportional to the product cost, part of which is labor cost. Assembly line balancing (ALB) is a crucial part of production/operations management [2]. On most mass-production assembly lines, workers repetitively perform a set of tasks predefined by using assembly line balancing techniques [3], which assign a number of work elements to various workstations to maximize the assembly line's balancing efficiency [4]. A balancing of tasks across the workstations results in the optimization of a given objective function without violating preceding constraints [5]. An unbalanced assembly line can be dangerous. For example, an operator without adequate training or who is focused on personal problems can cause an accident, decrease the efficiency of the assembly line, and increase production costs. Assembly lines maximize assembly operations, and most manufacturing plants have one or more lines, making assembly line balancing one of the oldest challenges in industry [6]. Moreover, a balanced assembly line can reduce a factory's work-in-process (WIP) inventory. Kim et al. [7] suggested a mathematically mixed integer programming model that accounts for the WIP balance and setup time in a semiconductor fabrication line. Maximum productivity and minimum load smoothness are benefits of solving the assembly line balancing problem (ALBP) [8,9].

Many types of production lines exist. Kuroda [10] presented a robust design for a cellular-line production system with unreliable facilities. A cellular-line production system comprises multiple flow shops, which are individual sets composed of functionally different facilities. Kuroda grouped facilities that perform similar operations.

This study initially considers a traditional assembly line that has four workstations with outputs λ_A, λ_B, λ_C, and λ_D (Figure 1). Thus, the production rate, or output rate, of this assembly line is equal to $\lambda_{line} = min.\{\lambda_A, \lambda_B, \lambda_C, \lambda_D\}$.

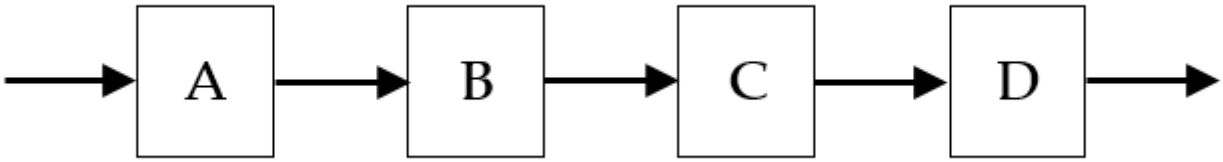

Figure 1. Traditional assembly line.

The cycle time (CT) refers to the longest period required for workstations to complete operations. Therefore, all station operation times are less than or equal to the CT. The aim of balancing the assembly line is to ensure that the output rate and operating time of each workstation are identical. Previous studies have examined assembly line balancing. According to Bartholdi and Eisenstein [11], the assembly line balances itself. They found that if workers are sequenced from slowest to fastest, a stable partition of work spontaneously emerges independent of the stations at which the workers begin. Moreover, the production rate converges to a value. Huang [12] determined that the CT and labor efficiency are negatively exponential to the number of assigned workers. Sawik [13] presented new mixed integer programming formulations for scheduling a flexible flow line with blocking. In this flexible flow line, each stage has one or more identical parallel machines. This study uses parallel operators instead of parallel machines. Rearranging operators may increase the production rate. Muth [14] examined whether a rearrangement in the order of servers affects the production rate. Muth speculated that the production rate remains invariant if the production line is reversed.

If the number of workers at a workstation is more than the number of machines, the machines are expected to limit the output rate and cause a bottleneck. However, Zavadlav et al. [15] showed that the workers, rather than the machines, limit the rate of output.

The manager of a factory always works under "cost" pressure; the company wants the manager to make continual improvements without increasing the cost. Managers typically try to increase the output by adding machines and workers, but managers must first consider whether assembly lines are balanced. When assembly lines are not balanced, productivity is low and workers complain. Workers might not only complain that they have to do more work at their station for the same pay, but also that their skills are not suited to their workstation. This work environment might lead workers to miss shifts on purpose or to resign.

Many ALBP methods have been examined in the study of manufacturing systems, including the heuristic algorithm [16–24], artificial bee colony algorithm [25–27], genetic algorithm [28–30], branch-and-bound algorithm [31–34], imperialist competitive algorithm [35], Pareto Greedy algorithm [36], memetic ant algorithm [37], whale optimization algorithm [38], recursive algorithm [39], fuzzy linear programming [40], swarm algorithm [41,42], simulated annealing algorithm [43], and data envelopment analysis [44]. The genetic algorithm [45,46] and artificial bee colony algorithm [47] have been used to solve the ALBP in remanufacturing systems. However, studies have seldom discussed how to arrange operators and group similarly skilled operators at the same workstation. This paper considers workers, not machines, in determining whether a workstation has the right equipment and machinery for operators to use. To achieve this goal, the study works out a method for rapidly balancing assembly lines by arranging operators among the workstations and uses a robust, cluster algorithm based on the concept of group technology (GT) to form operator skill cells and determine operator families.

2. Problem Statement

Traditionally, when arranging a limited number of operators on an assembly line, a manager begins with one operator per workstation, before adding an operator to the workstation with the lowest output. Step by step, all of the operators are arranged and the output rate can be found.

When a workstation has a lower output rate, the manager typically adds another operator. However, the lower output might be due to the "human" factor; an operator who is trained on many machines might be assigned to a workstation outside of their skill set. For example, if an operator knows how to work an auto-optical inspection machine but is assigned to a workstation with an in-circuit test machine, the station's output rate will drop because the operator does not have the required skills. Output can also fall when a worker is absent or resigns unexpectedly and the manager fills the vacancy with a random worker who has not been trained to operate that workstation.

Thus, the problem statement of this study is:

Ideal: The maximum output rate is found, all of the workstations have the same output rate, and the number of operators at each workstation can be found, with all operators assigned to a workstation that matches their skill set.

Reality: The arrangement algorithm is step by step, and the maximum output rate cannot be determined until all of the operators have been arranged. Operators can be assigned to unsuitable workstations.

Consequences: The maximum output rate is an unknown that will affect a boss's decisions, especially before setting up a new plant. An operator who works at an unsuitable workstation might complain, be absent, or quit unexpectedly, causing a lower output rate.

Proposal: An arrangement algorithm with a constant number of operators must consider the maximum output rate and balance the assembly line without any increase in cost. A sorting algorithm must consider the skills required at each workstation and the skill set of each operator.

3. Methodology

A manager adds a worker to a workstation that has a lower output than at other workstations. When the output of another workstation is lower, the manager adds another worker to that workstation. The use of this step-by-step method balances the assembly lines, especially when setting up a new plant.

3.1. Example of Step-by-Step Method

Suppose an assembly line has six workstations, as shown in Figure 2.

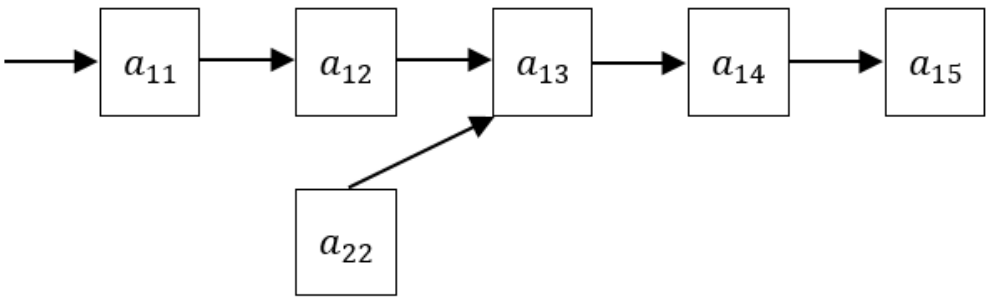

Figure 2. Assembly line with six workstations.

In addition, suppose that the terms t_{11}, t_{12}, t_{22}, t_{13}, t_{14}, and t_{15} are equal to 12, 15, 10, 16, 20, and 8 min, respectively. If one workday comprises 8 h, then the output per operator per day at each workstation, γ_{11}, γ_{12}, γ_{22}, γ_{13}, γ_{14}, and γ_{15}, are 40, 32, 48, 30, 24, and 60, respectively. Suppose that 500 workers are assigned to this assembly line. The workers can be assigned step by step, as follows (Table 1).

Table 1. Traditional ALB assignment.

	a_{11}	a_{12}	a_{22}	a_{13}	a_{14}	a_{15}	
p_{ij}	1	1	1	1	1	1	$\sum p_{ij} = 6$
Υ_{ij}	40	32	48	30	24	60	$\Upsilon_{line} = 24$

The term λ_{line} is limited in workstation a_{14}, which is a bottleneck workstation. Thus, this workstation should have one more operator to increase the output rate. The seventh worker is added to workstation a_{14}, and the following is obtained (Table 2).

Table 2. Traditional ALB assignment.

	a_{11}	a_{12}	a_{22}	a_{13}	a_{14}	a_{15}	
p_{ij}	1	1	1	1	2	1	$\sum p_{ij} = 7$
Υ_{ij}	40	32	48	30	48	60	$\Upsilon_{line} = 30$

The assignment of workers in a step-by-step manner uses considerable time to obtain the following final result (Table 3).

Table 3. Traditional ALB assignment.

	a_{11}	a_{12}	a_{22}	a_{13}	a_{14}	a_{15}	
p_{ij}	74	92	62	99	123	50	$\sum p_{ij} = 500$
Υ_{ij}	2960	2944	2976	2970	2952	3000	$\Upsilon_{line} = 2944$

Considerable time was spent and many steps performed to determine the optimal arrangement of operators at the assembly line's workstations. Although some studies have discussed and tried to solve the ALBP, the question remains: Is there another method for arranging the workers, instead of using the step-by-step method? This paper presents a novel approach for solving this problem.

