Performance Study of a Genetic Algorithm for Sequencing in Mixed Model Non-Permutation Flowshops using Constrained Buffers*

Gerrit Färber and Anna M. Coves Moreno

Instituto de Organización y Control de Sistemas Industriales (IOC),
Universidad Politécnica de Cataluña (UPC), Barcelona, Spain
Gerrit_Faerber@gmx.de; Anna.Maria.Coves@upc.edu

Abstract. This paper presents the performance study of a Genetic Algorithm applied to a mixed model non-permutation flowshop production line. Resequencing is permitted where stations have access to intermittent or centralized resequencing buffers. The access to the buffers is restricted by the number of available buffer places and the physical size of the products. Characteristics such as the difference between the intermittent and the centralized case, the number of buffer places and the distribution of the buffer places are analyzed. Improvements that come with the introduction of constrained resequencing buffers are highlighted.

1 Introduction

In the classical production line, only products with the same options were processed at once. Products of different models, providing distinct options, were either processed on a different line or major equipment modifications were necessary. For today's production lines this is no longer desirable and more and more rise the necessity of manufacturing a variety of different models on the same line, motivated by offering a larger variety of products to the client. Furthermore, the stock for finished products is reduced considerably with respect to a production with batches, and so are the expenses derived from it.

Mixed model production lines consider more than one model being processed on the same production line in an arbitrary sequence. Nevertheless, the majority of publications in this area are limited to solutions which determine the job sequence before the jobs enter the line and maintain it without interchanging jobs until the end of the production line, which is known as permutation flowshop. Exact approaches for makespan minimization can be found in [1], [2], and [3], among others. In two recent reviews [4], [5] heuristic methods for sequencing problems are presented.

In the case of more than three stations and with the objective function to minimize the makespan, a unique permutation is no longer optimal. In [6] a study of

* This work is partially supported by the Ministry of Science and Technology, and the funding for regional research DPI2004-03472

the benefit of using non-permutation flowshops is presented. Furthermore, there exist various designs of production lines which permit resequencing of jobs: using large buffers (Automatic-Storage-and-Retrieval-System) which decouple one part of the line from the rest of the line [7]; buffers which are located off-line [8]; hybrid or flexible lines [9]; and more seldomly, the interchange of job attributes instead of physically changing the position of a job within the sequence [10]. Resequencing of jobs on the line is even more relevant with the existence of an additional cost or time, occurring when at a station the succeeding job is of another model, known as setup-cost and setup-time [11].

The present work considers a flowshop with the possibility to resequence jobs between consecutive stations. The buffers are located off-line either accessible from a single station (intermittent case) or from various stations (centralized case). In both cases, it is considered that a job may not be able to be stored in a buffer place, due to its extended physical size, see figure 1.

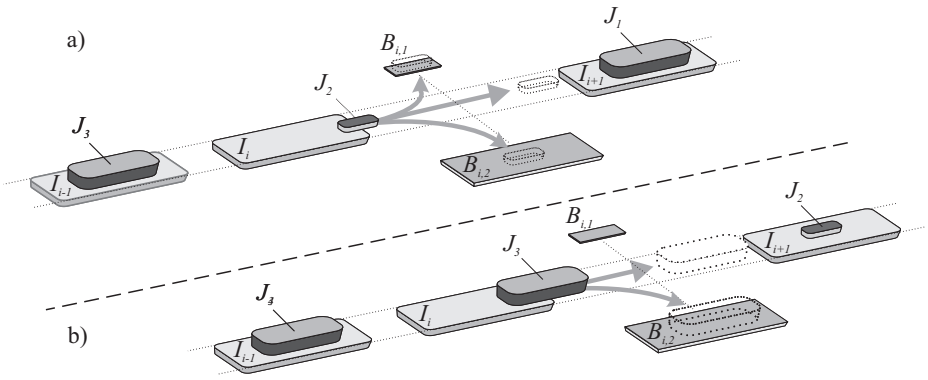


Fig. 1. Scheme of the considered flowshop. The jobs J_j pass consecutively through the stations I_i. The buffer B_i permits to temporally store a job with the objective of reinserting it at a later position in the sequence. a) Job J_2 can pass through any of the two buffer places $B_{i,1}$ or $B_{i,2}$ of buffer B_i. b) Job J_3 can pass only through buffer place $B_{i,2}$, due to its physical size.

