Hindawi
Mathematical Problems in Engineering
Volume 2018, Article ID 9168014, 12 pages
https://doi.org/10.1155/2018/9168014

Research Article

Genetic-Based Optimization of the Manufacturing Process of a Robotic Arm under Fuzziness

Paraskevi T. Zacharia, Sotirios A. Tsirkas, Georgios Kabouridis, Andreas Ch. Yiannopoulos, and Georgios I. Giannopoulos

Department of Mechanical Engineering, Technological Educational Institute of Western Greece, Megalou Alexandrou 1, 26334 Patras, Greece

Correspondence should be addressed to Georgios I. Giannopoulos; ggiannopoulos@teiwest.gr

Received 21 September 2018; Revised 16 November 2018; Accepted 9 December 2018; Published 24 December 2018

Guest Editor: Anders E. W. Jarfors

Fuzziness is a key concern in modern industry and, thus, its implementation in manufacturing process modeling is of high practical importance for a wide industrial audience. The scientific contribution of the present attempt lies on the fact that the assembly line balancing problem of type 2 (SALBP-2) is approached for a real manufacturing process by introducing fuzzy processing times. The main scope of this work is the solution of the SALBP-2, which is an NP-hard problem, for a real manufacturing process considering fuzziness in the processing times. Since the data obtained from realistic situations are imprecise and uncertain, the consideration of fuzziness for the solution of SALBP-2 is of great interest. Thus, real data values for the processing times are gathered and estimated with uncertainty. Then, fuzzy processing times are used for finding the optimum cycle time. The optimization tool for the solution of the fuzzy SALBP-2 is a Genetic Algorithm (GA). The validity of the proposed approach is tested on the construction process of a metallic robotic arm. The experimental results demonstrate the effectiveness and efficiency of the proposed GA in determining the optimum sequence of the tasks assigned to workstations which provides the optimum fuzzy cycle time.

1. Introduction

In mass production systems, an important problem is the assembly line problem. An assembly line is a manufacturing technique according to which parts are added in sequence from workstation to workstation until the final assembly is produced. Each station has to complete a set of tasks on parts moving along the line. There are numerous studies related with assembly line systems which focus on the determination of the set of tasks which have to be assigned to each workstation under a given cycle and the constraints of precedence relationships. This kind of problem is well known as the simple assembly line balancing problem (SALBP) [1].

The most famous versions of the abovementioned problem are the SALBP type 1 (SALBP-1) and the SALBP type 2 (SALBP-2). SALBP-1 [2–15] is present when the aim is to effectively assign tasks to workstations by minimizing the number of stations for a prespecified cycle time. This problem commonly arises when new assembly lines are to be designed by a company. On the other hand, SALBP-2 [16–22] is present when the aim is to minimize the cycle time for a specific number of stations. This kind of problem usually arises when changes in the production process of a product are to take place in the effort to improve the line efficiency. Several methods have been proposed for the solution of SALBP problems such as genetic algorithms (GAs) [2, 5, 17], ant colony optimization [5, 9, 22], particle swarm optimization [10, 11, 14], Petri net [4, 10, 18], tabu search [3], bacterial foraging optimization [15], or other heuristic algorithms [8, 12, 13, 19–21]. Battaïa and Dolgui [23] have provided a very interesting survey, covering about 300 studies, which analyzes relevant research on balancing flow lines within many different industrial contexts. A more recent review regarding the assembly line balancing problem (ALBP) has been provided by Sivasankaran and Shahabudeen [24].

In a realistic manufacturing environment, the task time may be random due to worker fatigue, low skill levels, job dissatisfaction, poorly maintained equipment, defects in raw materials, etc. Since data in real-world problems are often afflicted with uncertainty, imprecision, and vagueness due

to both machine and human factors, they can only be estimated with uncertainty. Several researchers [25] have been attempted to incorporate fuzzy information in their effort to solve SALBP through various types of algorithms. Kalender et al. [26] have been developed to solve traditional ALBP with fuzzy operation times. Ozcan and Toklu [27] have presented a fuzzy goal programming model for imprecise goals for two-sided assembly line balancing. Tapkan et al. [28] have solved two-sided ALPB by employing positional, zoning, and synchronous task constraints via a bees algorithm. Mutlu and Ozgormus [29] have considered the physical workload of a task as a fuzzy concept and proposed a fuzzy linear programming model to solve ALPB. La Scalia et al. [30] have used fuzzy set theory as a viable alternative method for modelling and solving the stochastic ALPB. In several attempts, GAs have been adopted to solve SALBP in conjunction with fuzzy logic. Tsujimura et al. [31] have illustrated via a numerical experiment that a GA is an appropriate tool to solve fuzzy scheduling problems. In another attempt, Gen et al. [32] have used a numerical example to solve ALPB with fuzzy processing time by using GAs with the objective of minimizing the total operation time in each workstation. Similarly, a GA based approach for both types of fuzzy ALPB has been presented by Khoshalhan and Zegordi [33]. Zacharia and Nearchou [34] have presented a fuzzy extension of the SALBP-2 with fuzzy job processing times to deal with the uncertainty occurring in production systems. In an interesting study [35], ALBP has been modeled through a multicriteria fuzzy GA.

Few are the very recent attempts regarding line balancing problems which incorporate fuzzy considerations. Characteristically, Alavidoost et al. [36] have formulated multiobjective straight and U-shaped ALBPs with the fuzzy task processing times. In another attempt, a new approach based on queuing theory has been demonstrated by Khalili et al. [37] for solving the ALBP using fuzzy prioritization techniques. On the other hand, a heuristic has been proposed by Avikal et al. [38] to assign the disassembly, though, tasks to the workstations under its precedence constraints incorporating fuzzy analytic hierarchy process. Nevertheless, the latest attention-grabbing studies associated with the ALBP commonly do not utilize fuzzy concepts. In an interesting effort, Su et al. [39] have conducted study on balancing a mixed-model assembly line of Type E based on a Petri net model. More recently, new lower bounds for the SALBP have been introduced and empirically evaluated [40]. Additionally, a mixed integer programming model has been developed for the parallel two-sided ALBP [41]. Furthermore, an innovative parallel assembly line configuration has been introduced for U-shaped lines [42]. Finally, Roshani and Giglio [43] have addressed the multimanned ALPB, with the objective of minimizing the cycle time.

In the present paper, the SALBP-2 regarding the construction of a real robotic arm is under investigation. The metal parts of the robotic arm are manufactured in four machining workstations. In order to deal with the variability of the task operation times, fuzzy set theory [44] is adopted as a very promising approach for modeling and solving stochastic problems. The fuzzy theory is then combined with an appropriate GA [45, 46] for solving the fuzzy SALBP-2 of the robot's metal frame. To handle more realistically the manufacturing of the robot's metal frame, the processing time for each job is considered as fuzzy and is represented by triangular fuzzy membership functions. In an attempt to treat relevant imprecise data, fuzzy numbers are introduced to represent the processing time of each job, where the membership function of a fuzzy data represents the grade of satisfaction of a decision maker. The main contribution of this paper is to enhance the real manufacturing process of a robotic arm in terms of time reduction, using an optimization algorithm (GA) by simultaneously considering the variability and ambiguity associated with real situations. To the authors' best knowledge, this is the first time that the SALBP-2 combined with GA and fuzziness is applied in the manufacturing process of a robotic structure.