Almost all workers require training before operating a workstation. Ideally, the operators on an assembly line would have all of the skills required for each workstation, but that would be very costly. Thus, the operators' skills must be determined so that they can be placed at a workstation that requires their skills. For example, consider an assembly line that has welding and drilling workstations. If operator κ possesses welding skills, assigning operator κ to the welding workstation rather than to the drilling workstation will reduce the training time for the welding workstation.

Suppose that there is a balanced assembly line with seven workstations (Figure 3), and workstations 1–7 require one, three, four, two, two, two, and one operator, respectively.

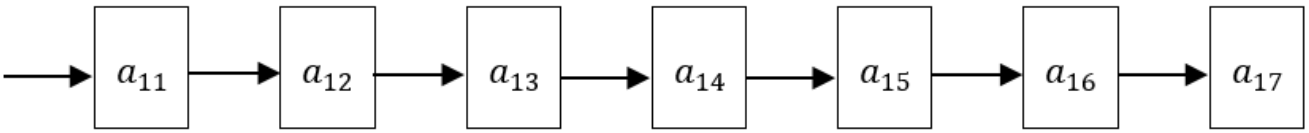

Figure 3. Assembly line with seven workstations.

Each workstation requires a skill, which is denoted as $s_1, s_2, ..., s_7$. A matrix $SO = \{so_{ij}\}_{uv}$ describes the relationship between skills and operators. For example, $so_{ij} = 1$ indicates

that operator j possesses skill i. For the assembly line in Figure 3 to function, the minimum matrix SO must be as follows:

$$SO = \begin{array}{c} \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \end{array} \begin{array}{c} \begin{array}{cccccccccccccc} o_1 & o_2 & o_3 & o_4 & o_5 & o_6 & o_7 & o_8 & o_9 & o_{10}^{1} & o_{11}^{1} & o_{12}^{1} & o_{13}^{1} & o_{14}^{1} & o_{15}^{1} \end{array} \\ \begin{bmatrix} 1 & & & & & & & & & & & & & & \\ & 1 & 1 & 1 & & & & & & & & & & & \\ & & & & 1 & 1 & 1 & 1 & & & & & & & \\ & & & & & & & & 1 & 1 & & & & & \\ & & & & & & & & & & 1 & 1 & & & \\ & & & & & & & & & & & & 1 & 1 & \\ & & & & & & & & & & & & & & 1 \end{bmatrix} \end{array}$$

Suppose it is known that:

$$SO = \begin{array}{c} \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \end{array} \begin{array}{c} \begin{array}{cccccccccccccc} o_1 & o_2 & o_3 & o_4 & o_5 & o_6 & o_7 & o_8 & o_9 & o_{10}^{1} & o_{11}^{1} & o_{12}^{1} & o_{13}^{1} & o_{14}^{1} & o_{15}^{1} \end{array} \\ \begin{bmatrix} & & & & 1 & & 1 & 1 & & & 1 & & & 1 & \\ 1 & 1 & & & & & & & & & & 1 & & 1 & \\ & & 1 & 1 & & & 1 & & & & 1 & & 1 & & 1 \\ 1 & & & & & 1 & 1 & & & 1 & & 1 & & & \\ & & & 1 & & & 1 & & 1 & & & & 1 & & 1 \\ & & & & 1 & 1 & & 1 & & & & & 1 & & \\ & 1 & & & & & & & & 1 & & 1 & & 1 & \end{bmatrix} \end{array}$$

On the basis of the first-come, first-served rule, operator 1 is assigned to s_1; operators 2, 3, and 4 are assigned to s_2; operators 5, 6, 7, and 8 are assigned to s_3; operators 9 and 10 are assigned to s_4; operators 11 and 12 are assigned to s_5; operators 13 and 14 are assigned to s_6; and operator 15 is assigned to s_7. Thus, at least 11 operators must take the training course, namely operators 1, 3, 4, 5, 6, 8, 9, 11, 12, 14, and 15, as illustrated in the following matrix where an "X" denotes that the operator needs the training course:

$$SO = \begin{array}{c} \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \end{array} \begin{array}{c} \begin{array}{cccccccccccccc} o_1 & o_2 & o_3 & o_4 & o_5 & o_6 & o_7 & o_8 & o_9 & o_{10}^{1} & o_{11}^{1} & o_{12}^{1} & o_{13}^{1} & o_{14}^{1} & o_{15}^{1} \end{array} \\ \begin{bmatrix} \boxed{X} & & & & 1 & & 1 & 1 & & & 1 & & & 1 & \\ 1 & \boxed{1} & \boxed{X} & \boxed{X} & & & & & & & & 1 & & 1 & \\ & & 1 & 1 & \boxed{X} & \boxed{X} & 1 & \boxed{X} & & & 1 & & 1 & & 1 \\ 1 & & & & & 1 & 1 & & \boxed{X} & 1 & & 1 & & & \\ & & & 1 & & & 1 & & 1 & & \boxed{X} & \boxed{X} & 1 & & 1 \\ & & & & 1 & 1 & & 1 & & & & & \boxed{1} & \boxed{X} & \\ & 1 & & & & & & & & 1 & & 1 & & 1 & \boxed{X} \end{bmatrix} \end{array}$$

A cluster algorithm is introduced in this paper to reduce the training time and increase the speed at which an assembly line functions. Workers' absences are also a problem for managers [48,49], but the proposed cluster algorithm can suggest which worker to rotate to a workstation with an absent operator.

Instead of using the traditional step-by-step method, this study aims to use the proposed algorithm to assign operators to each workstation. Before assigning the operators, some basic data must first be known about each workstation, such as the required skills, the standardization time, the work time per day, and the total number of operators. Then, the approximate number of operators needed at each workstation can be found by using Equations (1)–(7) in Section 3.1. Finally, the number of operators at each workstation can be adjusted by using the traditional step-by-step method. The training time can be reduced by assigning an operator with suitable skills to a suitable workstation, or by sorting the skill–operator matrix by using the cluster algorithm proposed in Section 3.2. The concept is shown in Figure 4.

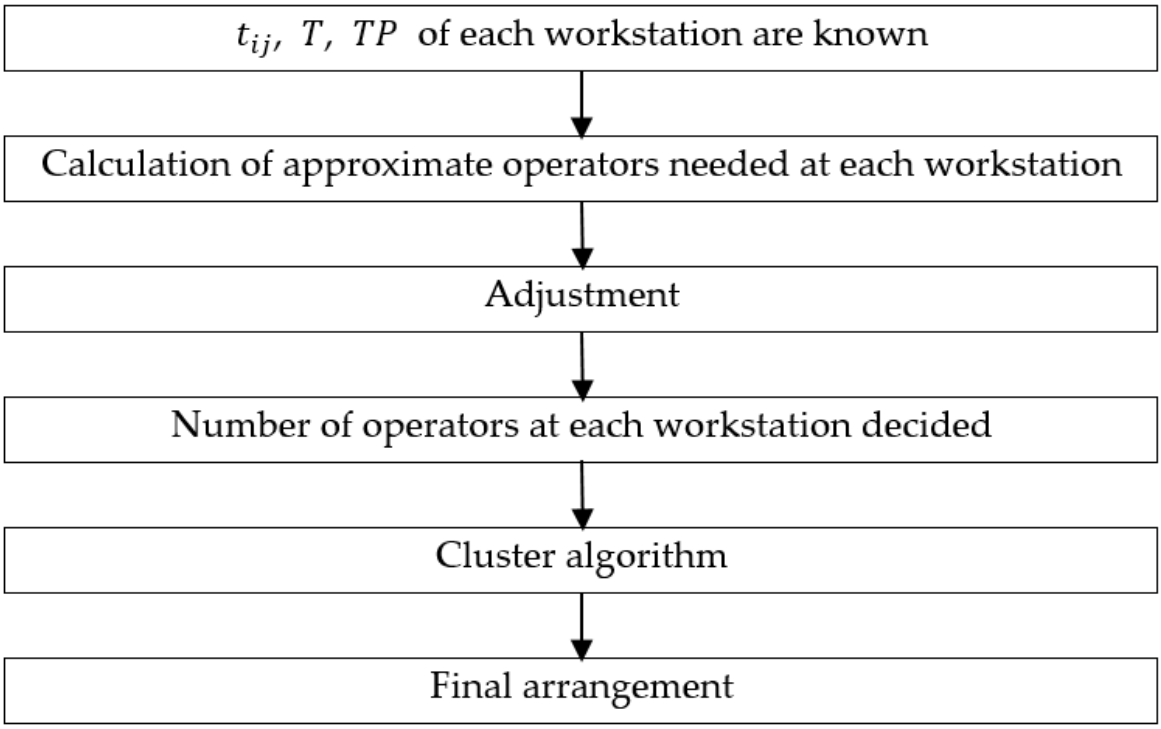

Figure 4. The novel algorithm proposed in this study.

3.2. Number of Operators at Each Workstation Decided

Suppose an assembly line has $m \times n$ workstations (Figure 5).

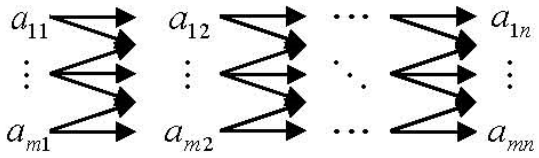

Figure 5. An assembly line with $m \times n$ workstations.

The matrix $A = \{a_{ij}\}_{mn}$ can be used instead of the previous assembly line.

