Hindawi Publishing Corporation
Advances in Decision Sciences
Volume 2014, Article ID 874031, 11 pages
http://dx.doi.org/10.1155/2014/874031

Research Article

Scheduling Mixed-Model Production on Multiple Assembly Lines with Shared Resources Using Genetic Algorithms: The Case Study of a Motorbike Company

Francesco Costantino,[1] Alberto Felice De Toni,[2] Giulio Di Gravio,[1] and Fabio Nonino[3]

[1] *Department of Mechanical and Aerospace Engineering, University of Rome "La Sapienza", Via Eudossiana 18, 00184 Rome, Italy*
[2] *Department of Electrical, Management and Mechanical Engineering, University of Udine, Via Delle Scienze 206, 33100 Udine, Italy*
[3] *Department of Computer, Control and Management Engineering "Antonio Ruberti", University of Rome "La Sapienza", Via Ariosto 25, 00186 Rome, Italy*

Correspondence should be addressed to Fabio Nonino; fabio.nonino@uniroma1.it

Received 27 May 2014; Accepted 15 September 2014; Published 2 October 2014

Academic Editor: Roger Z. Ríos-Mercado

The authors deal with the topic of the final assembly scheduling realized by the use of genetic algorithms (GAs). The objective of the research was to study in depth the use of GA for scheduling mixed-model assembly lines and to propose a model able to produce feasible solutions also according to the particular requirements of an important Italian motorbike company, as well as to capture the results of this change in terms of better operational performances. The "chessboard shifting" of work teams among the mixed-model assembly lines of the selected company makes the scheduling problem more complex. Therefore, a complex model for scheduling is required. We propose an application of the GAs in order to test their effectiveness to real scheduling problems. The high quality of the final assembly plans with high adherence to the delivery date, obtained in a short elaboration time, confirms that the choice was right and suggests the use of GAs in other complex manufacturing systems.

1. Introduction

Simulating the natural evolutionary process of human beings results in stochastic optimization techniques, called evolutionary algorithms, which can often outperform conventional methods when applied to complex real-world problems. Scheduling problems of manufacturing planning are a common example of complex problems where the interest in these groundbreaking techniques is growing among both scholars and practitioners.

The paper examines a case study of a scheduling system for a mixed-model assembly lines [1], also referred to as the permutation flowshop scheduling problem. In such a production system, the managers want to sequence different products, thus obtaining a high service level (product mix) without delays in products delivery while respecting the constraints of capacity. The focus of the present work is the application of the genetic algorithms (GAs) as a technique for the resolution of such a complex problem.

The paper is structured as follows. Section 2 depicts the principles and the successful characteristics of these techniques, providing an applicative model for their use in combinatory problems, in particular in the scheduling of final assembly phase of mixed-model assembly lines. We use the case study of an important Italian motorbike company as the outset of the study. Section 3 describes the company's production system and the demand management logics. Afterwards, in Section 4, we propose a scheduling approach consisting of two stages, a macrostep and a microstep. The two steps, respectively, carry out a "macroscheduling" and a "microscheduling" of the assembly operations; in particular, the genetic algorithms are applied in the microstep. We focus on the development of a scheduling model and provide the algorithms to implement in the scheduler. The paper ends with the comparisons of the results with five simple heuristics in terms of operational performances.

2. Genetic Algorithms in Scheduling Problems

There are three main research streams in the evolutionary algorithms: genetic algorithms (GAs), evolutionary programming (EP), and evolution strategies (ESs) [2]. While EP and ESs design algorithms to solve specific problems, GAs are highly customizable because they only require a performance measure, a representation of the problem, and the operators who generate the new population. Their main advantages are wide applicability and scalability to problems of any nature [3], representing a powerful and robust approach to develop heuristics for large-scale combinatorial optimisation problems [4].

The book *Adaptation in Natural and Artificial Systems* by Holland [5] describes the fundamental and basic principles of genetic algorithms; the author presents the GAs as an abstraction of biological evolution and provides a theoretical foundation for the concept of adaptation. The distinctiveness of this technique is that it draws inspiration from the natural evolution, founding on Darwinian principles of selection and adaptation and, naturally, on the reproduction and genetic mutation mechanisms.

According to Metaxiotis and Psarras [4], GAs are a strategic tool for many key business areas, like marketing, banking, finance, and forecasting, thus offering real benefits to the business decision support. In their review, Chaudry and Luo [6] highlight that GAs are commonly applied in two main areas of production and operation management, that is, scheduling and facility layout problems.

Due to the complexity of scheduling problems, in both practice and theory, heuristics are typical solution methodologies where GAs tend to highly perform [6, 7]. As an example of problem, GAs have been used in the scheduling problem in a two-machine flowshop [8] with a batch machine followed by a discrete machine in sequence, with the aim of scheduling the jobs to minimize the total completion time. Furthermore, Pongcharoen et al. [9] used GAs for scheduling complex products, thus achieving perfect time delivery and 63% of cost reduction in comparison to standard solutions. Many other authors (e.g., [10–13]) proved their applicability and usefulness in the job shop scheduling using performance criteria like the minimization of makespan (the completion time of the last job), or the weighted sum of different criteria, as for a multiobjectives minimization of makespan, of total idle time of machines, of total tardiness, and of total variation in parts consumption.

Considering also the computational results, GAs outperform heuristic algorithms for flowshop problems [14]. Murata et al. [15] applied the GAs in the flowshop scheduling problems with the aim of minimizing the makespan. Moreover, Murata et al. [16] proposed the GAs in the flowshop scheduling with the multiobjective function to minimize the makespan, the total tardiness, and the total flowtime (i.e., the sum of completion time over all jobs). Other authors state that the genetic algorithms proved to be slightly inferior to other search algorithms, just as local search, taboo search, and simulated annealing but the hybridizations of genetic algorithms showed high performances, as proved by computer simulation by Ruiz et al. [17]. By the way, the performance of the algorithm strictly depends on the core choice of the selection method [18, 19].