The rest of the paper is organized as follows: Section 2 summarizes the arithmetics of fuzzy numbers, gives the background of the fuzzy assembly line balancing of type 2, and presents the practical problem of manufacturing the robot components. Section 3 presents the proposed optimization approach applied for the fuzzy ALBP of the robotic arm. Computational results concerning the performance of the GA are presented in Section 4, while conclusions and directions for future work are pointed out and discussed in Section 5.

2. The Fuzzy Assembly Line Balancing Problem

2.1. Arithmetic and Ranking Fuzzy Number. Compared to traditional binary sets (where variables may take on true or false values), fuzzy logic variables may have a truth value that ranges in degree between 0 and 1. Fuzzy logic has been extended to handle the concept of partial truth, where the truth value may range between completely true and completely false.

A fuzzy set is a class of objects with a continuum of grades of membership. Such a set is characterized by a membership (characteristic) function which assigns to each object a grade of membership ranging between zero and one. The membership function which represents a fuzzy set $\widetilde{A}$ is usually denoted by μ_A. For an element x, the value $\mu_A(A)$ is called the membership degree of x in the fuzzy set $\widetilde{A}$. The membership degree $\mu_A(A)$ quantifies the grade of membership of the element x to the fuzzy set $\widetilde{A}$. The value 0 means that x is not a member of the fuzzy set; the value 1 means that x is fully a member of the fuzzy set. The values between 0 and 1 characterize fuzzy members, which belong to the fuzzy set only partially.

Due to the nature of processing times, the most commonly used fuzzy sets in depicting these values are triangular fuzzy numbers (TFNs) [47] that establish extreme points to represent the most likely and least likely values for the individual variables. In this work, the time variables will be represented as TFNs. TFN $\widetilde{A}$ is denoted as a triplet of points, i.e., $\widetilde{A} = (\alpha_1, \alpha_2, \alpha_3)$, where $\alpha_1 < \alpha_2 < \alpha_3$. In the adapted fuzzy heuristics, the tasks' fuzzy processing times are accumulated using the fuzzy addition operator. In particular, by assuming

a second TFN $\widetilde{B} = (\beta_1, \beta_2, \beta_3)$, where $\beta_1 < \beta_2 < \beta_3$, then the following arithmetics between $\widetilde{A}$ and $\widetilde{B}$ may be performed [48]:

$$\begin{aligned} \widetilde{A} + \widetilde{B} &= (\alpha_1 + \beta_1, \alpha_2 + \beta_2, \alpha_3 + \beta_3) \\ \widetilde{A} - \widetilde{B} &= (\alpha_1 - \beta_3, \alpha_2 - \beta_2, \alpha_3 - \beta_1) \\ \widetilde{A} \times \widetilde{B} &= (\alpha_1 \cdot \beta_1, \alpha_2 \cdot \beta_2, \alpha_3 \cdot \beta_3) \\ \frac{\widetilde{A}}{\widetilde{B}} &= \left(\frac{\alpha_1}{\beta_3}, \frac{\alpha_2}{\beta_2}, \frac{\alpha_3}{\beta_1}\right) \end{aligned} \tag{1}$$

To compare the fuzzy numbers, some criteria to rank the fuzzy sets should be presented. The ranking method in this work involves three ordered criteria [49] which are explained in the following.

The greatest associate ordinary number F_1 is used as a first criterion for ranking:

$$F_1\left(\widetilde{A}\right) = \frac{\alpha_1 + 2\alpha_2 + \alpha_3}{4} \tag{2}$$

If F_1 does not rank the fuzzy numbers, then those which have the best maximum presumption F_2 (the mode) will be chosen:

$$F_2\left(\widetilde{A}\right) = \alpha_2 \tag{3}$$

If F_1 and F_2 do not rank the fuzzy numbers, then the divergence F_3 (the distance between two end-points) will be used as third criterion:

$$F_3\left(\widetilde{A}\right) = \alpha_3 - \alpha_1 \tag{4}$$

Consider a set Q composed of the TFNs $\widetilde{A}_i$ with $i = 1, 2, \ldots, n$. A TFN is called major and denoted as $\widetilde{A}^*$ when dominates all the others in some criterion, in Q, that is, $\widetilde{A}^* = \max Q$ (the operator max is the discrete maximum). The decision maker chooses some criteria and determines its order of dominance. If the first criterion can not determine the major TFN, then the second criterion follows and so on. On the contrary, a TFN is called minor when dominated by all others in Q and this operation is represented as min.

2.2. Fundamentals of Fuzzy Assembly Line Balancing of Type 2. The fuzzy SALBP can be stated as follows: m workstations are arranged along an assembly line. Manufacturing a single product on the line requires the partitioning of the total work into a set $V = \{1, \ldots, n\}$ of n elementary operations called tasks. Each task j is performed on exactly one station and requires a fuzzy processing time $\widetilde{t}_j$. Let $S_z (z = 1, \ldots, m)$ be the station load of station z (i.e., the set of tasks assigned to z), with a cumulated fuzzy task time $\widetilde{tS_z} = \sum_{j \in S_z} \widetilde{t}_j$ $(z = 1, \ldots, m)$. The tasks are partially ordered by precedence relations defining a directed acyclic graph (DAG) $G = (V, E)$ with V being the set of the nodes denoting the tasks in V and E being the set of the edges representing the precedence constraints among these tasks. The assembly line is associated with a fuzzy cycle time $\widetilde{c}$ denoting the maximum processing time available for each station. The aim of SALBP-2 is the minimization of the fuzzy cycle time $\widetilde{c}$ (i.e., the maximization of the production rate) given the number of workstations m.

3. The Proposed Optimization Approach

GAs [45, 46] are optimization techniques which simulate the natural selection mechanism observed in the biological evolution process. A GA has global and parallel search capability which is suitable for solving demanding problems of high nonlinearity. A GA, in contrast with common search techniques, starts with an initial set of random solutions called population (individuals) which satisfy the constraints of the problem. Chromosome is called each individual in the population representing a solution to the problem at hand. Each chromosome comprises a number of structures known as genes. The chromosomes are then regressed via iterations which are commonly known as generations. During each iterative procedure, i.e., generation, the chromosomes are estimated by utilizing some measures of fitness. This means that, in every generation, the fitness of every individual in the population is calculated. Then the more fit individuals are selected from the current population, and each individual's genome is changed to form the population of the next generation. This new population is then used in the next iteration of the algorithm. The next generation is created according to the fitness values by forming new chromosomes. These chromosomes arise by merging two chromosomes from the current generation by using a crossover operator or by changing a chromosome via utilizing a mutation operator. The complete set of chromosomes is called genotype, and the resulting organism is called phenotype. Generally, the GA after numerous generations converges to the best chromosome, which represents the ideal solution to the problem. A typical GA includes a genetic representation of the solution domain and an efficient fitness function to evaluate the solution domain.