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

The output rate of the assembly line is λ_{line} where

$$\lambda_{line} = min.\{\lambda_{ij}\},\ 1 \le i \le m,\ 1 \le j \le n$$

The ideal assembly line means that

$$\lambda_{line} = \lambda_{11} = \lambda_{21} = \cdots = \lambda_{mn} \quad (1)$$

because

$$\lambda_{ij} = p_{ij}\gamma_{ij} \quad (2)$$

The term p_{ij} can be easily determined, as follows:

$$p_{ij} = \frac{\lambda_{ij}}{\gamma_{ij}} \tag{3}$$

Moreover, the total number of operators is $P.$, where

$$P = \sum_{i=1}^{m} \sum_{j=1}^{n} p_{ij} \tag{4}$$

Thus, the following equations are obtained:

$$P = \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{\lambda_{ij}}{\gamma_{ij}} = \lambda_{line} \cdot \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{1}{\gamma_{ij}} = \lambda_{line} \cdot \frac{1}{T} \sum_{i=1}^{m} \sum_{j=1}^{n} t_{ij} \tag{5}$$

and,

$$\lambda_{line} = \frac{P}{\sum_{i=1}^{m} \sum_{j=1}^{n} \frac{1}{\gamma_{ij}}} = \frac{P \cdot T}{\sum_{i=1}^{m} \sum_{j=1}^{n} t_{ij}} \tag{6}$$

Replace λ_{ij} from Equation (3) with λ_{line} from Equation (6) to obtain the following equation:

$$p_{ij} = \frac{\lambda_{ij}}{\gamma_{ij}} = \frac{P}{\gamma_{ij} \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{1}{\gamma_{ij}}} = \frac{P \cdot T}{\gamma_{ij} \sum_{i=1}^{m} \sum_{j=1}^{n} t_{ij}} \tag{7}$$

As term p_{ij} is not always an integer, it may need to be chopped off.

Almost all of the operators are now assigned. The traditional step-by-step method is adjusted (as demonstrated in the previous section) and used to assign the remaining workers.

3.3. Cluster Algorithm

The proposed cluster algorithm is based on the concept of GT, which has been applied to cellular manufacturing systems [50,51]. The concept involves grouping machines into machine cells and parts into part families [52–59]. Several algorithms, such as heuristic algorithms [60,61], genetic algorithms [62,63], and neural network algorithms [64], have been derived for solving the GT problem. The proposed algorithm begins with the closeness matrix and a comparison with other algorithms in example 3.

First, the relationship between u skills and v operators, which is represented by the matrix $SO = \left\{so_{ij}\right\}_{uv}$ is determined as follows:

$$so_{ij} = \begin{cases} 1, & if\ operator\ j\ has\ i\ skill \\ 0, & otherwise \end{cases}$$

The closeness matrix of skills is defined as $B_u = \{b_{rs}\}_{u \times u}$, where

$$b_{rs} = \begin{cases} \sum_{k=1}^{v} so_{rk} so_{sk}, & if\ r \neq s \\ 0 \qquad\qquad\quad , & otherwise. \end{cases} \tag{8}$$

For example, if there are three workers that know how to operate the welding and milling machines, the closeness between the welding and milling skills is three.

3.3.1. Skills Sorting

Whether two skills are grouped depends on their closeness. Closeness denotes how many operators have both of the skills. The more operators have the two skills, the closer the skills are. First, the skill with the highest closeness value is found and a new group is formed. Second, an unsorted skill follows a sorted skill, based on the highest value in its

row in the closeness matrix. If there is no sorted skill to follow, the skill is placed in a new group. Finally, all skills are grouped.

Thus, the skills can be sorted in three steps:

Step 1. Let $f \in G_1$, where $f = \{x | b_{xk} = max.\{b_{rs}\}, 1 \leq k \leq v\}$. A tie is broken by selecting the smaller one.

Step 2. Ignore all the grouped rows and form a new closeness matrix $B'u = \{b_{ij}\}$. Let

$$\begin{cases} f \in G_k, & if = \{b_{ij} = max\{b_{ij}\} \wedge y \in G_k \\ h \in G_{new\ group}, & otherwise \end{cases}$$

Step 3. Repeat Step 2 until u skills are grouped.

3.3.2. Operator Sorting

Operator sorting aims to place operators in the group that will use the highest number of their skills.

Thus, let $v \in G_y$, $y = \left\{ x \middle| \sum_{u \in G_x} so_{uv} = max. \left(\sum_{u \in G_i} so_{uv} \right), 1 \leq i \leq R \right\}$. A tie is broken by choosing the group with the lowest number of machines.

3.4. GT Efficiency

This work uses the GT efficiency measurement equation defined by Chandrasekharan and Rajagopalan [65], while letting q be equal to 0.5, as follows:

$$\eta = q\eta_1 + (1-q)\eta_2 = q\frac{e_b}{\sum \quad Q_i P_i} + (1-q)\left(1 - \frac{e_0}{uv - \sum \quad Q_i P_i}\right) \tag{9}$$

There are two concerns: the "1" in the group, and the "1" not in the group. The higher the GT efficiency, the more 1s are in the groups, and the fewer 1s are not in the groups. Thus, $\frac{e_b}{\sum \quad NS_i NO_i}$ is the ratio of the number of 1s in the groups to the number of 1s and 0s in all groups, while $\frac{e_0}{uv - \sum \quad NS_i NO_i}$ is the ratio of the number of 1s not in the groups to the number of 1s and 0s not in the groups.

4. Numerical Examples

4.1. Example 1: Number of Operators at Each Workstation Decided

On an assembly line, 500 operators must be assigned to workstations (Figure 6).

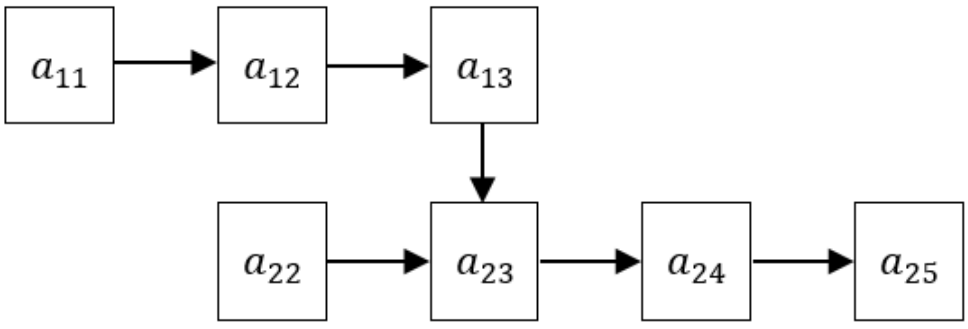

Figure 6. Assembly line with seven workstations.

Let

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & & \\ & a_{22} & a_{23} & a_{24} & a_{25} \end{bmatrix}$$

and

$$\{t_{ij}\} = \begin{bmatrix} 10 & 30 & 24 & 0 & 0 \\ 0 & 5 & 40 & 20 & 15 \end{bmatrix}$$

Let the work time per day be 8 h.

Thus,

$$\{\gamma_{ij}\} = \begin{bmatrix} 48 & 16 & 20 & 0 & 0 \\ 0 & 96 & 12 & 24 & 32 \end{bmatrix}$$

Thus, from Equation (6), $\lambda_{line} = 5000/3$, and from Equation (8),

$$\{p_{ij}\} \underset{\sim}{chop\ off} \begin{bmatrix} 34 & 104 & 83 & & \\ & 17 & 138 & 69 & 52 \end{bmatrix}$$

The approximate operators needed at each workstation are as follows (Table 4).

Table 4. Approximate operators needed at each workstation.

	a_{11}	a_{12}	a_{13}	a_{22}	a_{23}	a_{24}	a_{25}	
p_{ij}	34	104	83	17	138	69	52	$\sum p_{ij} = 497$
Υ_{ij}	1632	1664	1660	1632	1656	1656	1664	$\Upsilon_{line} = 1632$

The traditional step-by-step method is adjusted to obtain Table 5. Stations a_{11} and a_{22} are the bottleneck stations. Thus, if only one worker is assigned (the 498th one) to station a_{11} or station a_{22}, the assembly line cannot function. The production rate is increased if another worker is assigned to station a_{11} and another is assigned to station a_{22}. The final ALB solution is as follows (Table 5).

Table 5. Final ALB solution proposed by this study.

	a_{11}	a_{12}	a_{13}	a_{22}	a_{23}	a_{24}	a_{25}	
p_{ij}	35	104	83	18	138	69	52	$\sum p_{ij} = 499$
Υ_{ij}	1680	1664	1660	1728	1656	1656	1664	$\Upsilon_{line} = 1656$

4.2. Example 2: Auditing Production Rate

The proposed method can be used to audit the production rate, which could be increased by rearranging the operators at each station. Consider this numerical example: An assembly line has seven workstations and 1100 workers (Figure 7).

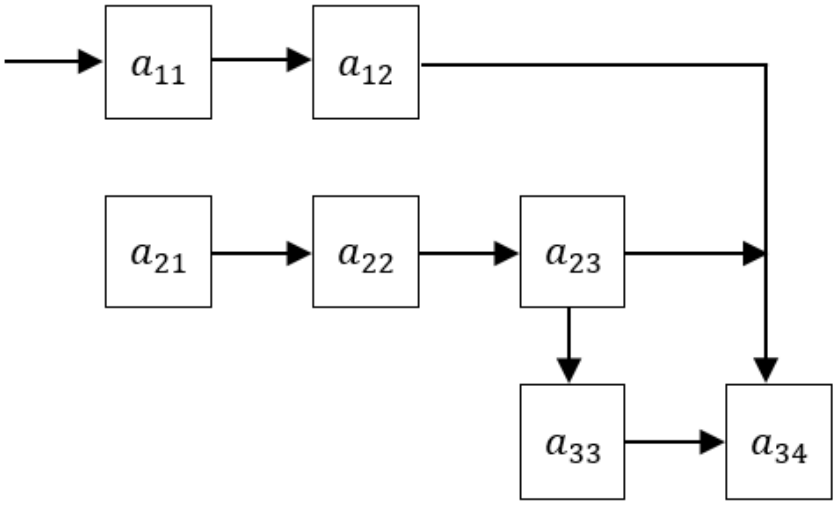

Figure 7. Assembly line with seven workstations.