3. The Case Study: Complexity in the Management of Production Flows

Genetic algorithms can be used as the core of production scheduling algorithms because, as recent studies highlighted, the GAs outperform the other heuristics and metaheuristics in large combinatorial complex problems like the permutation flowshop problem [20]. Moreover, as highlighted by Leu et al. [1] and Celano et al. [21], the GAs select a solution based on the evaluation of a fitness function. Any function can be used. This provides great flexibility in capturing particular cost structures in mixed-model sequences and it is also possible to combine several different objectives in a composite evaluation function or to consider several objectives as ranked goals.

Taking off from these considerations, we aimed to investigate the effectiveness of GAs in a complex scheduling problem. Therefore, we selected an important Italian group in the industry of motorbikes and scooters as the outset of the research. The company's factory is in the north of Italy and assembles final products without any manufacturing process.

3.1. Demand Management. The market demand for two-wheel motor vehicles is strongly seasonal. Though the firm is a global company, sales are influenced by the seasonality of demand, which mainly comes from Europe and the United States. Purchases are higher in warmer months where demand is up to six times higher. Company's production managers prefer to follow demand fluctuations rather than keep production volumes constant on an average level throughout the year. Therefore, the calendar year divides into two production periods: low and high season.

3.2. Assembly Operations. Different work teams assemble vehicles on single and mixed-model assembly lines. The production plant contains eight assembly lines: four lines to assemble the motorbikes and four lines to assemble the scooters. The plant can produce up to 2.500 motorcycles every day, that is, one motorbike every five minutes and one scooter every two minutes.

A determined number of stations compose the assembly lines, made of plate conveyers where the semifinished products move forward on a vice. We can talk about stop and go lines: the product stops in front of the operator for the assembly time and then moves among the stations without any operation between two positions. The assembly process of the motorcycles develops along the line: it starts upstream by fixing the engine to a vice and then, as the semifinished product proceeds, the operators add the other components until the final product. Each line is committed to a work team; the number and composition of a team can vary according to seasonality: from March to September (warmer months), the number of operators can double.

3.3. Organization of Work Teams: The Chessboard Shifting. The management of workforce changes according to production volumes by increasing the number of work teams with seasonal workers; as a result, the work teams' efficiency is extremely variable. With high production volumes (high season) the number of teams is the same as the number of lines, so that each team works on a single line. On the contrary, with low production volumes (low season), the number of teams is less than that of the lines, and each team works on two or three lines.

Therefore, some assembly lines stay idle for more or less short periods. As a result, the work teams follow this framework:

(i) high season: the team works on a single assembly line;

(ii) low season: the team shifts among the assembly lines.

A main criticality lies in the evaluation of the efficiency curve of the work teams. Indeed, in the low season the work teams are composed by expert workers while in the high season there are also untrained seasonal workers. During seasonal changes, the expert personnel must support the inexpert one and must frequently stop the assembly line in order to help the new workers to learn their job. As a result, the team efficiency, evaluated on historical data, starts from low values, around 40%, and then increases with the rise of assembled vehicles, following the learning by doing principle.

This choice of workforce management is particularly critical also for another reason: in order to operate with less work teams than assembly lines, a shifting logic called "chessboard shifting" is necessary. The chessboard shifting consists of moving a work team from one line to another, as soon as the team finishes a production batch and starts to assemble a different model, on a different line. These displacements can be even more frequent, according to the resettlement of the final assembly schedule (FAS) and lead to a big loss of time.

Each line is linked to a sole work team, to specialize in relevant models, while each work team can be linked to one or more line. This method of workforce management is much complex as each line could be in one of the following situations.

(i) Full line: line is with all the vices occupied by semiassembled motorbikes.

(ii) Empty line: work team shifting among lines can result in four different situations, according to the state of the outcoming line (empty or full line) and the state of the incoming line (empty or full line), as shown in Figure 1. We analysed the four possible shifting situations of work teams in order to point out the inefficiencies of each case with two reasonable simplifications as follows:

(iii) production lines without buffer,

(iv) slight shifting time necessary to move between lines.

3.3.1. Empty-Empty Shifting (Figure 1(a)). In this situation the workers, as they move on, can start to assemble the new model, but the fastest line has to adapt to the slowest one; therefore the loss of time depends on the speed of both incoming and outcoming lines. For example, we can consider that the outcoming line is the fastest one, so the time necessary to perform the assembly operation in the following line is higher than the time necessary for the assembly operation on the previous line. The first worker moves to the new line and starts to work; afterwards, the second worker has to wait for a period equal to the difference between the lines assembly time before receiving the semiassembled product from previous position in the new line. Once his task on the outcoming line finishes, the third worker has to wait for the two former workers to complete their tasks on the new line. This means that waiting time accumulates and moves downstream and turns out to be the highest for the last operation. On the contrary, if the following line is the fastest, the analysis is symmetrical to the previous one. In short, the worker has to wait for a certain time, once he finishes his operations, if the empty-empty passage takes place towards a faster line; on the contrary, he has to wait for a certain time before beginning his operations if the passage is towards a slower line. The total loss of time in chessboard shifting ($\Delta\text{time}_{\text{loss}}$) is

$$\Delta\text{time}_{\text{loss}} = \frac{\Delta\text{TC}}{2} \cdot (N-1), \tag{1}$$

where N is the number of workers and ΔTC is the difference between cycle time of previous assembly line and that of following assembly line.

3.3.2. Empty-Full Shifting (Figure 1(b)). Once completing the last operation on the outcoming line, the first operator moves to the incoming line. As it is a full line, he can perform the operations on the semiassembled product in his own position, but he can get the line moving forward till the last worker shifts on the new line. The first operator has the higher, and not the lower, loss of time. Therefore, in empty-full shifting, loss of time decreases from upstream. The total loss of time is

$$\Delta\text{time}_{\text{loss}} = \frac{N_p - 1}{2} \cdot \text{TC}_p, \tag{2}$$

where N_p is the number of workers in the previous assembly line and TC_p is the cycle time of previous assembly line.