The considered problem is relevant to various flowshop applications such as chemical productions dealing with client orders of different volumes and different sized resequencing tanks. Also in productions where split-lots are used for engineering purpose, such as the semiconductor industry. Even in the production of prefabricated houses with, e.g., large and small walls passing through consecutive stations where electrical circuits, sewerage, doors, windows and isolation are applied.

In what follows the problem is formulated with more detail and the applied Genetic Algorithm is described. Thereafter, the accomplished performance study is presented and finally conclusions are presented which are already useful at the time a production line is being designed.

2 Problem Definition

The realized work is based on the classical flowshop in which the jobs (J_1, J_2, ..., J_j, ..., J_n) pass consecutively through the stations (I_1, I_2, ..., I_i, ..., I_m). Furthermore, after determined stations, off-line buffers B_i permit to resequence jobs. The buffer provides various buffer places ($B_{i,1}$, $B_{i,2}$, ...) and each buffer place is restricted by the physical size of the jobs to be stored. As can be seen in figure 1a, job J_2 can be stored in buffer place $B_{i,1}$ as well as in $B_{i,2}$. Whereas, the next job J_3 can be stored only in buffer place $B_{i,2}$, because of the physical size of the job exceeding the physical size of the buffer place $B_{i,1}$, see figure 1b.

In a first step, the resequencing buffers are located intermittent, between two consecutive stations. In this case the buffer is assigned to the precedent station and may be accessed only by this station. Then, for an additional benefit, a single resequencing buffer is used, with access from various stations, while the limitations on the physical size of the buffer places are maintained.

3 Genetic Algorithm

The concept of the Genetic Algorithm was first formulated by [12] and [13] and can be understood as the application of the principles of evolutionary biology, also known as the survival of the fittest, to computer science. Genetic algorithms are typically implemented as a computer simulation in which a population of chromosomes, each of which represents a solution of the optimization problem, evolves toward better solutions. The evolution starts from an initial population which may be determined randomly. In each generation, the fitness of the whole population is evaluated and multiple individuals are stochastically selected from the current population, based on their fitness and modified to form a new population. The alterations are biologically-derived techniques, commonly achieved by inheritance, mutation and crossover. Multiple genetic algorithms were designed for mixed model assembly lines such as [14], [15], [16] and [17].

The heuristic used here is a variation of the Genetic Algorithm explained in [18]. The genes represent the jobs which are to be sequenced. The chromosomes υ, determined by a series of genes, represent a sequence of jobs. A generation is formed by R chromosomes and the total number of generations is G. In the permutation case, the size of a chromosome is determined by the number of jobs, the fraction Π. In the non-permutation case, the chromosomes are $L+1$ times larger, resulting in the fractions $\Pi_1^{'}, ..., \Pi_{L+1}^{'}$, being L the number of resequencing possibilities. In both cases, special attention is required when forming the chromosomes, because of the fact that for each part of the production line every job has to be sequenced exactly one time.

The relevant information for each chromosome is its fitness value (objective function), the number of job changes and the indicator specifying if the chromosome represents a feasible solution. A chromosome is marked unfeasible and is imposed with a penalty, if a job has to be taken off the line and no free buffer place is available or the physical size of the job exceeds the size limitation of the available buffer places. When two solutions result in the same fitness, the one with fewer job changes is preferred.

3.1 Genetic operators

The genetic operators specify in which way the subsequent population is generated by reproduction of the present population, taking into account that "fitter" solutions are more promising and therefore are more likely to reproduce. Even an unfeasible solution is able to reproduce, because of the fact that it may generate valuable and feasible solutions in one of the preceding generations. The used genetic operators are inheritance, crossover and mutation. The value p_X is the percentage with which a genetic operator X is applied to a chromosome.