Let

$$A = \begin{bmatrix} a_{11} & a_{12} & & \\ a_{21} & a_{22} & a_{23} & \\ & & a_{33} & a_{34} \end{bmatrix}$$

$$\{t_{ij}\} = \begin{bmatrix} 10 & 30 & & \\ 20 & 40 & 15 & \\ & & 24 & 48 \end{bmatrix}$$

and

$$\{p_{ij}\} = \begin{bmatrix} 70 & 180 & & \\ 120 & 200 & 80 & \\ & & 150 & 300 \end{bmatrix}$$

Let the work hours per day be 8 h. Thus, the following matrix is obtained:

$$\{\gamma_{ij}\} = \begin{bmatrix} 48 & 16 & & \\ 24 & 12 & 32 & \\ & & 20 & 10 \end{bmatrix}$$

From Equation (2), the following matrix is obtained:

$$\{\lambda_{ij}\} = \begin{bmatrix} 3360 & 2880 & & \\ 2880 & 2400 & 2560 & \\ & & 3000 & 3000 \end{bmatrix}$$

where λ_{line} = 2400. If the workers are rearranged, the output rate is increased without any increase in cost. Equation (6) is used in the method proposed in this study: λ_{line} = 528,000/187. Thus, the following matrix is obtained:

$$A = \begin{array}{c} \\ m_1 \\ m_2 \\ m_3 \\ m_4 \\ m_5 \\ m_6 \\ m_7 \\ m_8 \\ m_9 \\ m_{10} \\ m_{11} \\ m_{12} \\ m_{13} \\ m_{14} \\ m_{15} \\ m_{16} \\ m_{17} \\ m_{18} \\ m_{19} \\ m_{20} \end{array} \begin{array}{c} \begin{array}{ccccccccccccccccccccccccccccccccccccc} p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 & p_8 & p_9 & p_{10} & p_{11} & p_{12} & p_{13} & p_{14} & p_{15} & p_{16} & p_{17} & p_{18} & p_{19} & p_{20} & p_{21} & p_{22} & p_{23} & p_{24} & p_{25} & p_{26} & p_{27} & p_{28} & p_{29} & p_{30} & p_{31} & p_{32} & p_{33} & p_{34} & p_{35} \end{array} \\ \begin{bmatrix} 1 & & & 1 & & & 1 & & 1 & 1 & & 1 & & 1 & & 1 & 1 & 1 & & & 1 & & 1 & 1 & 1 \\ & 1 & & & & & & 1 & & 1 & 1 & & & 1 & 1 & & & & 1 & & 1 & & & 1 & & & & \\ 1 & & 1 & & 1 & & & & & & & 1 & & & & & & & & & & & 1 & & 1 & & & & \\ & 1 & & & & 1 & & & & 1 & 1 & & & & & & & & & 1 & & 1 & & & & & & & \\ & & & & 1 & & & & & & & 1 & & 1 & & & & & & 1 & & 1 & & & & & & & \\ & & & & & & 1 & & & & & 1 & & & 1 & & & & & & 1 & & & & 1 & & & & \\ 1 & & 1 & & 1 & & 1 & & & 1 & & & 1 & & 1 & 1 & & 1 & 1 & 1 & & 1 & & & 1 & 1 & 1 & 1 & & & \\ 1 & & & 1 & & & & 1 & & & 1 & 1 & & & 1 & & 1 & & 1 & & & & & & & & & & \\ & & & & & & 1 & & & & 1 & & & & & 1 & & & & & & & & & & & & & & 1 \\ & & & & & & 1 & & & & 1 & & 1 & & & 1 & & & & & & & & & & & & & & \\ & & 1 & & 1 & & & 1 & & & & 1 & & & & & & & & & 1 & & & & & & & & 1 \\ & & 1 & & 1 & & & 1 & & 1 & & & & & & & & & & & & & & & & & & 1 & & \\ & 1 & & & & & & & & & & & 1 & 1 & & & & & & & 1 & & & & & & & & \\ & 1 & & & & 1 & & & 1 & & & & & 1 & & & & & & & 1 & & 1 & & & & & & \\ & & 1 & & 1 & & & 1 & & & & 1 & & & & & & & & & & & & & 1 & & & & \\ & & 1 & & 1 & & & 1 & & & & & & & & 1 & & & & & 1 & & & 1 & & & & \\ 1 & & 1 & & 1 & & & & & & & & & 1 & & 1 & & & & & 1 & & & & & & & & 1 \\ & 1 & & & & 1 & & & & 1 & 1 & & & & & & 1 & & & & & 1 & & & & & & & \\ & & 1 & & & & 1 & & 1 & & & 1 & & & & & & & & & & & & & & & & & \\ & & & & & 1 & & & & 1 & & & & & & & & & & & & & & & & & & & \end{bmatrix} \end{array}$$

The traditional step-by-step method is adjusted to derive the final arrangement (Table 6):

Table 6. Final ALB solution proposed by this study.

	a_{11}	a_{12}	a_{21}	a_{22}	a_{23}	a_{33}	a_{34}	
p_{ij}	59	176	118	235	88	141	282	$\sum p_{ij}$ = 1099
Υ_{ij}	2832	2816	2832	2820	2816	2820	2820	Υ_{line} = 2816

4.3. Example 3: Comparison with Boe and Cheng's Algorithm

This example compares the proposed algorithm with Boe and Cheng's close neighbor algorithm [66]. The original part–machine incidence matrix is taken from Boe and Cheng as follows:

4.3.1. Machine Sorting

First, the relationship between the two machines must be obtained. For example, Equation (8) is used to obtain b_{37}.

	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	p_{13}	p_{14}	p_{15}	p_{16}	p_{17}	p_{18}	p_{19}	p_{20}	p_{21}	p_{22}	p_{23}	p_{24}	p_{25}	p_{26}	p_{27}	p_{28}	p_{29}	p_{30}	p_{31}	p_{32}	p_{33}	p_{34}	p_{35}
m_3	1	1	1											1																1	1				
m_7	1	1	1	1							1			1		1			1	1		1	1	1		1				1	1	1	1		

$$b_{37} = 1 \times 1 + 0 \times 0 + 1 \times 1 + \ldots + 0 \times 0 = 6$$

Thus, the closeness matrix of skills is as follows.

$$B_u = \begin{array}{c} \\ m_1 \\ m_2 \\ m_3 \\ m_4 \\ m_5 \\ m_6 \\ m_7 \\ m_8 \\ m_9 \\ m_{10} \\ m_{11} \\ m_{12} \\ m_{13} \\ m_{14} \\ m_{15} \\ m_{16} \\ m_{17} \\ m_{18} \\ m_{19} \\ m_{20} \end{array} \begin{array}{c} \begin{array}{cccccccccccccccccccc} m_1 & m_2 & m_3 & m_4 & m_5 & m_6 & m_7 & m_8 & m_9 & m_{10} & m_{11} & m_{12} & m_{13} & m_{14} & m_{15} & m_{16} & m_{17} & m_{18} & m_{19} & m_{20} \end{array} \\ \begin{bmatrix} 0 & 2 & 2 & 1 & 2 & 3 & 7 & 5 & 4 & 1 & 5 & 2 & 1 & 2 & 3 & 4 & 5 & 2 & 3 & 2 \\ 2 & 0 & 1 & 5 & 0 & 1 & 4 & 0 & 1 & 1 & 1 & 0 & 4 & 8 & 0 & 2 & 0 & 5 & 0 & 1 \\ 2 & 1 & 0 & 0 & 1 & 0 & 6 & 3 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 4 & 1 & 0 & 0 \\ 1 & 5 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 1 & 0 & 4 & 6 & 0 & 1 & 0 & 5 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 1 & 4 & 3 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 3 \\ 3 & 1 & 0 & 0 & 1 & 0 & 3 & 1 & 3 & 4 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 3 \\ 7 & 4 & 6 & 3 & 4 & 3 & 0 & 6 & 3 & 3 & 3 & 0 & 2 & 4 & 1 & 4 & 6 & 5 & 2 & 3 \\ 5 & 0 & 3 & 0 & 3 & 1 & 6 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 6 & 1 & 1 & 1 \\ 4 & 1 & 0 & 0 & 2 & 3 & 3 & 1 & 0 & 3 & 2 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 4 \\ 1 & 1 & 0 & 0 & 2 & 4 & 3 & 0 & 3 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 4 \\ 5 & 1 & 0 & 1 & 0 & 1 & 3 & 1 & 2 & 1 & 0 & 3 & 1 & 1 & 5 & 5 & 1 & 1 & 5 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 3 & 0 & 0 & 0 & 5 & 3 & 0 & 0 & 4 & 0 \\ 1 & 4 & 0 & 4 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 4 & 0 & 1 & 0 & 4 & 0 & 0 \\ 2 & 8 & 1 & 6 & 0 & 0 & 4 & 0 & 0 & 0 & 1 & 0 & 4 & 0 & 0 & 2 & 0 & 6 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 5 & 5 & 0 & 0 & 0 & 4 & 0 & 0 & 6 & 0 \\ 4 & 2 & 0 & 1 & 1 & 1 & 4 & 1 & 1 & 1 & 5 & 3 & 1 & 2 & 4 & 0 & 0 & 1 & 4 & 1 \\ 5 & 0 & 4 & 0 & 2 & 1 & 6 & 6 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 2 & 5 & 1 & 5 & 0 & 0 & 5 & 1 & 0 & 0 & 1 & 0 & 4 & 6 & 0 & 1 & 1 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 1 & 2 & 1 & 1 & 0 & 5 & 4 & 0 & 0 & 6 & 4 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 & 3 & 3 & 3 & 1 & 4 & 4 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