3.3.3. Full-Empty Shifting (Figure 1(c)). Once completing the batch on the line to leave, the work team moves together to a new empty line. From the first operator there is no loss of time, while for the other workers loss is equal to the time necessary to perform forthcoming operations. Therefore, in full-empty shifting, loss of time accumulates downstream. Loss of time is perfectly symmetrical to the former one:

$$\Delta\text{time}_{\text{loss}} = \frac{N_f - 1}{2} \cdot \text{TC}_f, \tag{3}$$

where N_f is the number of workers in the following assembly line and TC_f is the cycle time of following assembly line.

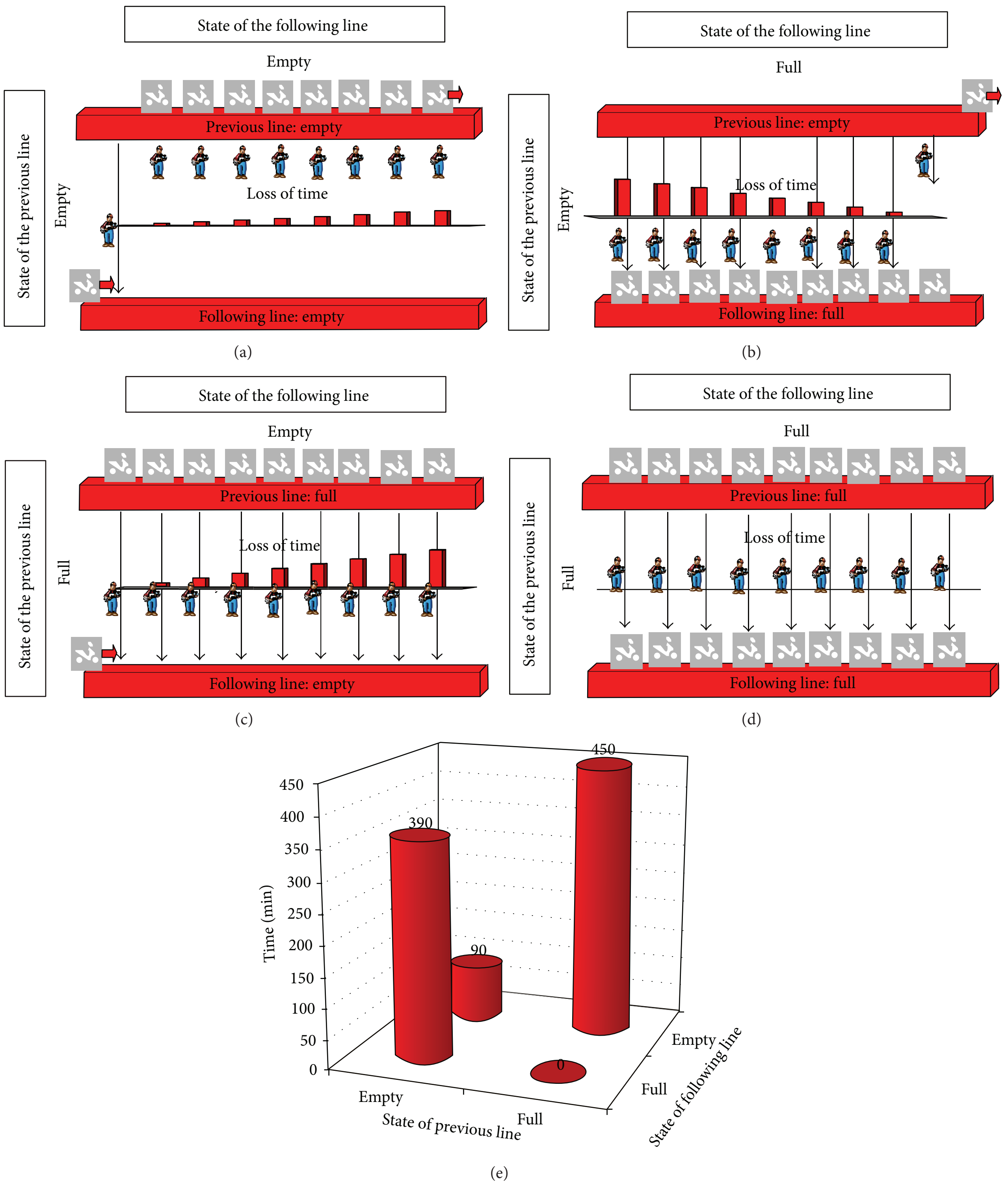

FIGURE 1: The loss of time in four types of chessboard shifting.

3.3.4. Full-Full Shifting (Figure 1(d)). In the full-full shifting, losses of efficiency are slight, as the work team moves together from one line to the following one. In this manner, production stops and restarts after the necessary time for the actual shifting of workers. Full-full passage acts if a work team, which is leaving a line, knows that when it comes back, it has to assemble a different model.

With reference to two lines with 10 work stations and cycle times of 80 and 100 minutes (see data on Table 1), it is possible to calculate the loss of time in the four chessboard conditions.

Table 1: Cycle times and number of workers.

	TC [min]	Number of workers
Previous line	80	10
Following line	100	10

Shifting between empty lines reports slight losses of time (90 minutes), depending on the differences between the cycle times of the assembly lines; on the contrary, shifting between full lines implies no losses of time. The passages between lines with different states, that is, full-empty and empty-full, are the most unfavorable ones, as they imply big losses of time, respectively, 450 and 390 minutes (Figure 1).

The experience led company to work in this way as follows.

(i) If shifting applies to work teams that operate on two lines, the firm assigns it to two lines whose models have a cycle time similar enough to perform only empty-empty shifting, in order to reduce both work in process (typical of full-full, empty-full, and full-empty) and losses of efficiency (see formula (3)).