The proposed GA according to the requirements of the present study has the following basic components: (a) the representation mechanism which is a method to transform phenotypes into genotypes, (b) the decoding mechanism which is a method to map phenotypes to solutions, (c) the evaluation mechanism which is a method to compute the cost-function for each genotype, (d) the mechanism which generates the initial population of the genotypes, (e) the mechanism which generates new genotypes by applying operators on the entire population, (f) the control parameters, and (g) the termination condition. The block diagram of the present GA is illustrated in Figure 1 while its main steps are explained with more detail in the following subsections.

3.1. The Representation Mechanism. A GA can only find possible solutions to a problem when the solutions are transformed into a representation which the GA may handle. Thus, an appropriate representation mechanism is required

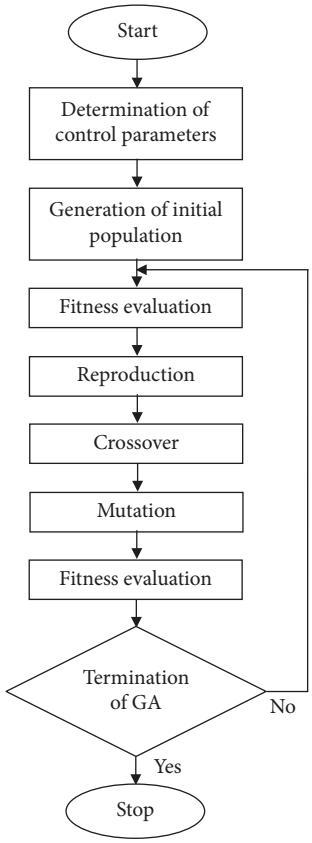

Figure 1: The block diagram of the proposed GA.

to transform possible solutions within the context of the original problem, called phenotypes, into individuals within the context of the GA, called genotypes. This encoding may be realized via a string of binary code, real-valued numbers, integers, or a tree structure. Nevertheless, for mechanical engineering problems the most efficient representation is a string of real-valued numbers. Thus, in the present study a real-valued GA has been adopted in which genotypes are represented by floating-point vectors. Since ALBP solutions are represented by strings of integers [1], the utilized representation mechanism should allow mapping from the genotypic state-level (the real-valued vectors) to the phenotypic level (the actual ALB solutions). This is realized by a simplified, however, efficient topological ordering scheme which is based on the relative priorities imposed by the components of a genotype. Assuming the n task of the ALBP with precedence relations given by a DAG $G = (V, E)$, the utilized representation mechanism aims to generate a topological sort of G from a specific n-dimensional floating-point vector ψ (genotype). Each vector's component ψ_i ($i = 1, 2, \ldots, n$) represents the relative priority of task i ($i \in V$). The topological sort is therefore a ranking of all the tasks in line with their priorities and precedence constraints. During the procedure, the tasks with no predecessors are identified and put in set V'. Then, the task in V' which has the highest gene's value in ψ is removed from V' and placed in the next available position of a new string (initially empty) called partial schedule (PS). The process is continued until the completion of PS is achieved.

3.2. The Decoding Mechanism. After encoding a specific genotype into a phenotype, the decoding mechanism is required to assign the tasks in the generated task-sequence into the stations. The proposed technique [49] seems to be more effective in contrast with other traditional schemes [1] and therefore preferred in the proposed GA. The utilized scheme consists of the following steps:

Step 1. Set $\widetilde{c}$ equal to the theoretical minimum fuzzy cycle time; i.e., $\widetilde{c}_{\text{th}} = \widetilde{t}_{\text{sum}}/m$.

Step 2. Assign as many tasks as possible into the first $m - 1$ workstations. Assign all the remaining tasks to the last workstation m.

Step 3. Calculate the fuzzy work load $\widetilde{\text{W}}_z$ for each workstation z ($z = 1, \ldots, m$) and the potential fuzzy workload $\widetilde{\text{PW}}_z$ ($z = 1, \ldots, m - 1$), where $\widetilde{\text{PW}}_z$ is the sum of $\widetilde{t}\text{S}_z$ and the processing time of the first task assigned to $(z + 1)$-th station ($z = 1, \ldots, m - 1$).

Step 4. Set $\widetilde{c}_{\text{W}} = \max\{\widetilde{\text{W}}_1, \widetilde{\text{W}}_2, \ldots, \widetilde{\text{W}}_{\text{m}}\}$ and $\widetilde{c} = \max\{\widetilde{\text{PW}}_1, \widetilde{\text{PW}}_2, \ldots, \widetilde{\text{PW}}_{\text{m}-1}\}$.

Step 5. If $\widetilde{c}_{\text{W}} > \widetilde{c}$, then go to Step 2; else return $\widetilde{c}_{\text{W}}$.

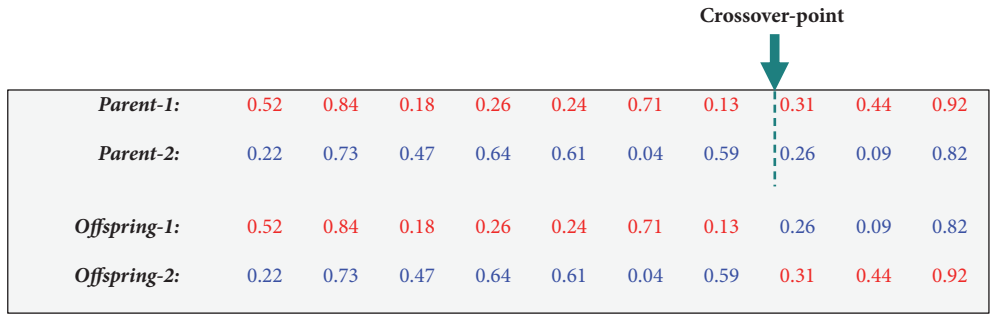

Figure 2: Example of one-point crossover: choose at random a cut-point and then exchange the genetic material between the parents.

Figure 3: Example of mutation operator: choose at random a gene and then replace it with another floating number randomly selected within U(0,1).

3.3. The Evaluation Mechanism. The evaluation mechanism corresponds to the computation of the objective function $\tilde{c}$ for each phenotype of the current population. The objective of SALBP-2 is to minimize the fuzzy cycle time $\tilde{c}$. The objective function has to be transformed to the fitness function f, which is evaluated for all the chromosomes of the population. The value of the fitness function for one chromosome expresses its ability to survive and be reproduced in the next generation. The fitness function is the inverse of the objective function:

$$f = \frac{1}{\tilde{c}} \tag{5}$$

Equation (5) aims to maximize the fitness function and consequently forcing $\tilde{c}$ to a minimum value. For the evaluation of the solution candidates, (2) is applied to compare the fuzzy numbers.