In step 1 of machine sorting, the first machine, belonging to the first group, is m_2. Thus, $G_1 = \{m_2\}$. Ignoring m_2, the next step is to form a new closeness matrix of skill $B'u = \{b_{kl}\}$, as follows:

$$B'_u = \begin{array}{c} \\ m_1 \\ m_3 \\ m_4 \\ m_5 \\ m_6 \\ m_7 \\ m_8 \\ m_9 \\ m_{10} \\ m_{11} \\ m_{12} \\ m_{13} \\ m_{14} \\ m_{15} \\ m_{16} \\ m_{17} \\ m_{18} \\ m_{19} \\ m_{20} \end{array} \begin{array}{c} \begin{array}{cccccccccccccccccccc} m_1 & m_2 & m_3 & m_4 & m_5 & m_6 & m_7 & m_8 & m_9 & m_{10} & m_{11} & m_{12} & m_{13} & m_{14} & m_{15} & m_{16} & m_{17} & m_{18} & m_{19} & m_{20} \end{array} \\ \begin{bmatrix} 0 & 2 & 2 & 1 & 2 & 3 & 7 & 5 & 4 & 1 & 5 & 2 & 1 & 2 & 3 & 4 & 5 & 2 & 3 & 2 \\ 2 & 1 & 0 & 0 & 1 & 0 & 6 & 3 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 4 & 1 & 0 & 0 \\ 1 & 5 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 1 & 0 & 4 & 6 & 0 & 1 & 0 & 5 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 1 & 4 & 3 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 3 \\ 3 & 1 & 0 & 0 & 1 & 0 & 3 & 1 & 3 & 4 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 3 \\ 7 & 4 & 6 & 3 & 4 & 3 & 0 & 6 & 3 & 3 & 3 & 0 & 2 & 4 & 1 & 4 & 6 & 5 & 2 & 3 \\ 5 & 0 & 3 & 0 & 3 & 1 & 6 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 6 & 1 & 1 & 1 \\ 4 & 1 & 0 & 0 & 2 & 3 & 3 & 1 & 0 & 3 & 2 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 4 \\ 1 & 1 & 0 & 0 & 2 & 4 & 3 & 0 & 3 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 4 \\ 5 & 1 & 0 & 1 & 0 & 1 & 3 & 1 & 2 & 1 & 0 & 3 & 1 & 1 & 5 & 5 & 1 & 1 & 5 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 3 & 0 & 0 & 0 & 5 & 3 & 0 & 0 & 4 & 0 \\ 1 & 4 & 0 & 4 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 4 & 0 & 1 & 0 & 4 & 0 & 0 \\ 2 & 8 & 1 & 6 & 0 & 0 & 4 & 0 & 0 & 0 & 1 & 0 & 4 & 0 & 0 & 2 & 0 & 6 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 5 & 5 & 0 & 0 & 0 & 4 & 0 & 0 & 6 & 0 \\ 4 & 2 & 0 & 1 & 1 & 1 & 4 & 1 & 1 & 1 & 5 & 3 & 1 & 2 & 4 & 0 & 0 & 1 & 4 & 1 \\ 5 & 0 & 4 & 0 & 2 & 1 & 6 & 6 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 2 & 5 & 1 & 5 & 0 & 0 & 5 & 1 & 0 & 0 & 1 & 0 & 4 & 6 & 0 & 1 & 1 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 1 & 2 & 1 & 1 & 0 & 5 & 4 & 0 & 0 & 6 & 4 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 & 3 & 3 & 3 & 1 & 4 & 4 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

The next machine is m_{14}, which belongs to G_1 because $b_{m_{14}m_2}$ is the largest number in $B'u$. Thus, $G_1 = \{m_2,\ m_{14}\}$. The new closeness matrix of skill $B'u = \{b_{kl}\}$ is as follows:

$$B'_u = \begin{array}{c} \\ m_1 \\ m_3 \\ m_4 \\ m_5 \\ m_6 \\ m_7 \\ m_8 \\ m_9 \\ m_{10} \\ m_{11} \\ m_{12} \\ m_{13} \\ m_{15} \\ m_{16} \\ m_{17} \\ m_{18} \\ m_{19} \\ m_{20} \end{array} \begin{array}{c} \begin{array}{cccccccccccccccccccc} m_1 & m_2 & m_3 & m_4 & m_5 & m_6 & m_7 & m_8 & m_9 & m_{10} & m_{11} & m_{12} & m_{13} & m_{14} & m_{15} & m_{16} & m_{17} & m_{18} & m_{19} & m_{20} \end{array} \\ \begin{bmatrix} 0 & 2 & 2 & 1 & 2 & 3 & 7 & 5 & 4 & 1 & 5 & 2 & 1 & 2 & 3 & 4 & 5 & 2 & 3 & 2 \\ 2 & 1 & 0 & 0 & 1 & 0 & 6 & 3 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 4 & 1 & 0 & 0 \\ 1 & 5 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 1 & 0 & 4 & 6 & 0 & 1 & 0 & 5 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 1 & 4 & 3 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 3 \\ 3 & 1 & 0 & 0 & 1 & 0 & 3 & 1 & 3 & 4 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 3 \\ 7 & 4 & 6 & 3 & 4 & 3 & 0 & 6 & 3 & 3 & 3 & 0 & 2 & 4 & 1 & 4 & 6 & 5 & 2 & 3 \\ 5 & 0 & 3 & 0 & 3 & 1 & 6 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 6 & 1 & 1 & 1 \\ 4 & 1 & 0 & 0 & 2 & 3 & 3 & 1 & 0 & 3 & 2 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 4 \\ 1 & 1 & 0 & 0 & 2 & 4 & 3 & 0 & 3 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 4 \\ 5 & 1 & 0 & 1 & 0 & 1 & 3 & 1 & 2 & 1 & 0 & 3 & 1 & 1 & 5 & 5 & 1 & 1 & 5 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 3 & 0 & 0 & 0 & 5 & 3 & 0 & 0 & 4 & 0 \\ 1 & 4 & 0 & 4 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 4 & 0 & 1 & 0 & 4 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 5 & 5 & 0 & 0 & 0 & 4 & 0 & 0 & 6 & 0 \\ 4 & 2 & 0 & 1 & 1 & 1 & 4 & 1 & 1 & 1 & 5 & 3 & 1 & 2 & 4 & 0 & 0 & 1 & 4 & 1 \\ 5 & 0 & 4 & 0 & 2 & 1 & 6 & 6 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 2 & 5 & 1 & 5 & 0 & 0 & 5 & 1 & 0 & 0 & 1 & 0 & 4 & 6 & 0 & 1 & 1 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 1 & 2 & 1 & 1 & 0 & 5 & 4 & 0 & 0 & 6 & 4 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 & 3 & 3 & 3 & 1 & 4 & 4 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

The third machine is m_1, which belongs to a new group, G_2, because $b_{m_1m_7}$ is the largest number in $B'u$, but machine m_7 does not belong to any group. Thus, $G_2 = \{m_1\}$. Applying steps 2–3 in Section 3.3.1 gives four groups: $G_1 = \{m_2,\ m_{14},\ m_4,\ m_{18},\ m_{13}\ \}$, $G_2 = \{m_1,\ m_7,\ m_3,\ m_8,\ m_{17},\ m_5,\ m_9\}$,$G_3 = \{m_{15},\ m_{19},\ m_{11},\ m_{12},\ m_{16}\ \}$, and $G_4 = \{m_6,\ m_{10},\ m_{20}\}$.

4.3.2. Part Sorting

By using the operator sorting step in Section 3.3.2, the parts can be sorted. For example, p_{24}, the $\sum_{u \in G_1} so_{up_{24}} = 5$, and 2, 1, and 0 in G_2, G_3, and G_4, respectively. Thus, p_{24} belongs to G_1.

Finally, the four groups are ordered (Table 7).

Table 7. The four groups in example 3.

Groups	Group Members
G_1	m_2,m_{14},m_4,m_{18},m_{13},p_2,p_7,p_{10},p_{12},p_{13},p_{18},p_{24},p_{27},p_{31}
G_2	m_1,m_7,m_3,m_8,m_{17},m_5,m_9,p_1,p_3,p_5,p_{15},p_{17},p_{20},p_{23},p_{25},p_{29},p_{34},p_{35}
G_3	m_{15},m_{19},m_{11},m_{12},m_{16},p_4,p_6,p_9,p_{11},p_{21},p_{28},p_{30},p_{32},p_{33}
G_4	m_6,m_{10},m_{20},p_8,p_{14},p_{16},p_{19},p_{22},p_{26},

The sorted incidence matrix is as follows:

$A^{sort} =$

	p_2	p_7	p_{10}	p_{12}	p_{13}	p_{18}	p_{24}	p_{27}	p_{31}	p_1	p_3	p_5	p_{15}	p_{17}	p_{20}	p_{23}	p_{25}	p_{29}	p_{34}	p_{35}	p_4	p_6	p_9	p_{11}	p_{21}	p_{28}	p_{30}	p_{32}	p_{33}	p_8	p_{14}	p_{16}	p_{19}	p_{22}	p_{26}
m_2	1		1	1	1	1	1	1	1																								1		
m_{14}	1	1	1	1	1	1	1	1	1																										
m_4	1	1		1	1		1	1																											
m_{18}	1	1		1	1		1		1						1																				
m_{13}	1			1	1		1																												
m_1						1	1			1		1		1	1	1		1	1		1		1			1	1			1					
m_7		1		1			1		1	1	1	1	1	1	1	1		1								1	1					1	1	1	
m_3									1	1	1	1	1					1																	
m_8										1		1	1	1	1	1	1						1												
m_{17}										1	1	1	1	1	1	1		1			1														
m_5												1		1		1															1				1
m_9																1					1									1	1		1		
m_{15}																					1	1	1	1	1	1	1								
m_{19}																					1		1	1	1	1	1	1							
m_{11}						1														1	1	1	1			1	1	1				1			
m_{12}																					1	1	1	1	1				1						
m_{16}				1		1															1	1	1			1	1								1
m_6																1										1				1	1		1	1	
m_{10}																														1	1	1	1	1	1
m_{20}																1														1	1		1		1

Boe and Cheng's solution of this example is as follows:

$$A^{sort} = \begin{array}{c} \\ m_7 \\ m_1 \\ m_{11} \\ m_{16} \\ m_{19} \\ m_{15} \\ m_{12} \\ m_8 \\ m_{17} \\ m_3 \\ m_2 \\ m_{14} \\ m_{18} \\ m_4 \\ m_{13} \\ m_9 \\ m_{20} \\ m_{10} \\ m_6 \\ m_5 \end{array} \begin{array}{c} \begin{array}{ccccccccccccccccccccccccccccccccccc} p_{30} & p_{32} & p_6 & p_{34} & p_{35} & p_4 & p_9 & p_{11} & p_{21} & p_{28} & p_{33} & p_1 & p_5 & p_{15} & p_{17} & p_{20} & p_{25} & p_3 & p_{29} & p_{31} & p_2 & p_{10} & p_{12} & p_{13} & p_{18} & p_{24} & p_{27} & p_7 & p_8 & p_{14} & p_{19} & p_{23} & p_{26} & p_{16} & p_{22} \end{array} \\ \left[\begin{array}{ccccccccccccccccccccccccccccccccccc} 1 & 1 & & & & & & & & & & 1 & 1 & 1 & 1 & 1 & & 1 & 1 & 1 & & 1 & & 1 & & 1 & & & & 1 & 1 & 1 & & 1 \\ 1 & 1 & 1 & 1 & 1 & & & 1 & & & & 1 & & 1 & & 1 & 1 & & & & & & & 1 & 1 & & & 1 & & 1 & & & \\ 1 & 1 & 1 & & 1 & 1 & 1 & & & 1 & & & & & & & & & & & & & & & & 1 & & & & & & 1 & \\ 1 & 1 & 1 & & & 1 & 1 & & & & & & & & & & & & & & 1 & 1 & & & & & & & & & 1 & & \\ 1 & 1 & & & & 1 & 1 & 1 & 1 & 1 & & & & & & & & & & & & & & & & & & & & & & & \\ 1 & & 1 & & & 1 & 1 & 1 & 1 & 1 & & & & & & & & & & & & & & & & & & & & & & & \\ & & 1 & & & 1 & 1 & 1 & 1 & & 1 & & & & & & & & & & & & & & & & & & & & & & \\ & & & & & & 1 & & & & & 1 & 1 & 1 & 1 & 1 & 1 & & & & & & & & & & & & & & & 1 & & \\ & & & & 1 & & & & & & & 1 & 1 & 1 & 1 & 1 & 1 & 1 & & & & & & & & & & & & & & & & \\ & & & & & & & & & & & 1 & 1 & 1 & & & & 1 & 1 & 1 & & & & & & & & & & & & & & \\ & & & & & & & & & & & & & & & & & & & & 1 & 1 & 1 & & & & & & & & & & \\ & & & & & & & & & & & & & & & & & & & & 1 & 1 & 1 & 1 & & & & & & & & & \\ & & & & & & & & & & & & & & & & & & & & 1 & 1 & 1 & & & & & & & & & & \\ & & & & & & & & & & & & & & & & & & & & 1 & 1 & 1 & & & & & & & & & & \\ & 1 & & 1 & & & & & & & & & & & & & & & & & & & & & & & & 1 & 1 & 1 & 1 & & & \\ & & & & & & & & & & & & & & & & & & & & & & & & & & & & 1 & 1 & 1 & 1 & 1 & \\ & & & & & & & & & & & & & & & & & & & & & & & & & & & & 1 & 1 & 1 & & 1 & 1 & 1 \\ 1 & & & & & & & & & & & & & & & 1 & & & & & & & & & & & & & & 1 & 1 & 1 & & & 1 \\ & & & & & & & & & & & & 1 & & 1 & & & & & & & & & & & & & & & & & 1 & & 1 & 1 & \end{array}\right] \end{array}$$

The GT efficiency of the proposed algorithm is obtained using Equation (9), as follows:

$$\eta = 0.5 \times \frac{33 + 40 + 31 + 14}{5 \times 9 + 7 \times 11 + 5 \times 9 + 3 \times 6} + 0.5 \times \left(1 - \frac{35}{20 \times 35 - (5 \times 9 + 7 \times 11 + 5 \times 9 + 3 \times 6)}\right) = 78.5\%$$

In the same way, GT efficiency is calculated using Boe and Cheng's algorithm, as follows:

$$\eta = 0.5 \times \frac{40 + 18 + 33 + 22}{7 \times 11 + 3 \times 8 + 5 \times 9 + 5 \times 7} + 0.5 \times \left(1 - \frac{40}{20 \times 35 - (7 \times 11 + 3 \times 8 + 5 \times 9 + 5 \times 7)}\right) = 77.4\%$$

4.4. Example 4: Reducing Training Time

This example demonstrates how to find an operator with the training and skills that matches a workstation's skill requirements so that the operator does not need to take any training courses. Consider the problem displayed in Figure 3 (Section 2). The proposed algorithm is used to obtain a sorted skill-operator matrix, as follows:

$$SO^{sort} = \begin{array}{c} \\ s_3 \\ s_5 \\ s_2 \\ s_7 \\ s_4 \\ s_1 \\ s_6 \end{array} \begin{array}{c} \begin{array}{cccccccccccccccc} o_7 & o_{13} & o_4 & o_{15} & o_{11} & o_3 & o_9 & o_{12} & o_{14} & o_2 & o_1 & o_{10} & o_6 & o_5 & o_8 \end{array} \\ \left[\begin{array}{cccccccccccccccc} 1 & 1 & 1 & 1 & 1 & 1 & & & & & & & & & \\ 1 & 1 & 1 & 1 & & & 1 & & & & & & & & \\ & & & & & & & 1 & 1 & 1 & 1 & & & & \\ & & & & & & & 1 & 1 & 1 & & 1 & & & \\ 1 & & & & & & & 1 & & & 1 & 1 & 1 & & \\ 1 & & & & 1 & & & & 1 & & & & & 1 & 1 \\ & 1 & & & & & & & & & & & 1 & 1 & \end{array}\right] \end{array}$$

A manager could use the sorted skill–operator matrix as a database for choosing an operator with the right training and skills. Based on the database, the manager could quickly rotate a suitable worker to a workstation. Based on the sorted skill–operator matrix SO^{sort}, the manager could choose operator o_{11} for the workstation that requires skill s_1; o_2,

o_{12}, and o_{14} because of s_2; o_4, o_7, o_{13}, and o_3, because of s_3; o_1 and o_6 because of s_4; o_{15} and o_9 because of s_5; o_5 and o_8 because of s_6; and o_{10} because of s_7, as follows:

$$SO = \begin{array}{c|cccccccccccccccc} & o_{1_1} & o_2 & o_{1_2} & o_{1_4} & o_4 & o_7 & o_{1_3} & o_3 & o_1 & o_6 & o_{1_5} & o_9 & o_5 & o_8 & o_{1_0} \\ s_1 & \boxed{1} & & & 1 & & 1 & & & & & & & 1 & 1 & \\ s_2 & & \boxed{1} & 1 & 1 & & & & & 1 & & & & & & \\ s_3 & 1 & & & & \boxed{1} & 1 & 1 & 1 & & & 1 & & & & \\ s_4 & & & 1 & & & 1 & & & \boxed{1} & 1 & & & & & 1 \\ s_5 & & & & & 1 & 1 & 1 & & & & \boxed{1} & 1 & & & \\ s_6 & & & & & & & 1 & & & 1 & & & \boxed{1} & 1 & \\ s_7 & & 1 & 1 & 1 & & & & & & & & & & & \boxed{1} \end{array}$$

Operators in the same skills group can support other operators in the group.

5. Results and Discussion

In example 1, stations a_{23} and a_{24} are the bottleneck stations. However, only one operator, namely the 500th one, can be assigned. Assigning the 500th operator to either station a_{23} or a_{24} cannot increase the output of the assembly line. This study suggests that the 500th worker should be laid off or shifted to another department. Alternatively, a 501st operator could be hired, which would allow another worker to be assigned to station a_{23} and another to be assigned to station a_{24}. This example demonstrates the proposed method in detail. The maximum output rate and the number of workers at each workstation can be decided. It also shows that multiple workstations can simultaneously become bottleneck stations.

In example 2, stations a_{12} and a_{23} are the bottleneck stations, and only one operator can be assigned. This study suggests that the 1100th worker be laid off or shifted to another department. Alternatively, a 1101st operator could be hired, which would allow another worker to be assigned to station a_{12} and another to be assigned to station a_{23}. With these solutions, a higher production rate than in the original arrangement (2816 > 2400) is achieved without any increase in cost. It shows that a manager can improve productivity by rearranging the number of workers at each workstation, while not increasing costs. It will also cut down on worker complaints because the workers' workloads are balanced.