(ii) If shifting applies to work teams that operate on three lines, full-full shifting is preferable in order to achieve the maximum efficiency (no loss of time) despite the reduction of WIP.

4. The Scheduling Approach

We designed and developed a model and a scheduler in order to manage the complexity of the production plans while considering the constraints of a system with mixed-model assembly lines. Although the model design started from the case study, the scheduling approach was to be a general purpose and customizable to several manufacturing companies. The mixed-model scheduling of assembly lines refers to the flowshop scheduling problem where m machines in the same order process a number n of jobs (batches for the assembly lines).

As above, the final assembly scheduling is very complex in the low season due to the difficult work force management and chessboard shifting. To this extent, the new scheduling model considers products as linked to the work teams and not to the assembly lines. The model schedules a feasible final assembly plan managing the work teams and not the assembly lines in order to solve the critical states coming from chessboard shifting and considering the internal constraints of capacity. Indeed, an empirical table (Table 2) presents the efficiency of work teams according to the team composition and the number of assembled motorbikes.

4.1. Macrostep e Microstep. The scheduling approach divides into two stages, a macrostep and a microstep (Figure 2). The main input is the master production schedule (MPS), where orders in batches are sequenced by request from the commercial unit which assigns a due date and a sequence number. The sequence number identifies the week when production batches must be assembled to deliver on time to the dealers. The other necessary input data scheduling is the cycle time of the assembly lines, the link among the products and the assembly lines, the work calendars, the work team composition, the efficiency table, priorities in the orders, and the penalty rates for earliness, tardiness, setup, and chessboard.

The macrostep is responsible for the following:

(i) defining the weekly MPS orders subset to process, with the lower sequence number,

(ii) assigning the orders and the work teams to the right assembly lines,

(iii) starting the microstep for building a set of solutions,

(iv) selecting the solutions to feed the next macrostep and the final state of the assembly lines.

The microstep is responsible for building a set of solutions (permutations) including nonpreferential orders selected by the macrostep. In particular, it uses two methods according to the numbers of MPS orders (N) as follows:

(i) if $N < 10$, all the $N!$ permutations are generated and passed to the evaluation module where the best one is selected;

(ii) if $N \geq 10$, the genetic algorithms are applied following the scheme in the next subsection and in Figure 3.

4.2. The Genetic Algorithms. Genetic algorithms are stochastic search techniques based on the mechanism of natural selection and evolution [22]. The main concepts of genetic algorithms are population, chromosome, gene, and fitness. GAs start with a population. A *population* is a set of solutions that could be generated randomly or by another algorithm. A chromosome, which is a single solution to the problem, represents the individuals of the population. A chromosome consists of a finite number of *genes*. Each gene represents an entity: for instance, in the scheduling problem it represents a job. At each generation of the algorithm, the chromosomes with a better *fitness* have higher probabilities to evolve. The evolution occurs with the genetic operators, which are crossover and mutation. After numerous generations, the algorithm converges to a chromosome, which is the solution of the problem. In our scheduling algorithm, we use the two-point crossover and the shift change mutation genetic operators. Murata et al. [15] showed that these two operators are effective for the flowshop problem.

4.2.1. Coding. Genetic algorithms encode the problem into a set of strings, each of which is composed of several bits, and then operate on the strings to simulate the process of evolution [23]. The solution of an optimization problem is usually encoded as a bit string. In the flowshop problem, which includes a number of jobs, the sequence of job has been encoded as a string like 12345 (the number indicates different batches). More generally, we consider an N-dimensional vector $\{J_1, \ldots, J_i, \ldots, J_N\}$, where J_i indicates the ith job. The solution is a permutation of the N given jobs.

Table 2: Work teams' efficiency.

	Number of vehicles			
	<500	501–2000	2001–6000	>6000
Seasonal workers				
Team composition A	0.39	0.61	0.69	0.88
Team composition B	0.46	0.67	0.80	0.91
Team composition C	0.47	0.70	0.80	0.91
Team composition D	0.48	0.73	0.86	0.93
Expert workers				
Team composition E	0.48	0.74	0.86	0.95
Team composition F	0.49	0.77	0.87	0.95
Team composition G	0.56	0.79	0.90	0.97
Team composition H	0.67	0.91	0.96	0.97