Inheritance: This operator is determined by two parameters. The parameter p_{BS} determines the percentage of the best solutions which will be copied directly to the next generation, called the cluster of promising chromosomes, and ensures that promising chromosomes are not extinct. Then, in order to not remain in a local minimum, the parameter p_{b} determines the percentage of chromosomes which are removed from this cluster.

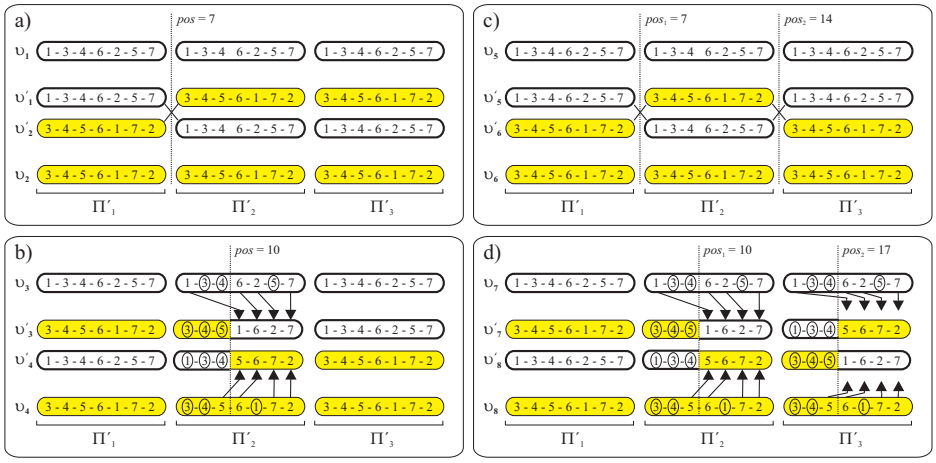


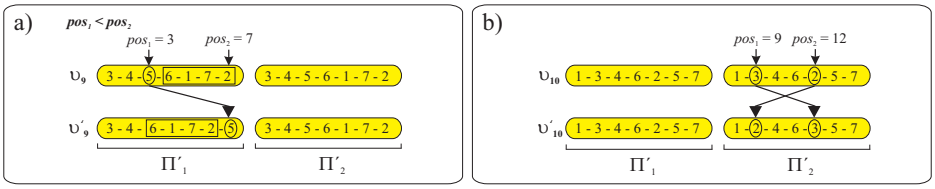
Fig. 2. Operators crossover-I and crossover-II. a) and c) In the simple case the crossing takes place between two main fractions of the chromosome. After the crossover point the chromosomes are completely crossed over. b) and d) In the more complex case it has to be assured that each job is sequenced exactly one time for each fraction of the chromosome.

Crossover: This operator specifies the operation of interchanging information of two chromosomes. Two crossover operations are applied, crossover-I (figure 2a,b) and crossover-II (figure 2c,d). The probabilities with which these operations are applied to a chromosome are $p_{\text{c-I}}$ and $p_{\text{c-II}}$, and the crossover points are defined by the random number pos, and the pair pos_1 and pos_2, respectively.

If the crossover point (pos, pos_1 and pos_2) is a multiple of the number of jobs to be sequenced, the crossover operation is simple and takes place between two main fractions of the chromosome, i.e. after the crossover point the chromosomes

are completely crossed over. Whereas, in the complex case the crossover points are located within a main fraction of the chromosome and it has to be assured explicitly that each job is sequenced exactly one time for each fraction of the chromosome.

Mutation: This operator specifies the operation of relocating jobs at position pos_1 to position pos_2 within the same fraction of a chromosome. Two mutation operators are applied, mutation-I and mutation-II (figure 3). Furthermore, there exist two cases for mutation-I: forward mutation, where $pos_1 < pos_2$; and backward mutation, where $pos_1 > pos_2$. In the first case, a single job has to be taken off the line, and in the second case, in order to let a single job pass, a group of succeeding jobs has to be taken off the line, resulting in a larger effort to realize. The probabilities of this operator are $p_{\text{m-I(f)}}$, $p_{\text{m-I(b)}}$ and $p_{\text{m-II}}$.