3.4. The Initial Population. In order to initialize the process, the solutions in the first generation have to be defined. Commonly, this is done randomly since the main desire is to spread the first individuals over the complete search space before converging to more promising regions. Nevertheless, when the area of the optimum solution may be estimated beforehand, then the algorithm could be initiated around this area to speed up the convergence.

3.5. The Genetic Operators. A genetic operator is used in GAs to maintain the genetic diversity. Genetic operators such as crossover, mutation, and selection are used in GAs to assure genetic variation for the process of evolution. Roulette wheel selection [50] is the best known selection operator and thus adopted here. The concept is to evaluate selection probability for each chromosome proportionally to the fitness value. Then a model roulette wheel is made which displays these probabilities. The selection process is based on spinning the specific wheel the number of times equal to the population size. Crossover is an operator which aims to produce new individuals by joining parts of several individuals of the previous generation. The selection of individuals is made randomly according to a predefined probability, i.e., one-point crossover rate [50]. Mutation is required to insert new genetic material into the population by slightly modifying the genotype representation. In this way, the early convergence to local minima is avoided. The modification is applied stochastically to discover potential better solutions based on the current best solutions. The mutation operator is applied by changing a random gene (i.e., a floating number) according to a small-predefined probability (mutation rate) [50]. Figures 2 and 3 demonstrate two application examples of these operators on random selected chromosomes. For the sake of simplicity, the presented chromosomes have a length of 10, although in our problem the length of the chromosomes is 57.

3.6. Control Parameters. The most important control parameters are the population size, the crossover ratio, the mutation rate, and the elitism.

Population size is one of the most important topics in evolutionary computation since small population size may lead the algorithm to poor solutions while a large population size may require a much more computational time to find a solution. The population size selection should be made according to the nature and the complexity of the current problem. In the present work, a variety of tests have been made considering the influence of the population size on the convergence of the algorithm. The algorithm has been run several times starting with different population sizes and the

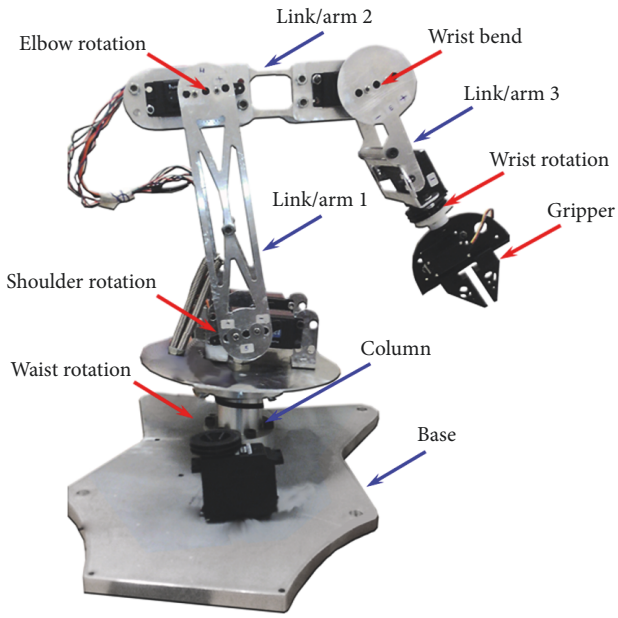

Figure 4: Robotic arm description.

convergence has been evaluated. Finally, the population size has been set to 100 for the SALBP-2 under investigation which involves 4 workstations and 57 tasks.

The crossover rate controls the frequency with which the crossover is applied. The higher crossover rate is, the more quickly new individuals are added to the population. Several convergence tests have proved the effectiveness of a crossover rate equal to 0.8 for the purpose of the present research.

The mutation rate defines the probability according to which the position of each individual in the intermediate population undergoes a random change. A GA with a too high mutation rate will inevitably become a random search. Thus, the mutation rate is typically chosen to be less than 0.4. The value of 0.1 has been chosen here after a considerable number of trials.

When creating new population via crossover and mutation, there is a chance that the best chromosome will be lost. Elitism [50] is a method according to which the best chromosome is copied to the new population. A comparison involving the best chromosome between current and previous generation is required. Elitism increases the performance of the GA since it prevents the loss of the best-found solution.

3.7. Termination Conditions. The termination condition should be theoretically satisfied and thus end the algorithm when the optimum solution has been found. However, for many optimization problems, the ideal solution is unknown and therefore there is always an uncertainty whether better solutions exist. In addition stochastic procedures may require a significant computational cost and take a long time to converge to the optimum solution. Commonly used termination conditions are therefore based on the maximum allowed number of iterations (generations) which however present limitations since it is difficult to determine beforehand the number of generations needed to find near-optimum solutions. Thus, in the proposed scheme, the termination of the GA is chosen to occur when the repetitions of the same solution have reached a predefined number. Consequently, the algorithm terminates when a specific chromosome has appeared for a sufficiently large number of times.

4. Description of Assembly under Investigation

The robotic arm, which is manufactured for the requirements of the present study, is a 6-degree-of-freedom manipulator. It is mainly constructed from a 3 mm aluminum sheet, has a length of about 600 mm, and has the ability to pick up a small object of 0.3 kg. In order to keep the design as simple as possible, the pivoting parts include six standard step motors. Additionally, the robotic arm under investigation incorporates a controller which governs the arm movements and operations. A simple, scalable control system has been adopted to allow control in a Cartesian coordinate system, which offers expandability for future research. The idea behind the specific design is to investigate the potential of developing a simple robotic arm capable of moving small objects. The skeleton of the robotic arm is constructed by using exclusively aluminum bars and sheets, whereas basic conventional machines are used during the manufacturing process.

Figure 4 depicts the manufactured robotic arm consisting of a waist, a shoulder, an elbow, a wrist, and a gripper which are connected to each other via metal links. These links are appropriately designed and machined so that they may offer stability, smooth motion, and, if required, effective support for specific motors.