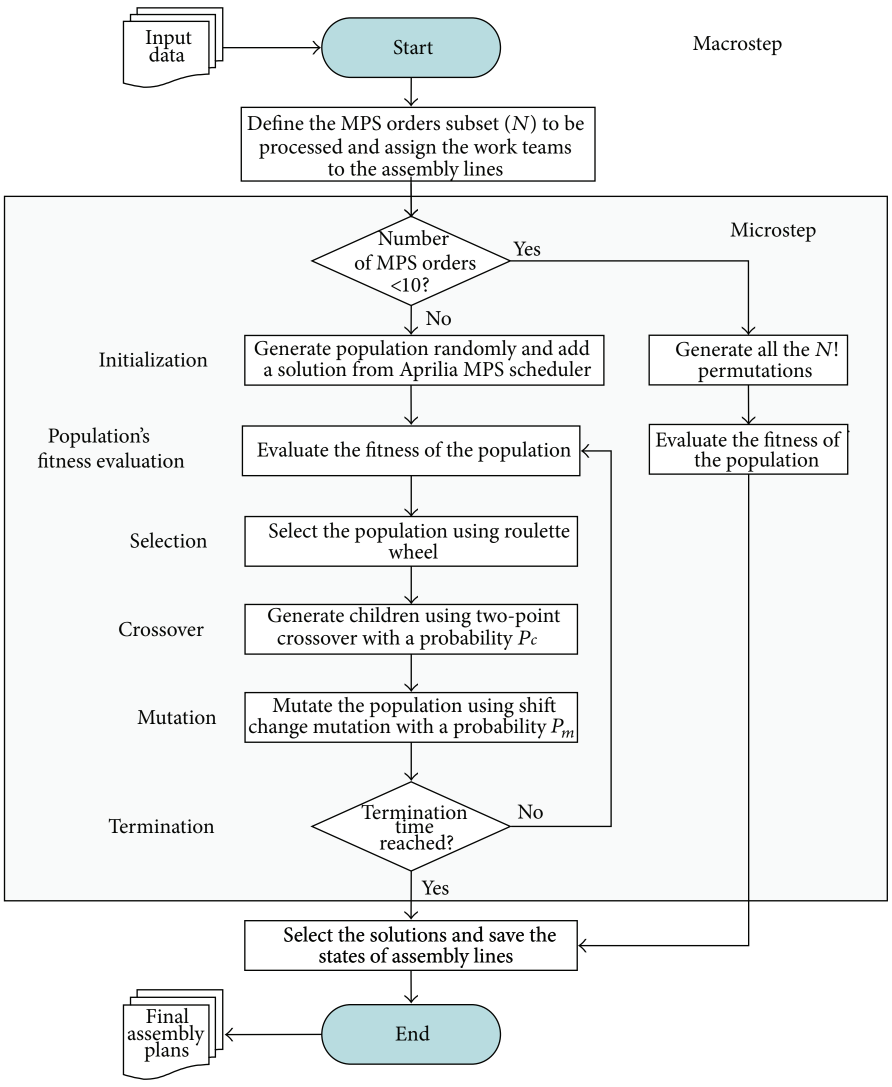
Input data
Start
Macrostep
Define the MPS orders subset (N) to be processed and assign the work teams to the assembly lines
Microstep
Number of MPS orders <10?
Yes
No
Initialization
Generate population randomly and add a solution from Aprilia MPS scheduler
Generate all the N! permutations
Population's fitness evaluation
Evaluate the fitness of the population
Evaluate the fitness of the population
Selection
Select the population using roulette wheel
Crossover
Generate children using two-point crossover with a probability Pc
Mutation
Mutate the population using shift change mutation with a probability Pm
Termination
Termination time reached?
No
Yes
Select the solutions and save the states of assembly lines
End
Final assembly plans

Figure 2: The scheduling approach.

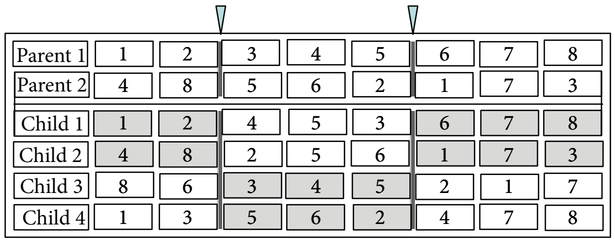

FIGURE 3: The two-point crossover.

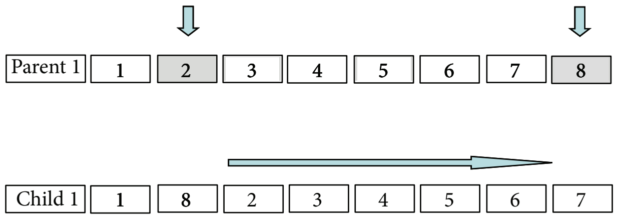

FIGURE 4: The shift change mutation.

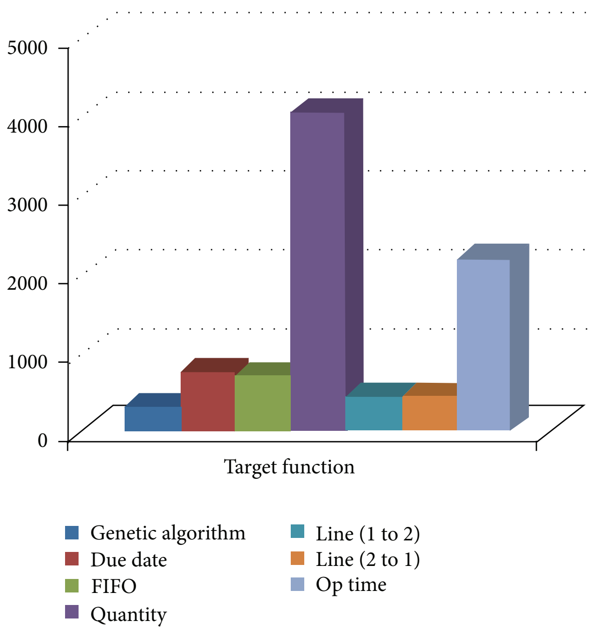

FIGURE 5: Comparison of objective function values.

4.2.2. Initialization. Usually, the initial population can be chosen completely random. However, we follow Reeves [24] and seed the initial population with a good solution. The solution comes from the macroscheduling and is engendered by constructive heuristics of an old scheduler, which is not able to consider the chessboard shifting.

4.2.3. Population's Fitness Evaluation. The microstep contains a module for evaluating the assembly plans. The evaluation module estimates the quality of the plans (permutations) of the microstep by using a multitarget function, which is the sum of weighted quantitative indicators. The target function assesses the fitness of the chromosome in assembling a batch. The weights represent the real priorities of the firm, coming from a discussion with the manufacturing planner. The multiobjective function has four targets of minimization: setup time, chessboard time, tardiness, and earliness, expressed by the following equation:

$$\begin{aligned}\text{Target function} = {} & \alpha_i \cdot \Sigma_{i-1,1}\text{SU} + \beta \cdot \Sigma_{i-1,1}\text{CB} \\ & + \Sigma_{i-1,1}\gamma_i \text{ET} + \Sigma_{i-1,1}\delta_i \text{TT},\end{aligned} \tag{4}$$

where SU = time lost to set up the line and start to assemble the batch i [day], CB = time lost for the chessboard and to start to assemble the batch i [day], ET = earliness towards the due date [day], TT = tardiness towards the due date [day], α_i = penalty rate for the setup of the line where batch i is to assemble [€/day], β_i = penalty rate for the time lost in the chessboard between the lines [€/day], γ_i = penalty rate of earliness to assemble batch i [€/day], and δ_i = penalty rate of tardiness to assemble batch i [€/day].