Mutation-I Mutation-II

Fig. 3. Operators mutation-I and mutation-II. a) The job at position pos_1 is taken off the line and reinserted to the line at position pos_2. b) The two jobs at position pos_1 and pos_2 are interchanged.

3.2 Cascading

In order to further improve the Genetic Algorithm, it is partitioned into two steps. In the first step, the possibility of resequencing jobs within the production line is ignored, furthermore only permutation sequences are considered as possible solutions and the chromosome size is reduced to the number of jobs. The last generation, together with the best found solution, form the initial generation for the next cascade where the resequencing possibilities, provided by stations with access to resequencing buffers, are taken into account.

3.3 Parameter adjustment

The adjustment of the values of the genetic operators, which are used for the two cascades of the Genetic Algorithm, is determined with an extended experimentation and consists of three steps:

Rough adjustment: In order to adjust the parameters in a robust manner, different sets of parameters are applied to a series of 14 differently sized problems, varying the number of jobs to be sequenced, the number of stations and the number of resequencing possibilities. During the rough adjustment only one unique seed is used for the random number generation in the Genetic Algorithm. The sets of parameters are summarized and the 300 most promising which show good performance on all problem sizes are used for further adjustment.

Repeatability: The use of only one seed in the rough adjustment requires to determine amongst the promising sets of parameters which set achieves good results for a multitude of seeds. The fact that a set of parameters achieves good results for different seeds indicates that the same set of parameters also performs well for different plant setups. The promising sets of parameters are verified with 16 different seeds for the 14 differently sized problems. Once the sets of promising parameters are examined with respect to repeatability, one set is used for the fine adjustment, determined by grouping into clusters [19].

Fine adjustment: Due to the fact that in the previous analysis predetermined discrete values for the parameters are used, a fine adjustment succeeds. The genetic operators are subject to an adjustment of 0.1 for the previously determined sets of parameters and are revised with 16 seeds for the 14 differently sized problems, used for the repeatability.

Table 1. Characteristic values of the Genetic Algorithm. The first cascade is applied to determine a generation with only permutation solutions, which is then used as an initial generation for the second cascade.

Cascade	R	G	p_{BS}	p_{b}	$p_{\text{c-I}}$	$p_{\text{c-II}}$	$p_{\text{m-I(f)}}$	$p_{\text{m-I(b)}}$	$p_{\text{m-II}}$
Step 1	100	1000	0.05	0.1	0.3	0.6	0.25	0.25	0.25
Step 2	100	10000	0.05	0.4	0.5	0.35	0.45	0.1	0.1

The experimentation is first performed on the permutation case (first cascade), and then on the non-permutation case (second cascade). Table 3.3 lists the resulting values of the genetic operators used in the following performance study.

4 Performance Study

For the study of performance, a flowshop which consists of 5 stations is considered. The range of the production time is [0...20] such that for some jobs exists zero-processing time at some stations, for the setup-time [2...8] and for the setup-time [1...5]. The number of jobs is varied from 5 to 100 with increments of 5. The objective function, is the weighted sum of the makespan (factor of 1.0) and the setup-cost (factor of 0.3), where the setup-time has is not concerned with a weight but is indirectly included in the calculation of the makespan.

4.1 Intermittent versus centralized location

Replacing the intermittent resequencing buffer places with centralized resequencing buffer places has two benefits. On the one hand, for the case of the same number of buffer places, the objective function of the final solution is expected to be at least as good. This is caused by the fact that in some instances of time, all buffer places of a certain intermittent resequencing buffer may be occupied and do not allow an additional job to be removed from the line, while buffer places from other intermittent resequencing buffers are not accessible. Whereas, in the case of a centralized buffer, blocking only appears when all buffer places are occupied.