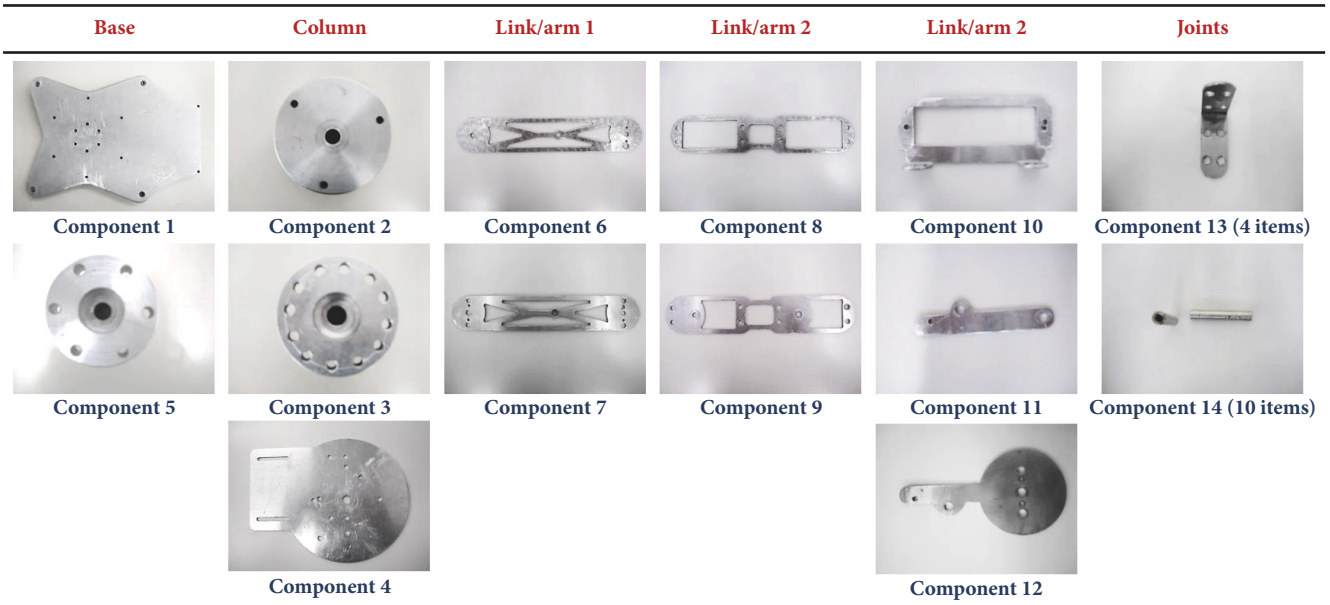

Figure 5: Numbered components of the robotic aluminum frame.

Figure 6: The four types of machines of the assembly line.

The mechanical design of the robotic arm includes a heavy bottom base of 20 mm thick aluminum. The aluminum base is designed in such way that it provides enough space for the more powerful motor which is responsible for the waist rotation. A column joined with a lighter upper base is manufactured to offer smooth shoulder rotation and appropriate motor support. The link/arm 1 connects the shoulder and elbow while the link/arm 2 connects the elbow with the wrist. The second link is formed in such a way to bracket both the elbow rotation and wrist bend motors. The wrist rotation motor is attached on link/arm 3 which connects the wrist bend with the wrist rotation mechanisms. The gripper is appropriately attached on the outer edge of this third link. All these robotic parts are coupled with appropriate cylindrical joints. The metal frame of the above described robotic arm is composed of several aluminum components which are illustrated and numbered in Figure 5.

The metal components of the robotic frame are machined using conventional processes in which appropriate pieces of raw material are progressively cut and/or formed into the desired final shape and size. The whole machining process takes place in a simple assembly line of four workstations. The assembly line under consideration includes four types of machining operations, i.e., milling, drilling, lathing, and bending. Conventional machines such as those depicted in Figure 6 are utilized for the purpose of the present study.

Table 1: Fuzzy execution times for each robot component.

No. of tasks	No. of components	Type of task	Total fuzzy time (s)
1	1	milling	(3096, 3105, 3111)
2	1	drilling	(73, 74, 76)
3	2	milling	(788, 792, 795)
4	2	drilling	(51, 53, 55)
5	3	milling	(822, 826, 831)
6	3	drilling	(40, 41, 42)
7	4	milling	(858, 863, 868)
8	4	drilling	(66, 67, 69)
9	5	milling	(518, 523, 527)
10	5	drilling	(44, 45, 46)
11	6	milling	(842, 849, 854)
12	6	drilling	(42, 43, 45)
13	7	milling	(1043, 1052, 1058)
14	7	drilling	(17, 18, 19)
15	8	milling	(780, 785, 789)
16	8	drilling	(57, 59, 61)
17	9	milling	(803, 806, 810)
18	9	drilling	(54, 55, 57)
19	10	milling	(540, 544, 549)
20	10	drilling	(24, 25, 26)
21	10	bending	(40, 41, 42)
22	11	milling	(390, 396, 401)
23	11	drilling	(25, 26, 27)
24	12	milling	(381, 385, 390)
25	12	drilling	(40, 41, 42)
26	13 (1 out of 4 items)	milling	(947, 951, 956)
27	13	drilling	(248, 252, 255)
28	13	bending	(39, 41, 42)
29-37	13 (3 items)		
38	14 (1 out of 10 items)	lathing	(1560, 1569, 1576)
39	14	drilling	(63, 65, 66)
40-57	14 (9 items)		

5. Computational Study

According to Figure 5, in order to construct the metal structure of the robotic arm, 14 different types of components should be manufactured using the aforementioned machines that do not work in parallel. To be more precise, a total of number of 26 parts should be machined since the robotic arm design requires 4 and 10 items of components 13 and 14, respectively. All experiments have taken place in the laboratory of Computer Numerical Control Machines (CNC Lab) in the Mechanical Engineering Department of the Technological Educational Institute (TEI) of Western Greece, using milling, lathe, bending, and drilling machines. A number of 10 students (representing 10 "workers" with different capabilities) have participated actively in the experiments. Table 1 summarizes the fuzzy execution times for the 57 total tasks associated with the robot components. The computational study is focused on finding the optimum sequence of tasks assigned to stations which may lead to the minimum total fuzzy time for executing all tasks.

Considering fuzziness for the processing times, fuzzy data are represented by triangular fuzzy numbers. TFN is one of the most commonly used in the literature shapes of fuzzy numbers representation [47] composed of three estimates (the lowest expected, the most likely, and the highest expected) of the unknown individual value. The reason of using triangular fuzzy shapes is because of their computational simplicity in comparison with other fuzzy shapes, considering the calculations in (1). TFNs differ from statistical distributions in the fact that they do not require historical data to establish their values. This is a major advantage of using TFNs as opposed to statistics.

In practice, this was achieved by assigning the 57 total tasks to 10 "workers" and writing down the resulting times.

For all tasks, the most likely values for the TFNs (i.e., the fuzzy element with the membership value of 1) are considered to be the middle written times at the last column of Table 1. These most likely values for each task were set equal to the average of the corresponding ten measured times by the ten "workers". The extreme TFN values in Table 1 were taken equal to the minimum and maximum times of the 10 execution times provided by the "workers", respectively. In a fuzzy set $\widetilde{A} = (\alpha_1, \alpha_2, \alpha_3)$, the minimum time represents the optimistic value α_1, the maximum time represents the pessimistic value α_3, and the middle time represents the most plausible value α_2. Although, in practice, the actual time is more possible to be longer and not shorter than α_2, we decided to set the most likely value in the middle of the two extreme values in order to keep "equal distances" between the optimistic and pessimistic value. Generally speaking, a shifted triangular fuzzy number for modeling the uncertainty would be more realistic; however, in our problem, the variations in times were rather small giving us the opportunity for the above admittance.