4.2.4. Selection. The selection process creates a new population for the next generation, selected among the parents and their children. We used a common roulette wheel selection [22]. The new population is selected with respect to the probability distribution based upon the fitness values.

4.2.5. Crossover. We use the two-point crossover, in order to generate four children for each couple of parents. The crossover operator has a crossover probability P_c = 1.0, as suggested by [15]. As for Figure 4, two points between two jobs are randomly selected for dividing one parent. From the first two children, the jobs outside the selected two points are kept from one parent to the child, and the other jobs are placed in the order they appeared in the other parent. From the other two children, the job inside is kept and the other jobs are placed in the same manner.

4.2.6. Mutation. The strings generated by the crossover operator are modified with the second genetic operator, the mutation. We applied the shift change mutation with a mutation probability P_m = 1.0, as suggested by [15]. As for Figure 5, two jobs inside the parent are randomly selected, a job at one position is removed and put at another position, and the other jobs shift to the right.

4.2.7. Termination. The process usually repeats for a fixed number of generations.

4.3. Results and Discussion. We implemented the model of the previous section in a scheduler coded in C++ and tested on a fourth-monthly master production schedule (MPS). We designed the scheduling software in order that the generations stop after 4 minutes (termination). This computation time is aligned with the requirements of the company, where scheduling is daily up to date.

To test the model and the scheduler we used real data of the company coming from an old MPS that refers to the first four months of the year (low season). According

Table 3: Results of the scheduling using GA.

Company's priorities	Penalty rates				Average number of delays	Moment of production completion
	α_i (setup)	β_i (chessboard)	γ_i (earliness)	δ_i (tardiness)		
Setup reduction	1	0	0	0	48.73	Day: 30/4 Minute: 186
Chessboard reduction	0	1	0	0	21.80	Day: 24/4 Minute: 417
Setup and chessboard reduction	1	1	0	0	51.03	Day: 23/4 Minute: 180
Delivery in time	0	0	0	1	6.03	Day: 28/4 Minute: 246
Setup and chessboard reduction and delivery in the right time	0.02	0.26	0.08	0.64	6.03	Day: 23/4 Minute: 340

to the experience of the company and to the analysis of Section 3, the chessboard logic is an empty-empty shifting. Furthermore, we scheduled the assembly plan of a single work team that operates shifting on two assembly lines with similar cycle time.

The information provided is as follows:

(i) MPS orders;

(ii) work calendar of the work team;

(iii) batch quantity;

(iv) delivery date of the order (due date);

(v) model-line relationship (assembly line of the model);

(vi) cycle time and number of work stations in the assembly lines (in relation to different models);

(vii) efficiency curves of the work teams.

The total number of MPS orders (batches) is 330 and the number of weekly orders (N) varies from 9 to 35 as follows:

(i) total population size: $N_{\text{poptot}} = 330$;

(ii) minimum weekly population: $N_{\min} = 9$;

(iii) maximum weekly population: $N_{\max} = 35$.

The starting date of the MPS plan is January 30 (30/01, minute 0) while the due date of the last batch is the end of April 24 (24/01, minute 480). The FAS plan of the company's scheduler had as moment of completion 30/04, minute 400 with a number of 50 batches late.

In Table 3, we synthesize the best and average results obtained by the scheduler in 100 runs with a termination time of 4 minutes. The use of the sole penalty rate for the setup (setting the other value to zero) leads to a reduction of the makespan (moment of production completion: day 30/4, minute 186). However, there are a high average number of delays compared to the due date (48.73 out of total 330 batches) because the scheduler takes into account only the loss of time in changing the batch. The use of the sole penalty rate for the chessboard leads to a higher reduction of the makespan and a less average number of delays (21.80) because the scheduler takes into account the loss of time for the movements of team between the assembly lines (day 24/4, minute 417). The simultaneous assignment of the two penalty rates with equal weight in the objective function leads to the best results in term of makespan reduction (day 23/4, minute 180) but gives the worst results in terms of average number of delays (51.03). The use of the sole penalty rate for the delay on delivery results in a deterioration in the makespan as the scheduler takes into account the sum of the delays (day 28/4, minute 246) but the average number of delays is the lower.

Finally, we interviewed the company planner and, according to his requirements, the penalty rates present the following values:

(i) $\alpha_i = 0.02$;

(ii) $\beta_i = 0.26$;

(iii) $\gamma_i = 0.08$;

(iv) $\delta_i = 0.64$.

Using all the four penalty rates, the scheduler produces a final assembly schedule with the best average number of delays (6.03) and a completion date near to the results of the sole penalty rate for the delay (day 23/4, minute 340).

In order to evaluate the quality of the output, we compared the results of the software using the four penalty rates with those produced by five heuristic rules of the company's scheduler as follows:

(i) due date: priority to the orders with the nearest due date,

(ii) FIFO: first-in first-out,

(iii) quantity: priority to the orders with the lower amount,

(iv) line: grouping the orders on a single line to limit the chessboard,

(v) Op time: priority to the orders with the fastest operation time.

As highlighted in Figure 6, the best result in terms of target function comes from the use of the genetic algorithms.

The operational performance parameters considered for the benchmark of the results are the average flowtime, the makespan, and the average saturation coefficient of the production system and of the human resources.