In example 3, the proposed algorithm is used to categorize four groups, as shown in Section 4.3. Boe and Cheng's algorithm is compared with the GT efficiency measure proposed by Chandrasekharan and Rajagopalan. The GT efficiency of the proposed algorithm is 78.5%, higher than the 77.4% of Boe and Cheng's algorithm. Thus, the proposed GT algorithm works and can perform better than other algorithms in the literature. Workers are assigned to the workstation that best uses their skills, so they feel that their contribution is meaningful and more substantial. Workers at the same workstation also share more skills in common. This makes it easier for the manager to alternate workers.

In example 4, operators within the same skill group can support each other, giving managers more worker rotation solutions. With workers suggested by the sorted skill–operator matrix, managers can quickly match suitable operators with workstations so that productivity remains high. Using the matrix to match workers' training and skills to workstations' skill requirements also enables managers to reduce operator training time and cover workers' absences. For example, if o_4 is absent or busy, then o_{15} can support or share the workload without needing too much training. The risk of an unexpected worker absence or resignation can be mitigated by alternating the workers according to the sorted skill–operator matrix.

6. Conclusions

Company managers aim to achieve sustainable operations, part of which is managing labor costs. To reduce labor costs, an assembly line manager increases the line's efficiency and reduces the training time needed by operators. Inadequate training or distractions,

such as an operator focused on a personal problem, can be dangerous, something a plant manager seeks to eliminate, while worker absences hurt efficiency and increase costs. This paper presents an algorithm that helps managers keep assembly lines balanced by arranging operators, so a line's workstations achieve the same output rates. The first example demonstrates how the proposed method can decide the maximum output rate and the number of workers at each workstation given a constant number of operators, especially when setting up a new plant. The second example demonstrates how the method can audit the existing number of operators at each workstation. Using the method to readjust the number of operators at a workstation can increase the output rate, without raising costs. As indicated in the numerical examples, another challenge is that multiple workstations can simultaneously become bottleneck stations. Knowing the number of operators at each workstation, the proposed cluster algorithm can group operators with similar skills and training, which reduces the training time. The third example demonstrates how the proposed clustering algorithm performs better than a previously described algorithm. Within the same group, operators from one workstation can also support operators at other workstations, as demonstrated in example 4. The clustering algorithm could be used by the human resources department to make a database that includes all of the company employees. When an employee is absent, a manger could quickly find a replacement, mitigating the risk of an unexpected absence. The manager could also use such a database to determine which employees should take a training course. By not having all of the employees take the training course, training costs could be reduced. Future applications of the proposed method could include the creation of a course-faculty database, where it could be used to find a replacement instructor for a class when the assigned instructor is absent.

Traditional GT considers only machines and parts, but this study uses the concept of GT to reduce the training time of workstation operators so that managers can achieve sustainable operations. The sorted skill–operator matrix in this study can, similar to a database, provide a manager with suggestions. However, most managers prefer a direct solution to a problem; they want the specific worker whose training and skills match the skill requirements of the problem workstation. Thus, future studies could develop an algorithm to assign specific workers to specific workstations by using a sorted skill–operator matrix, especially for large factories with multiple assembly lines.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The author declares no conflict of interest.

Nomenclature

$A = \left\{a_{ij}\right\}_{mn}$	Matrix of workstations
t_{ij}	Standard time of workstation a_{ij}
γ_{ij}	Output of workstation a_{ij} per operator per day
p_{ij}	Number of operators at workstation a_{ij}
λ_{ij}	Output of workstation a_{ij} per day
λ_{line}	Output of assembly line per day
TP	Total number of operators
T	Work time per day
$SO = \left\{so_{ij}\right\}_{uv}$	Matrix of the skill-operator
u	Number of skills
v	Number of operators
$B_u = \{b_{rs}\}_{uu}$	Closeness matrix of skills

$B\prime_u = \{b_{ij}\}$	New closeness matrix of skills without consideration of sorted rows
G_i	group
R	Total number of groups
η	Group efficiency
e_b	Total number of 1s in the groups
e_0	Total number of 1s not in the groups
NS_i	Number of skills in the ith group
NO_i	Number of operators in the ith group
	Weighting factor

References

1. Netessine, S.; Taylor, T.A. Product line design and production technology. *Mark. Sci.* **2007**, *26*, 101–117. [CrossRef]
2. Sabuncuoglu, I.; Erel, E.; Tanyer, M. Assembly line balancing using genetic algorithms. *J. Intell. Manuf.* **2000**, *11*, 295–310. [CrossRef]
3. Koo, P.-H. A New Self-Balancing Assembly Line Based on Collaborative Ant Behavior. *Appl. Sci.* **2020**, *10*, 6845. [CrossRef]
4. Adeppa, A. A Study on Basics of Assembly Line Balancing. *Int. J. Emerg. Technol.* **2015**, *6*, 294–297.
5. Nagy, L.; Ruppert, T.; Abonyi, J. Analytic Hierarchy Process and Multilayer Network-Based Method for Assembly Line Balancing. *Appl. Sci.* **2020**, *10*, 3932. [CrossRef]
6. Suer, G.A. Designing parallel assembly lines. *Comput. Ind. Eng.* **1998**, *35*, 467–470. [CrossRef]
7. Kim, S.; Lee, Y.H.; Yang, T.; Park, N. Robust production control policies considering WIP balance and setup time in a semiconductor fabrication line. *Int. J. Adv. Manuf. Technol.* **2008**, *39*, 333–343. [CrossRef]
8. Yuan, M.; Yu, H.; Huang, J.; Ji, A. Reconfigurable assembly line balancing for cloud manufacturing. *J. Intell. Manuf.* **2019**, *30*, 2391–2405. [CrossRef]
9. Rafael, P.; Alberto, G.V.; Manuel, L.; Rafael, M. Metaheuristic procedures for the lexicographic bottleneck assembly line balancing problem. *J. Oper. Res. Soc.* **2015**, *66*, 1815–1825. [CrossRef]
10. Kuroda, M.; Tomita, T. Robust design of a cellular-line production system with unreliable facilities. *Comput. Ind. Eng.* **2005**, *48*, 537–551. [CrossRef]
11. Barthold, J.J.; Eisenstein, D.D. A production line that balances itself. *Oper. Res.* **1996**, *44*, 21–31. [CrossRef]
12. Huang, L.H. Labour assignment and workload balance evaluation for a production line. *Est. J. Eng.* **2009**, *15*, 34–47. [CrossRef]
13. Sawik, T. Mixed integer programming for scheduling flexible flow lines with limited intermediate buffers. *Math. Comput. Mdlng.* **2000**, *31*, 39–52. [CrossRef]
14. Muth, E.J. The reversibility property of production lines. *Manag. Sci.* **1979**, *25*, 152–158. [CrossRef]
15. Zavadlav, E.; McClain, J.O.; Thomas, L.J. Self-buffering, self-balancing, self-flushing production lines. *Manag. Sci.* **1996**, *42*, 1151–1164. [CrossRef]
16. Hazır, O.; Agi, M.A.N.; Guerin, J. A fast and effective heuristic for smoothing workloads on assembly lines: Algorithm design and experimental analysis. *Comput. Oper. Res.* **2020**, *115*, 104857. [CrossRef]
17. Dimitriadis, S.G. Assembly line balancing and group working: A heuristic procedure for workers' groups operating on the same product and workstation. *Comput. Oper. Res.* **2006**, *33*, 2757–2774. [CrossRef]
18. Lian, J.; Liu, C.G.; Li, W.J.; Yin, Y. A multi-skilled worker assignment problem in seru production systems considering the worker heterogeneity. *Comput. Ind. Eng.* **2018**, *118*, 366–382. [CrossRef]
19. Samouei, O.; Fattahi, P.; Ashayeri, J.; Ghazinoory, S. Bottleneck easing-based assignment of work and product mixture determination: Fuzzy assembly line balancing approach. *Appl. Math. Model.* **2016**, *40*, 4323–4340. [CrossRef]
20. Miralles, C.; Garcia-Sabater, J.P.; Andres, C.; Cardos, M. Branch and bound procedures for solving the Assembly Line Worker Assignment and Balancing Problem: Application to Sheltered Work centres for Disabled. *Discrete Appl. Math.* **2008**, *156*, 352–367. [CrossRef]
21. Borba, L.; Ritt, M. A heuristic and a branch-and-bound algorithm for the Assembly Line Worker Assignment and Balancing Problem. *Comput. Oper. Res.* **2014**, *45*, 87–96. [CrossRef]
22. Pearce, B.W.; Antani, K.; Mears, L.; Funk, K.; Mayorga, M.E.; Kurz, M.E. An effective integer program for a general assembly line balancing problem with parallel workers and additional assignment restrictions. *J. Manuf. Syst.* **2019**, *50*, 180–192. [CrossRef]
23. Nourmohammadi, A.; Fathi, M.; Zandieh, M.; Ghobakhloo, M. A Water-Flow Like Algorithm for Solving U-Shaped Assembly Line Balancing Problems. *IEEE Access* **2019**, *7*, 129824–129833. [CrossRef]
24. Pastor, R. LB-ALBP: The lexicographic bottleneck assembly line balancing problem. *Int. J. Prod. Res.* **2011**, *49*, 2425–2442. [CrossRef]
25. Karas, A.; Ozcelik, F. Assembly line worker assignment and rebalancing problem: A mathematical model and an artificial bee colony algorithm. *Comput. Ind. Eng.* **2021**, *156*, 107195. [CrossRef]
26. Oksuz, M.K.; Buyukozkan, K.; Satoglu, S.I. U-shaped assembly line worker assignment and balancing problem: A mathematical model and two meta-heuristics. *Comput. Ind. Eng.* **2017**, *112*, 246–263. [CrossRef]
27. Saif, U.; Guan, Z.; Zhang, L.; Zhang, F.; Wang, B.; Mirza, J. Multi-objective artificial bee colony algorithm for order oriented simultaneous sequencing and balancing of multi-mixed model assembly line. *J. Intell. Manuf.* **2019**, *30*, 1195–1220. [CrossRef]