The precedence constraints are summarized in Table 2. For each row, the task in the first column precedes the task in the second column and the task in the second column precedes the task in the third column.

The minimum fuzzy cycle time is yielded considering the total fuzzy processing time for each of the 57 tasks (see Table 1) as well as the precedence relations given above. The resulting minimum fuzzy cycle time after running the proposed GA is (8150, 8211, 8259) s. Concerning the station load for each workstation, it is provided by the GA best solution, considering the aforementioned constraints, and involves the task sequence assigned to each workstation. In particular, Workstation 1 starts with task 53 and after finishing, it continues with task 24 and so on until it finishes with task 3. Similarly, Workstation 2 starts with task 22, then accomplishes task 38, and finishes with task 10. The robotic arm is manufactured when all tasks assigned to each workstation have been accomplished. For the problem addressed here, the resulting task sequence and the fuzzy station time for each workstation are presented in Table 3. The simulation test was implemented in Matlab and run on a Pentium IV 2.13 GHz core2 PC and the CPU time was 993 sec. Given the nature and the complexity of the present problem, a rather insignificant computational effort was required. However, it has to be mentioned that the selection of a larger population size would significantly increase the CPU time. Auspiciously, various numerical tests on the population size showed that the selection of a population size larger than 100 had a negligible effect on the convergence of the method.

Figure 7 depicts the triangular membership function for the fuzzy cycle time. It is clear that the least likely values are 8150 and 8259 while it is found that the most likely value is 8211.

To compare the fuzzy numbers, the greatest associate ordinary number (see (2)) is used. Thus, the value for the cycle time presented in the following has been calculated through (2), and Figures 8, 9, and 10 illustrate the evolution of the GA through the generations for the best, average, and worst individuals, respectively. It is clear from Figure 8 that the best cycle time (equal to 8207.75 s corresponding to (8150, 8211, 8259) s) is achieved in the 1459th generation.

Table 2: The precedence constraints.

No. of tasks		
1	2	
3	4	
5	6	
7	8	
9	10	
11	12	
13	14	
15	16	
17	18	
19	20	21
22	23	
24	25	
26	27	28
29	30	31
32	33	34
35	36	37
38	39	
40	41	
42	43	
44	45	
46	47	
48	49	
50	51	
52	53	
54	55	
56	57	

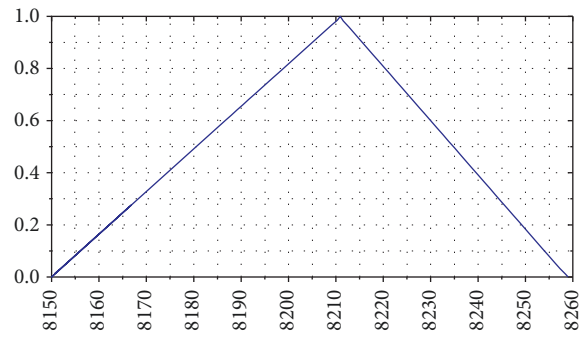

Figure 7: The fuzzy cycle time $\tilde{c}$.

6. Conclusions

Due to the fact that the nature of manufacturing systems is accompanied with uncertainty, the main idea for this paper is to treat the problem considering fuzziness in the operation times. Thus, the main focus of this work lies on the

Table 3: Results provided by the proposed GA.

Workstation z	**Task sequence** (S_z)	**Fuzzy station time** ($\widetilde{tS}_z$) (s)
1	52-53-24-40-35-17-5-6-29-30-3	(8159, 8207, 8253)
2	22-38-39-31-4-25-15-16-11-19-20-21-23-12-1-9-10	(8151, 8210, 8259)
3	18-42-43-48-49-56-57-32-33-34-50-51-2-36-37	(8140, 8202, 8251)
4	26-27-54-55-44-45-28-13-14-46-47-7-8-41	(8150, 8211, 8259)
Fuzzy cycle $\widetilde{c}$ (s)	(8150, 8211, 8259)	

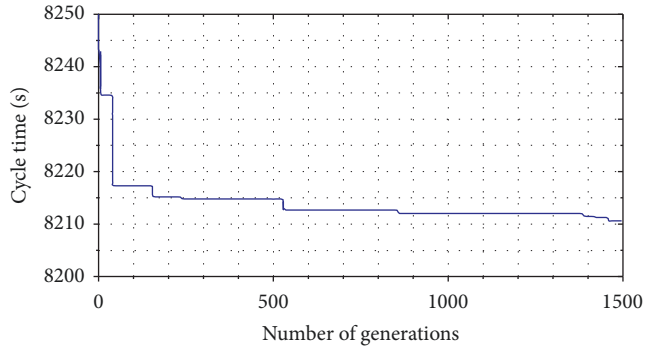

Figure 8: Best cycle time versus generations.

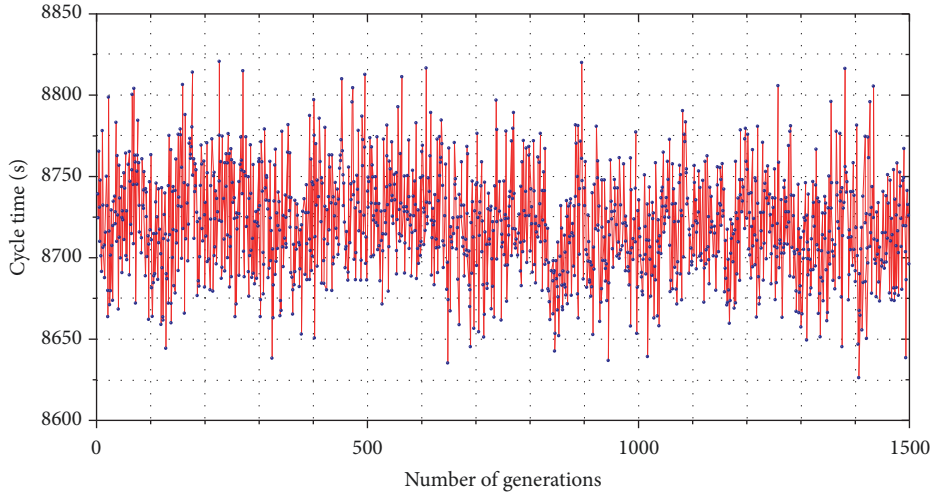

Figure 9: Average cycle time versus generations.

construction process considering variability and ambiguity associated with real situations. In this context, this paper studies the solution of the fuzzy ALBP problem type 2 for the real problem of constructing a robotic arm. The robot components are formed using a number of machine tools, which are handled by a specific number of "workers". The metal parts of the robotic arm are manufactured in four machining workstations. The construction process is enhanced in terms of time reduction using a GA that takes into account fuzziness in times and is subject to the constraints imposed by the precedence relations.