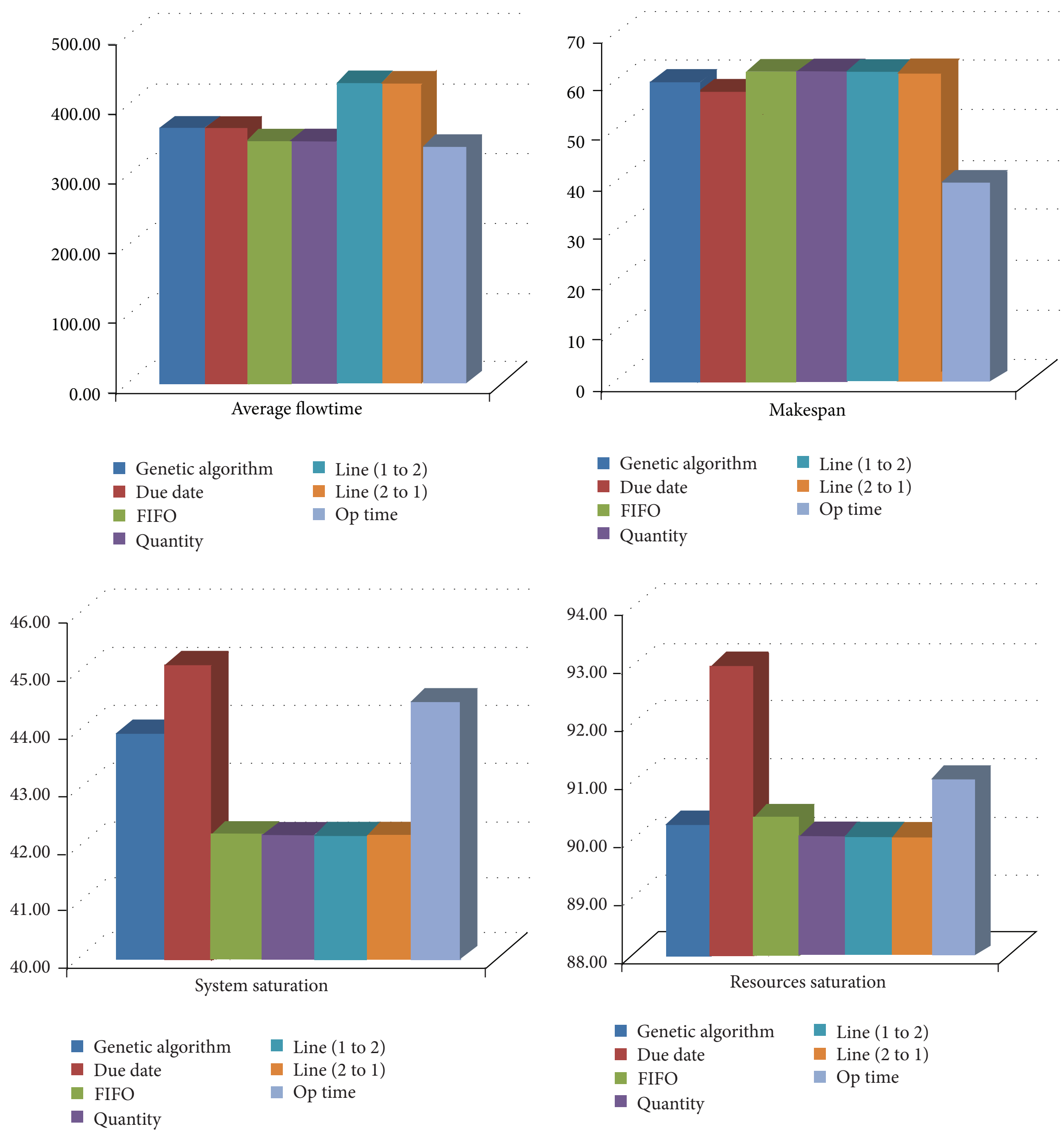

Figure 6: Comparison of performances parameters.

Average flowtime indicates how much time is elapsed, on average, from the moment a job enters until the production to the time is completed. It is expressed by the following equation:

$$F_{\text{med}} = \frac{\sum_{j=1}^{N} F_j}{N}, \tag{5}$$

where N = total number of jobs.

Minimizing this delay means increasing the number of operations the company performs in the timeline, leading to an increase in remaining production capacity. This principle however must be used only in case of real need (speed of response to the customer), as if the number of requests remain stable, the saturation level of the machines would decrease.

Makespan is the extent of time necessary to complete all the tasks. As in the case of the minimization of flowtime, a solution that minimizes the makespan should be adopted when company wants to increase the residual capacity of the existing resources with the aim of increasing the production volumes. The evaluation in this case is on a parameter that does not aggregate average values (such as the flowtime average) but retains the total impact. It is expressed by the following equation:

$$\text{MAK} = \max\left(C_j\right) - \min\left(I_j\right), \tag{6}$$

where C_j is the job j production completion date and I_j is job j production starting date.

The makespan depends on the scheduling since a bad operative planning may repeatedly cause the occurrence of bottlenecks.

Average saturation coefficient of the system indicates how long, compared to the makespan, the M machines are engaged in processing different orders. It is expressed by the following equation:

$$S_{\text{med}} = \frac{\sum_{i=1}^{M} \sum_{j=1}^{N} t_{i,j}}{\text{MAK}}, \tag{7}$$

where $t_{i,j}$ is the job j production time on the machine i.

This policy is convenient when every start and stop of the machine bring inefficiencies or when the purchase and maintenance of the resources are expensive. The main advantage of this choice is the ease of configuration of the line and of its arrest but with expenses in the saturation of the lines themselves.

Average saturation coefficient of human resources indicates the percentage of time the operator is engaged in assembling: any inefficiencies bring clearly a loss of competitiveness.

The following histograms (Figure 6) show the comparison among the results of the different heuristics and the GA, where each one represents a value of a specific operational performance parameter. We can observe that results of GA scheduler are not a dominant best, but results are good in all of the four performance indicators because the minimization of the objective function can be achieved as a trade-off of different subobjectives. The best results in all performance indicators come from the Op time heuristic; however, this heuristic does not consider lateness and earliness. The effect is the completion of batches before or after the due date and, consequently, a high fluctuation of the stock levels and delays in the delivery to the dealers. The schedule generated by GA produces a low level of stocks and a high service level.