28. Mutlu, O.; Polat, O.; Supciller, A.A. An iterative genetic algorithm for the assembly line worker assignment and balancing problem of type-II. *Comput. Oper. Res.* **2013**, *40*, 418–426. [CrossRef]
29. Zacharia, P.T.; Nearchou, A.C. A population-based algorithm for the bi-objective assembly line worker assignment and balancing problem. *Eng. Appl. Artif. Intell.* **2016**, *49*, 1–9. [CrossRef]
30. Zacharia, P.T.; Nearchou, A.C. Multi-objective fuzzy assembly line balancing using genetic algorithms. *J. Intell. Manuf.* **2012**, *23*, 615–627. [CrossRef]
31. Vila, M.; Pereira, J. A branch-and-bound algorithm for assembly line worker assignment and balancing problems. *Comput. Oper. Res.* **2014**, *44*, 105–114. [CrossRef]
32. Blum, C.; Miralles, C. On solving the assembly line worker assignment and balancing problem via beam search. *Comput. Oper. Res.* **2011**, *38*, 328–339. [CrossRef]
33. Erel, E.; Sabuncuoglu, I.; Sekerci, H. Stochastic assembly line balancing using beam search. *Int. J. Prod. Res.* **2005**, *43*, 1411–1426. [CrossRef]
34. Sun, B.Q.; Wang, L. A decomposition-based matheuristic for supply chain network design with assembly line balancing. *Comput. Ind. Eng.* **2019**, *131*, 408–417. [CrossRef]
35. Ramezanian, R.; Ezzatpanah, A. Modeling and solving multi-objective mixed-model assembly line balancing and worker assignment problem. *Comput. Ind. Eng.* **2015**, *87*, 74–80. [CrossRef]
36. Zhang, Z.; Tang, Q.H.; Ruiz, R.; Zhang, L. Ergonomic risk and cycle time minimization for the U-shaped worker assignment assembly line balancing problem: A multi-objective approach. *Comput. Oper. Res.* **2020**, *118*, 104905. [CrossRef]
37. Zhang, Z.; Tang, Q.; Chica, M. Multi-manned assembly line balancing with time and space constraints: A MILP model and memetic ant colony system. *Comput. Ind. Eng.* **2020**, *150*, 106862. [CrossRef]
38. Meng, K.; Tang, Q.; Zhang, Z.; Qian, X. An Improved Lexicographical Whale Optimization Algorithm for the Type-II Assembly Line Balancing Problem Considering Preventive Maintenance Scenarios. *IEEE Access* **2020**, *8*, 30421–30435. [CrossRef]
39. Song, B.L.; Wong, W.K.; Chan, S.F. A recursive operator allocation approach for assembly line-balancing optimization problem with the consideration of operator efficiency. *Comput. Ind. Eng.* **2006**, *51*, 585–608. [CrossRef]
40. Mutlu, O.; Ozgormus, E. A fuzzy assembly line balancing problem with physical workload constraints. *Int. J. Prod. Res.* **2012**, *50*, 5281–5291. [CrossRef]
41. Zhong, Y.; Deng, Z.; Xu, K. An effective artificial fish swarm optimization algorithm for two-sided assembly line balancing problems. *Comput. Ind. Eng.* **2019**, *138*, 106121. [CrossRef]
42. Sahin, M.; Kellegoz, T. A new mixed-integer linear programming formulation and particle swarm optimization based hybrid heuristic for the problem of resource investment and balancing of the assembly line with multi-manned workstations. *Comput. Ind. Eng.* **2019**, *133*, 107–120. [CrossRef]
43. Pilati, F.; Ferrari, E.; Gamberi, M.; Margelli, S. Multi-Manned Assembly Line Balancing: Workforce Synchronization for Big Data Sets through Simulated Annealing. *Appl. Sci.* **2021**, *11*, 2523. [CrossRef]
44. Jolai, F.; Jahangoshai Rezaee, M.; Vazifeh, A. Multi-criteria decision making for assembly line balancing. *J. Intell. Manuf.* **2009**, *20*, 113. [CrossRef]
45. Meng, W.; Zhang, X. Optimization of Remanufacturing Disassembly Line Balance Considering Multiple Failures and Material Hazards. *Sustainability* **2020**, *12*, 7318. [CrossRef]
46. Xia, X.; Liu, W.; Zhang, Z.; Wang, L.; Cao, J.; Liu, X. A Balancing Method of Mixed-model Disassembly Line in Random Working Environment. *Sustainability* **2019**, *11*, 2304. [CrossRef]
47. Cao, J.; Xia, X.; Wang, L.; Zhang, Z.; Liu, X. A Novel Multi-Efficiency Optimization Method for Disassembly Line Balancing Problem. *Sustainability* **2019**, *11*, 6969. [CrossRef]
48. Szwarc, E.; Wikarek, J. Proactive Planning of Project Team Members' Competences. *Found. Manag.* **2020**, *12*, 71–84. [CrossRef]
49. Szwarc, E.; Bocewicz, G.; Bach-Dąbrowska, I.; Banaszak, Z. Declarative Model of Competences Assessment Robust to Personnel Absence. In Proceedings of the International Conference on Information and Software Technologies (ICIST 2019), Vilnius, Lithuania, 10–12 October 2019; pp. 12–23.
50. Shahdi-Pashaki, S.; Teymourian, E.; Kayvanfar, V.; Komaki, G.M.; Sajadi, A. Group technology-based model and cuckoo optimization algorithm for resource allocation in cloud computing. *IFAC* **2015**, *48*, 1140–1145. [CrossRef]
51. Zohrevand, A.M.; Rafiei, H.; Zohrevand, A.H. Multi-objective dynamic cell formation problem: A stochastic programming approach. *Comput. Ind. Eng.* **2016**, *98*, 323–332. [CrossRef]
52. McAuley, J. Machine grouping for efficient production. *Prod. Engr.* **1972**, *51*, 53–57. [CrossRef]
53. Li, M.-L.; Parkin, R.E. Group technology revisited: A simple and robust algorithm with enhanced capability. *Int. J. Prod. Res.* **1997**, *35*, 1969–1992. [CrossRef]
54. Boulif, M.; Atif, K. A new branch-&-bound-enhanced genetic algorithm for the manufacturing cell formation problem. *Comput. Oper. Res.* **2006**, *33*, 2219–2245. [CrossRef]
55. Li, M.-L. Arrangement of Machines Under Spatial Constraint by Using a Novel Algorithm. *IEEE Access* **2020**, *8*, 144565–144574. [CrossRef]
56. Agustin-Blas, L.E.; Salcedo-Sanz, S.; Ortiz-Garcia, E.G.; Portilla-Figueras, A.; Perez-Bellido, A.M.; Jimenez-Fernandez, S. Team formation based on group technology: A hybrid grouping genetic algorithm approach. *Comput. Oper. Res.* **2011**, *38*, 484–495. [CrossRef]

57. Hazarika, M.; Laha, D. Machine-part cell formation for maximum grouping efficacy based on genetic algorithm. In Proceedings of the 2015 IEEE Workshop on Computational Intelligence: Theories, Applications and Future Directions (WCI), Kanpur, India, 14–17 December 2015; pp. 1–6.
58. Laha, D.; Hazarika, M. A heuristic approach based on Euclidean distance matrix for the machine-part cell formation problem. *Matl. Tdy. Proc.* **2017**, *4*, 1442–1451. [CrossRef]
59. Li, M.-L. A novel algorithm of cell formation with alternative machines and multiple-operation-type machines. *Comput. Ind. Eng.* **2021**, *154*, 107172. [CrossRef]
60. Suzic, N.; Stevanov, B.; Cosic, I.; Anisic, Z.; Sremcev, N. Customizing Products through Application of Group Technology: A Case Study of Furniture Manufacturing. *J. Mech. Eng.* **2012**, *58*, 724–731. [CrossRef]
61. Kumar, R.; Singh, S.P. A similarity score-based two-phase heuristic approach to solve the dynamic cellular facility layout for manufacturing systems. *Eng. Optim.* **2017**, *49*, 1848–1867. [CrossRef]
62. Feng, Y.L.; Li, G.; Sethi, S.P. A three-layer chromosome genetic algorithm for multi-cell scheduling with flexible routes and machine sharing. *Int. J. Prod. Econ.* **2018**, *196*, 269–283. [CrossRef]
63. Imran, M.; Kang, C.; Lee, Y.H.; Jahanzaib, M.; Aziz, H. Cell formation in a cellular manufacturing system using simulation integrated hybrid genetic algorithm. *Comput. Ind. Eng.* **2017**, *105*, 123–135. [CrossRef]
64. Zeidi, J.R.; Javadian, N.; Tavakkoli-Moghaddam, R.; Jolai, F. A hybrid multi-objective approach based on the genetic algorithm and neural network to design an incremental cellular manufacturing system. *Comput. Ind. Eng.* **2013**, *66*, 1004–1014. [CrossRef]
65. Chandrasekharan, M.P.; Rajagopalan, R. MODROC: An extension of rank order clustering for group technology. *Int. J. Prod. Res.* **1986**, *24*, 1221–1233. [CrossRef]
66. Boe, W.J.; Cheng, C.H. A close neighbour algorithm for designing cellular manufacturing systems. *Int. J. Prod. Res.* **1991**, *29*, 2097–2116. [CrossRef]