On the other hand, the number of buffer places may be reduced in order to obtain values of the objective function similar to the case of the intermittent resequencing buffer. Depending on the number of buffer places which are reduced, this reduction in area is significant.

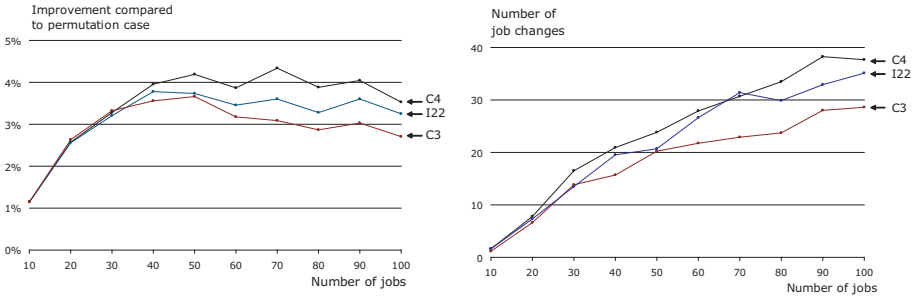


Fig. 4. Comparison of three cases: one centralized resequencing buffer with four buffer places (C4); two intermittent resequencing buffers, each providing two buffer places (I22); one centralized resequencing buffer with three buffer places (C3).

Figure 4 shows the comparison of the intermittent and the centralized case. After the second station and after the third station there exists access to resequencing buffers. The compared cases are: (I22) two intermittent resequencing buffers, each buffer provides two buffer places; (C3, C4) one centralized resequencing buffer providing three and four buffer places, respectively. On the one hand better solutions are obtained by arranging the buffers centralized; on the other hand, the reduction from four to three buffer places leads to solutions nearly as good as in the intermittent case.

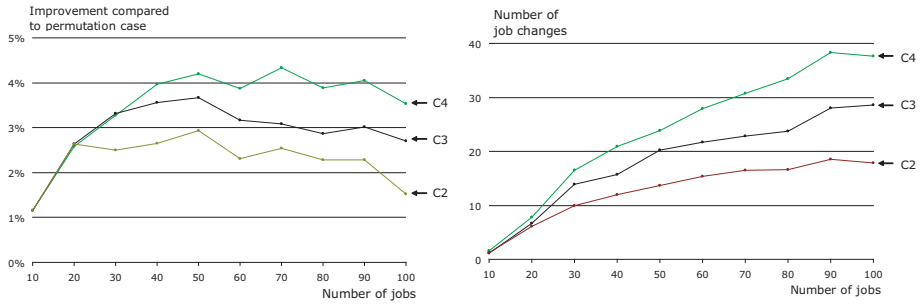


Fig. 5. Variation of the number of buffer places for the centralized case.

4.2 Number of buffer places

The increase in the number of buffer places makes the limitations less strict and as already seen in the previous case, solutions are expected to improve. Figure 5 shows the centralized case without physical size limitations. Jobs leaving the second or the third station have access to the centralized buffer, provided with 2, 3 or 4 buffer places. Providing more buffer places results in better solutions together with an elevated number of job changes.

Figure 6 illustrates an intermittent case. In I22 the second and the third station have 2 buffer places each, in I20 the buffer after the second station has two buffer places and in I02 the buffer after the third station has two buffer places. Providing two more buffer places results in slightly better solutions together with an elevated number of job changes.

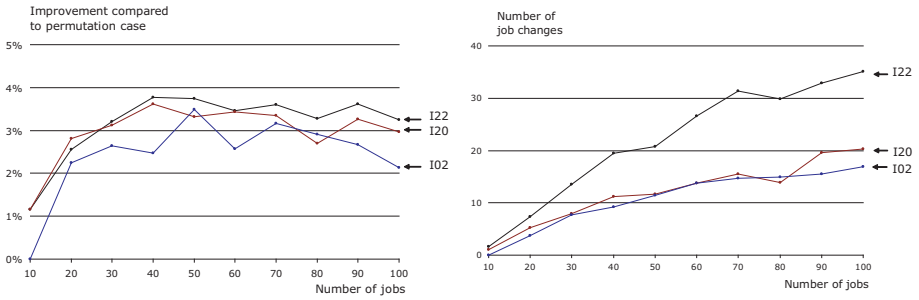


Fig. 6. Variation of buffer places for the case of the intermittent case.