The proposed approach was tested in a real manufacturing environment, where real data was yielded for the simulation tests. The experimental results demonstrated that the approach is effective and efficient at determining the optimum fuzzy cycle time for the accomplishment of robotic arm construction without violating the precedence constraints. Considering future research, a U-shaped assembly line layout could be applied, where stations work at two segments of the line facing each other simultaneously.

Data Availability

All the data (numerical, experimental, figures, diagrams, tables, etc.) used to support the findings of our study are included within the article. Thus, data sharing regarding the aforementioned paper is totally allowed and any reader can access the data supporting the conclusions of the study.

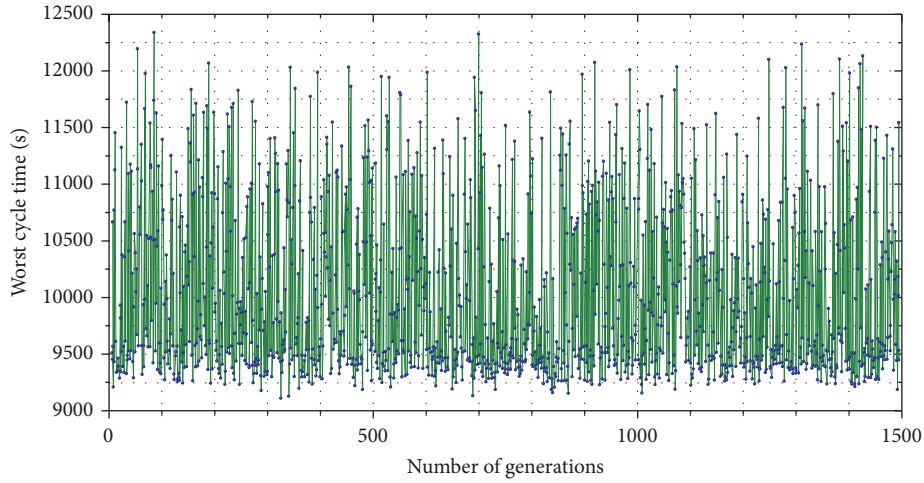

Figure 10: Worst cycle time versus generations.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research has been cofinanced by the European Union (European Social Fund, ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF), Research Funding Program: ARCHIMEDES III. Investing in Knowledge Society through the European Social Fund.

References

[1] A. Scholl and C. Becker, "State-of-the-art exact and heuristic solution procedures for simple assembly line balancing," *European Journal of Operational Research*, vol. 168, no. 3, pp. 666–693, 2006.

[2] J. F. Gonçalves and J. R. De Almeida, "A hybrid genetic algorithm for assembly line balancing," *Journal of Heuristics*, vol. 8, no. 6, pp. 629–642, 2002.

[3] S. D. Lapierre, A. Ruiz, and P. Soriano, "Balancing assembly lines with tabu search," *European Journal of Operational Research*, vol. 168, no. 3, pp. 826–837, 2006.

[4] O. Kilincci and G. M. Bayhan, "A Petri net approach for simple assembly line balancing problems," *The International Journal of Advanced Manufacturing Technology*, vol. 30, no. 11-12, pp. 1165–1173, 2006.

[5] Z.-Q. Zhang, W.-M. Cheng, L.-S. Tang, and B. Zhong, "Ant algorithm with summation rules for assembly line balancing problem," in *Proceedings of the International Conference on Management Science and Engineering, (ICMSE'07)*, pp. 369–374, China, 2007.

[6] A. C. Nearchou, "Multi-objective balancing of assembly lines by population heuristics," *International Journal of Production Research*, vol. 46, no. 8, pp. 2275–2297, 2008.

[7] O. Kilincci and G. M. Bayhan, "A P-invariant-based algorithm for simple assembly line balancing problem of type-1," *The International Journal of Advanced Manufacturing Technology*, vol. 37, no. 3-4, pp. 400–409, 2008.

[8] D.-H. Yeh and H.-H. Kao, "A new bidirectional heuristic for the assembly line balancing problem," *Computers & Industrial Engineering*, vol. 57, no. 4, pp. 1155–1160, 2009.

[9] M. N. I. Sulaiman, Y. H. Choo, and K. E. Chong, "Ant colony optimization with look forward ant in solving assembly line balancing problem," in *Proceedings of Conference on Data Mining and Optimization (3rd DMO 2011)*, pp. 115–121, 2011.

[10] J. Dou, J. Li, and Q. Lv, "A hybrid particle swarm algorithm for assembly line balancing problem of type 1," in *Proceedings of the IEEE International Conference on Mechatronics and Automation, (ICMA 2011)*, pp. 1664–1669, China, 2011.

[11] D. Jianping, S. Chun, and L. Jun, "A Discrete Particle Swarm Optimization Algorithm for Assembly Line Balancing Problem of Type 1," in *Proceedings of the International Conference on Measuring Technology and Mechatronics Automation (ICMTMA)*, vol. 1, pp. 44–47, Shanghai, China, 2011.

[12] M. Fathi, A. Jahan, M. K. A. Ariffin, and N. Ismail, "A new heuristic method based on CPM in SALBP," *Journal of Industrial Engineering International*, vol. 7, no. 13, pp. 1–11, 2011.

[13] M. K. A. Ariffin, M. Fathi, and N. Ismail, "A new heuristic method to solve straight assembly line balancing problem," *Pertanika Journal of Science & Technology*, vol. 20, no. 2, pp. 355–369, 2012.

[14] J. Dou, J. Li, and C. Su, "A novel feasible task sequence-oriented discrete particle swarm algorithm for simple assembly line balancing problem of type 1," *The International Journal of Advanced Manufacturing Technology*, vol. 69, no. 9-12, pp. 2445–2457, 2013.

[15] Y. Atasagun and Y. Kara, "Bacterial Foraging Optimization Algorithm for assembly line balancing," *Neural Computing and Applications*, vol. 25, no. 1, pp. 237–250, 2014.

[16] R. Zhang, D. Chen, Y. Wang, Z. Yang, and X. Wang, "Study on line balancing problem based on improved genetic algorithms," in *Proceedings of the International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM 2007*, pp. 2033–2036, 2007.

[17] L. Gu, S. Hennequin, A. Sava, and X. Xie, "Assembly Line Balancing Problems Solved by Estimation of Distribution," in *Proceedings of the IEEE International Conference on Automation*

Science and Engineering (IEEE CASE 2007), pp. 123–127, Scottsdale, AZ, USA, 2007.

[18] O. Kilincci, "A Petri net-based heuristic for simple assembly line balancing problem of type 2," *The International Journal of Advanced Manufacturing Technology*, vol. 46, no. 1-4, pp. 329–338, 2010.