5. Conclusions

The effectiveness of the final assembly plans, in a short elaboration time, supports the choice of GAs. The main objective of the scheduling model, that is, a feasible FAS that takes into account the constraints coming from a real manufacturing planning model, is fully achieved. Furthermore, significant savings in terms of time loss, in particular in the chessboard shifting, are attained, thus ensuring a high adherence to the delivery dates as for input data. Another great advantage comes in terms of flexibility of the model and robustness of the plans.

The limits of this research lie in the fact that the model and the scheduler have been developed and tested on a single case study. Despite this consideration, the model can be easily adapted to several manufacturing companies. In fact, we considered a multiobjective genetic algorithm for the scheduling problem with four targets: minimize the setup time, minimize the chessboard time, minimize the tardiness, and minimize the earliness. The target function can be easily modified by changing the penalty rate of a single objective to zero, for instance, the penalty rate of the chessboard time, which is typical of the motorbike company. Thanks to the generalization of the assumptions during the design stage, the model and the scheduler can easily become general purpose and customizable to several production systems. The high applicability of the model is widely exploitable, especially where the management of the work teams has a large impact on the productive performances.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] Y.-Y. Leu, L. A. Matheson, and L. P. Rees, "Sequencing mixed-model assembly lines with genetic algorithms," *Computers and Industrial Engineering*, vol. 30, no. 4, pp. 1027–1036, 1996.

[2] S. G. Ponnambalam, P. Aravindan, and P. S. Rao, "Comparative evaluation of genetic algorithms for job-shop scheduling," *Production Planning & Control*, vol. 12, no. 6, pp. 560–574, 2001.

[3] D. Levine, "Genetic algorithms: a practitioner's view," *INFORMS Journal of Computing*, vol. 9, no. 3, pp. 256–259, 1997.

[4] K. Metaxiotis and J. Psarras, "The contribution of neural networks and genetic algorithms to business decision support: academic myth or practical solution?" *Management Decision*, vol. 42, no. 2, pp. 229–242, 2004.

[5] J. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, Mich, USA, 1975.

[6] S. S. Chaudhry and W. Luo, "Application of genetic algorithms in production and operations management: a review," *International Journal of Production Research*, vol. 43, no. 19, pp. 4083–4101, 2005.

[7] S. Wadhwa, J. Madaan, and R. Raina, "A genetic algorithm based scheduling for a flexible system," *Global Journal of Flexible Systems Management*, vol. 8, no. 3, pp. 15–24, 2007.

[8] B. Kim and S. Kim, "Application of genetic algorithms for scheduling batch-discrete production system," *Production Planning and Control*, vol. 13, no. 2, pp. 155–165, 2002.

[9] P. Pongcharoen, C. Hicks, P. M. Braiden, and D. J. Stewardson, "Determining optimum genetic algorithm parameters for scheduling the manufacturing and assembly of complex products," *International Journal of Production Economics*, vol. 78, no. 3, pp. 311–322, 2002.

[10] F. Della Croce, R. Tadei, and G. Volta, "A genetic algorithm for the job shop problem," *Computers and Operations Research*, vol. 22, no. 1, pp. 15–24, 1995.

[11] S. G. Ponnambalam, V. Ramkumar, and N. Jawahar, "A multiobjective genetic algorithm for job shop scheduling," *Production Planning and Control*, vol. 12, no. 8, pp. 764–774, 2001.

[12] J. Yu, Y. Yin, and Z. Chen, "Scheduling of an assembly line with a multi-objective genetic algorithm," *International Journal of Advanced Manufacturing Technology*, vol. 28, no. 5-6, pp. 551–555, 2006.

[13] X. Shao, B. Wang, Y. Rao, L. Gao, and C. Xu, "Metaheuristic approaches to sequencing mixed-model fabrication/assembly systems with two objectives," *International Journal of Advanced Manufacturing Technology*, vol. 48, no. 9–12, pp. 1159–1171, 2010.

[14] S. G. Ponnambalam, P. Aravindan, and S. Chandrasekaran, "Constructive and improvement flow shop scheduling heuristics: an extensive evaluation," *Production Planning and Control*, vol. 12, no. 4, pp. 335–344, 2001.

[15] T. Murata, H. Ishibuchi, and H. Tanaka, "Genetic algorithms for flowshop scheduling problems," *Computers and Industrial Engineering*, vol. 30, no. 4, pp. 1061–1071, 1996.

[16] T. Murata, H. Ishibuchi, and H. Tanaka, "Multi-objective genetic algorithm and its applications to flowshop scheduling," *Computers and Industrial Engineering*, vol. 30, no. 4, pp. 957–968, 1996.

[17] R. Ruiz, C. Maroto, and J. Alcaraz, "Two new robust genetic algorithms for the flowshop scheduling problem," *Omega*, vol. 34, no. 5, pp. 461–476, 2006.

[18] C. J. Hyun, Y. Kim, and Y. K. Kim, "A genetic algorithm for multiple objective sequencing problems in mixed model assembly lines," *Computers and Operations Research*, vol. 25, no. 7-8, pp. 675–690, 1998.

[19] S. G. Ponnambalam, P. Aravindan, and M. Subba Rao, "Genetic algorithms for sequencing problems in mixed model assembly lines," *Computers and Industrial Engineering*, vol. 45, no. 4, pp. 669–690, 2003.

[20] R. Ruiz and C. Maroto, "A comprehensive review and evaluation of permutation flowshop heuristics," *European Journal of Operational Research*, vol. 165, no. 2, pp. 479–494, 2005.

[21] G. Celano, S. Fichera, V. Grasso, U. la Commare, and G. Perrone, "Evolutionary approach to multi-objective scheduling of mixed model assembly lines," *Computers and Industrial Engineering*, vol. 37, no. 1, pp. 69–73, 1999.

[22] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, Mass, USA, 1989.

[23] L. Davis, Ed., *Handbook of Genetic Algorithm*, Van Nostrand Reinhold, New York, NY, USA, 1991.

[24] C. R. Reeves, "A genetic algorithm for flowshop sequencing," *Computers and Operations Research*, vol. 22, no. 1, pp. 5–13, 1995.