4.3 Difference in physical size of buffer places

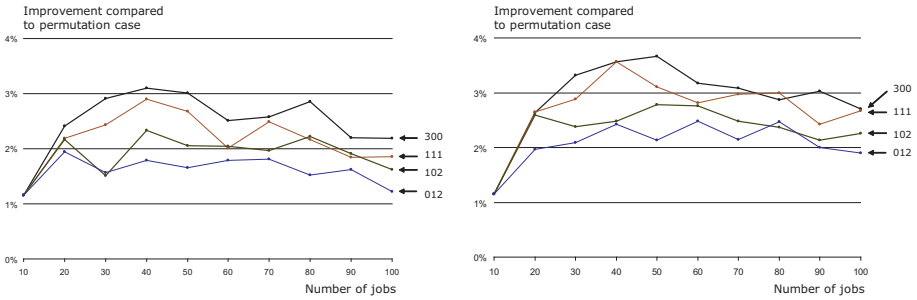
Introducing limitations on the physical size of the buffer places on one side restricts possible solutions but on the other side minimizes the necessary buffer area. This limitation arises, for example, in a chemical production. The arrangement of two tanks which are located off the line, accessible after a certain station, equals an intermittent resequencing buffer with two buffer places. With tank capacities of 50 and 100 liters, a client order of 80 liters can be stored only in the larger of the two tanks which is capable of storing this volume. Whereas, a client order of 50 liters can be stored in either of the tanks. A close look at the local conditions may amortize an increase in the objective function compared to a reduction of investment with respect to tank size and gained area.

Table 2. Allocation of the buffer places to the buffers. In the intermittent case the allocation is done to two different buffers.

Case	Intermittent			Centralized		
Size	l	m	s	l	m	s
(300)	1/2	0/0	0/0	3	0	0
(111)	0/1	1/0	0/1	1	1	1
(102)	0/1	0/0	1/1	1	0	2
(012)	0/0	0/1	1/1	0	1	2

As a concrete example, three differently sized buffer places (large, medium, small) are available and the ratio of jobs is $\frac{3}{10}$ large, $\frac{3}{10}$ medium and $\frac{4}{10}$ small. As in the previous section, the second and the third station have access to the resequencing buffers and table 4.3 shows the allocation of the buffer places to the buffers, considering eight scenarios. ”300” represents 3 large, 0 medium

and 0 small buffer places. In the intermittent case the first buffer is provided with 1 and the second buffer with 2 large buffer places. In the centralized case the same two stations have access to a single centralized buffer, containing the three buffer places. Figure 7 shows the influence of the limitation of the physical size. The variation of the size of the buffer places towards smaller buffer places on the one hand decreases the benefit achieved by the possibility of resequencing jobs. On the other hand, it may amortize when taking into account the reduction of investment with respect to tank size and gained area.



a) Intermittent case b) Centralized case

Fig. 7. Influence of the variation of the physical size of the buffer places. "102" represents 1 large, 0 medium and 2 small buffer places. In the intermittent case, the buffer places are divided to two buffers, each with access from a designated station. In the centralized case, the same two stations have simultaneously access to the buffer, containing all three buffer places. The ratio of jobs is $\frac{3}{10}$ large, $\frac{3}{10}$ medium and $\frac{4}{10}$ small.

5 Conclusions

This paper has presented a study of performance of a genetic algorithm which was applied to a mixed model non-permutation flowshop. The algorithm uses the genetic operators inheritance, crossover and mutation and is designed to consider intermittent or centralized resequencing buffers. Furthermore, the buffer access is restricted by the number of buffer places and the physical size of jobs.