[19] C. Blum, "Iterative beam search for simple assembly line balancing with a fixed number of work stations," *SORT. Statistics and Operations Research Transactions*, vol. 35, no. 2, pp. 145–164, 2011.

[20] Q. Tang, S. Lu, M. Li, and C. A. Floudas, "Novel cellular automata algorithm for assembly line balancing problem of type-2," in *Proceedings of the 6th International Conference on Pervasive Computing and Applications, (ICPCA 2011)*, pp. 422–428, South Africa, 2011.

[21] S. Avikal, P. K. Mishra, and R. Jain, "A model for assembly line balancing problems," in *Proceedings of the Students Conference on Engineering and Systems (SCES 2012)*, pp. 1–4, Allahabad, Uttar Pradesh, India, 2012.

[22] Q. Zheng, M. Li, Y. Li, and Q. Tang, "Station ant colony optimization for the type 2 assembly line balancing problem," *The International Journal of Advanced Manufacturing Technology*, vol. 66, no. 9-12, pp. 1859–1870, 2013.

[23] O. Battaïa and A. Dolgui, "A taxonomy of line balancing problems and their solutionapproaches," *International Journal of Production Economics*, vol. 142, no. 2, pp. 259–277, 2013.

[24] P. Sivasankaran and P. Shahabudeen, "Literature review of assembly line balancing problems," *The International Journal of Advanced Manufacturing Technology*, vol. 73, no. 9–12, pp. 1665–1694, 2014.

[25] A. Caggiano, "Fuzzy Logic," in *CIRP Encyclopedia of Production*, pp. 562–568, Springer, 2014.

[26] F. Y. Kalender, M. M. Yilmaz, and O. Türkbey, "A fuzzy approach to assembly line balancing problem," *Journal of the Faculty of Engineering and Architecture of Gazi University*, vol. 23, no. 1, pp. 129–138, 2008.

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

[28] P. Tapkan, L. Ozbakir, and A. Baykasoglu, "Bees Algorithm for constrained fuzzy multi-objective two-sided assembly line balancing problem," *Optimization Letters*, vol. 6, no. 6, pp. 1039–1049, 2012.

[29] Ö. Mutlu and E. Özgörmüş, "A fuzzy assembly line balancing problem with physical workload constraints," *International Journal of Production Research*, vol. 50, no. 18, pp. 5281–5291, 2012.

[30] G. La Scalia, R. Micale, G. Aiello, and M. Enea, "Solving type-2 assembly line balancing problem with fuzzy binary linear programming," *Journal of Intelligent & Fuzzy Systems. Applications in Engineering and Technology*, vol. 25, no. 3, pp. 517–524, 2013.

[31] Y. Tsujimura, M. Gen, and E. Kubota, "Solving fuzzy assembly-line balancing problem with genetic algorithms," *Computers & Industrial Engineering*, vol. 29, no. 1-4, pp. 543–547, 1995.

[32] M. Gen, Y. Tsujimura, and Y. Li, "Fuzzy assembly line balancing using genetic algorithms," *Computers & Industrial Engineering*, vol. 31, no. 3-4, pp. 631–634, 1996.

[33] F. Khoshalhan and S. H. Zegordi, "Solving type one and type two fuzzy assembly line balancing problems using genetic algorithms," *Amirkabir (Journal of Science and Technology)*, vol. 14, no. 55 D, pp. 910–923, 2003.

[34] P. T. Zacharia and A. C. Nearchou, "Multi-objective fuzzy assembly line balancing using genetic algorithms," *Journal of Intelligent Manufacturing*, vol. 23, no. 3, pp. 615–627, 2012.

[35] H. Rajabalipour Cheshmehgaz, H. Haron, F. Kazemipour, and M. I. Desa, "Accumulated risk of body postures in assembly line balancing problem and modeling through a multi-criteria fuzzy-genetic algorithm," *Computers & Industrial Engineering*, vol. 63, no. 2, pp. 503–512, 2012.

[36] M. H. Alavidoost, M. Tarimoradi, and M. H. F. Zarandi, "Fuzzy adaptive genetic algorithm for multi-objective assembly line balancing problems," *Applied Soft Computing*, vol. 34, pp. 655–677, 2015.

[37] S. Khalili, H. Mohammadzade, and M. S. Fallahnezhad, "A new approach based on queuing theory for solving the assembly line balancing problem using fuzzy prioritization techniques," *Scientia Iranica*, vol. 23, no. 1, pp. 387–398, 2016.

[38] S. Avikal, P. K. Mishra, and R. Jain, "A fuzzy AHP and PROMETHEE method-based heuristic for disassembly line balancing problems," *International Journal of Production Research*, vol. 52, no. 5, pp. 1306–1317, 2014.

[39] P. Su, N. Wu, and Z. Yu, "A Petri net-based heuristic for mixed-model assembly line balancing problem of Type-E," *International Journal of Production Research*, vol. 52, no. 5, pp. 1542–1556, 2014.

[40] J. Pereira, "Empirical evaluation of lower bounding methods for the simple assembly line balancing problem," *International Journal of Production Research*, vol. 53, no. 11, pp. 3327–3340, 2015.

[41] K. Ağpak and S. Zolfaghari, "Mathematical models for parallel two-sided assembly line balancing problems and extensions," *International Journal of Production Research*, vol. 53, no. 4, pp. 1242–1254, 2015.

[42] I. Kucukkoc and D. Z. Zhang, "Balancing of parallel U-shaped assembly lines," *Computers & Operations Research*, vol. 64, pp. 233–244, 2015.

[43] A. Roshani and D. Giglio, "Simulated annealing algorithms for the multi-manned assembly line balancing problem: minimising cycle time," *International Journal of Production Research*, vol. 55, no. 10, pp. 2731–2751, 2017.

[44] L. A. Zadeh, "Fuzzy sets," *Information and Computation*, vol. 8, no. 3, pp. 338–353, 1965.

[45] J. H. Holland, *Adaption in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, MI, USA, 1975.

[46] D. E. Goldberg, *Genetic algorithms in search, optimization, and machine learning*, vol. 27, Addison-Wesley, Reading, Mass, 1989.

[47] D. J. Fonseca, C. L. Guest, M. Elam, and C. L. Karr, "A fuzzy logic approach to assembly line balancing," *Mathware & Soft Computing*, vol. 12, pp. 57–74, 2005.

[48] A. Kaufmann and M. M. Gupta, *Introduction to Fuzzy Arithmetic*, Van Nostrand Reinhold Company, 1985.

[49] Y. K. Kim, Y. J. Kim, and Y. Kim, "Genetic algorithms for assembly line balancing with various objectives," *Computers & Industrial Engineering*, vol. 30, no. 3, pp. 397–409, 1996.

[50] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, Heidelberg, Germany, 3rd edition, 1996.