The realized study of performance demonstrates the effectiveness of resequencing by examining certain characteristics. The results of the simulation experiments reveal the benefits that come with a centralized buffer location, compared to the intermittent buffer location. It either improves the solution or leads to the utilization of fewer resequencing buffer places. An increased number of buffer places clearly improves the objective function and including buffers, constrained by the physical size of jobs to be stored, on one side limits the solutions but on the other side minimizes the necessary buffer area.

In order to take full advantage of the possibilities of resequencing jobs in a mixed model flowshop, additional installations may be necessary to mount, like buffers, but also extra efforts in terms of logistics complexity may arise. The additional effort is reasonable if it pays off the necessary investment. Due to the dependency on local conditions, a general validation is not simple and was not part of this work.

References

[1] Ignall, E., Schrage, L.: Application of the branch and bound technique to some flow-shop problems. Operations Research **13**(3) (1965) 400–412

[2] Potts, C.: An adaptive branching rule for the permutation flowshop problem. European Journal of Operational Research **5**(2) (1980) 19–25

[3] Carlier, J., Rebai, I.: Two branch and bound algorithms for the permutation flowshop problem. European Journal of Operational Research **90**(2) (1996) 238–251

[4] Framinan, J., Gupta, J., Leisten, R.: A review and classification of heuristics for permuation flowshop scheduling with makespan objective. Technical Report OI/PPC-2001/02 (2002) Version 1.2.

[5] Framinan, J., Leisten, R.: Comparison of heuristics for flowtime minimisation in permuation flowshops. Technical Report IO-2003/01 (2003) Version 0.5.

[6] Potts, C., Shmoys, D., Williamson, D.: Permutation vs. non-permutation flow shop schedules. Operations Research Letters **10**(5) (1991) 281–284

[7] Lee, H., Schaefer, S.: Sequencing methods for automated storage and retrieval systems with dedicated storage. Computers and Industrial Engineering **32**(2) (1997) 351–362

[8] Lahmar, M., Ergan, H., Benjaafar, S.: Resequencing and feature assignment on an automated assembly line. IEEE Transactions on Robotics and Automation **19**(1) (2003) 89–102

[9] Engstrőm, T., Jonsson, D., Johansson, B.: Alternatives to line assembly: Some swedish examples. International Journal of Industrial Ergonomics **17**(3) (1996) 235–245

[10] Rachakonda, P., Nagane, S.: Simulation study of paint batching problem in automobile industry. http://sweb.uky.edy/˜pkrach0/Projects/MFS605Project.pdf (2000) consulted 14.07.2004.

[11] Bolat, A.: Sequencing jobs on an automobile assembly line: objectives and procedures. International Journal of Production Research **32**(5) (1994) 1219–1236

[12] Holland, J.: Genetic algorithms and the optimal allocation of trials. SIAM J. Comput. **2**(2) (1973) 88–105

[13] Holland, J.: Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor (1975)

[14] Bolat, A., Al-Harkan, I., Al-Harbi, B.: Flow-shop scheduling for three serial stations with the last two duplicate. Computers & Operations Research **32**(3) (2005) 647–667

[15] Levitin, G., Rubinovitz, J., Shnits, B.: A genetic algorithm for robotic assembly line balancing. European Journal of Operational Research **168** (2006) 811–825

[16] Wang, L., Zhang, L., Zheng, D.: An effective hybrid genetic algorithm for flow shop scheduling with limited buffers. Computers & Operations Research (2006) Article in Press.

[17] Rubén, R., Concepción, M.: A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility. European Journal of Operational Research **169**(3) (2006) 781–800

[18] Michaelewicz, Z.: Gentic Algorithms + Data Structures = Evolution Programs. 3rd edn. Springer Verlag (1996)

[19] Balasko, B., Abonyi, J., Feil, B.: Fuzzy clustering and data analysis toolbox; for use with matlab. (2005)