Journal of Industrial and Systems Engineering
Vol. 10, special issue on production and inventory, pp 57-72
Winter (February) 2017

An improved memetic algorithm to minimize the earliness–tardiness on a single batch processing machine

Neda Rafiee Parsa[1], Behrooz Karimi[1*], Seyed Mohammad Moattar Husseini[1]

[1]*Department of Industrial Engineering and Management Systems, Amirkabir University of Technology, Tehran, Iran*

n.rafieeparsa@aut.ac.ir, b.karimi@aut.ac.ir, moattarh@aut.ac.ir

Abstract

In this research, a single batch processing machine scheduling problem with minimization of total earliness and tardiness as the objective function is investigated.We first formulate the problem as a mixed integer linear programming model. Since the research problem is shown to be NP-hard, an improved memetic algorithmis proposed to efficiently solve the problem. To further enhance the memetic algorithm and avoid premature convergence, we hybridize it with a variable neighborhood search procedureas its local search engine. A dynamic programming approach is also proposed to find optimal schedule for a given set of batches. Wedesign a Taguchi experiment to evaluate the effects of different parameters on the performance of the proposed algorithm. The results of an extensive computational study demonstrate the efficacy of the proposed algorithm.

Keywords: Batch processing machine, total earliness and tardiness, memetic algorithm, variable neighborhood search, dynamic programming.

1- Introduction

A batch processing machine (BPM) can process several jobs simultaneously. The jobs that are processed together on the machine are referred as a batch. All the jobs in a batch have the same start and completion time on the machine. The processing time of a batch is equal to the largest processing time of the jobs in the batch. There are several applications of batch processing machines in industry practice. Burn-inoven operations in semiconductor industries and chemical processes performed in tanks or kilns arethe examples of such problems. Mönch et al. (2011) performed a comprehensive literature review of scheduling semiconductor manufacturing operations.

This paper is motivated by burn-in oven operations in semiconductor manufacturing. Burn-in ovens are batch processing machine and are used to test heat-stress of integrated circuit chips. Several chips can be tested in a burn-in oven simultaneously. The burn-in process is often a bottleneck step in the back-end process of semiconductor manufacturing because its processing time is much longer than that of the other steps.Therefore, optimal scheduling on burn-in ovens is very important in semiconductor manufacturing.

The earliness and tardiness penaltiesare important measurement in Just-in-time production systems.Early jobs may increase holding costs and costs related to the deterioration of finished or perishable goods.

*Corresponding author
ISSN: 1735-8272,

However, tardy jobs lead to lost sales and customer dissatisfaction and hence loss of reputation. Minimizing the earliness and tardiness penalties can improve the efficiency and cost-effectiveness of a production system.

So, incorporating earliness and tardiness considerations is so important in the current competitive environment.

In this paper, minimization of total earliness and tardiness on a single batch processing machine with non-identical job sizes is considered. It is assumed that all the jobs have a common and loose due date. Common due date assumption is applicable in many production systems,such as base wafers in the front-end of burn-in ovens. Base wafersare preprocessed wafers that held on stock for further processing based on the specific customer requests. In this situation, a large number of chips have the same external due date and hence the same internal due date with respect to the burn-in oven (Mönch et al., 2006).

Despite the research problem is shown to be NP-hard byBrucker et al. (1998), there are a few research that propose heuristic and metaheuristic algorithmsfor the research problem in the literature. Thus, there are rooms for proposing effective algorithms for the problem. This motivates us to focus on developing an improved memetic algorithm which employs variable neighborhood search as its local search procedure.

The rest of the paper is organized as follows: Related literature to our problem is briefly provided in Section 2. The characteristics of the problem and a mathematical formulation are presented in Section 3.The details ofthe proposed improved memetic algorithm, including variable neighborhood search and dynamic programming, are introducedin Section 4.The experimental design to evaluatethe mathematical model and the proposed algorithmare reported in Section 5. Finally, conclusions and directions for the future research are discussed in Section 6.

2- Related literature

Many researchers focused on batch processing machine scheduling problems due to its industrial applications. In this section, only the papers having the most similarities with our assumptions are reviewed.A single batch processing machine with non-identical job sizes was considered by Uzsoy (1994). He gave complexity results for both makespan minimization and total completion time minimization and also provided some heuristics and a branch and bound algorithm for these problems. Dupont and Jolai (1998) and Jolai and Dupont (1998)proposed Several heuristic algorithms for the same problems. A branch and bound algorithm to minimize the makespan was developed by Dupont and Dhaenens-Flipo (2002).Rafiee Parsa et al. (2010) proposed a branch and price algorithm for the same problem. They showed that their proposed algorithm has a better performance than the branch and bound algorithm proposed by Dupont and Dhaenens-Flipo (2002).An algorithm based on clustering techniques to minimize the makespan was proposed byChen et al. (2011).The problem of minimizing total weighted tardiness with job release dates was considered byWang (2011). Heproposed a two-phase heuristic to obtain approximate solutions.Malapert et al. (2012) presented a constraint programming approach to minimize the maximum lateness. The problem of minimizing makespan with dynamic job arrivals was addressed byZhou et al. (2014). They proposed a number of constructive heuristics. Cabo et al. (2015) introduced a new neighborhood search, called split–merge, to minimize the maximum lateness of the jobs. For more details about the literature of batch scheduling problems, we refer the reader toPotts and Kovalyov (2000), Mathirajan and Sivakumar (2006),and Mönch et al. (2011).

There are several research efforts focused on developing metaheuristic algorithms for the single batch processing machine in recent years,e.g.,Cheng et al. (2010), Xu et al. (2012), Damodaran et al. (2013), Jia and Leung (2014), and Al-Salamah (2015).

Considering non-regular objective functions, Qi and Tu (1999) addressed the problem ofminimization earliness and tardiness on a single batch processing machine when the jobs have a distinct due date.They assumed that all of the jobs have identical job sizes and the same processing times. They proposed a dynamic programming algorithm to solve the problem in polynomial time. Brucker et al. (1998) proved that all the batch scheduling problems with due date related criteria are NP-hard. It follows that the minimization of earliness and tardiness is also NP-hard.The earliness and tardiness minimization problem with a common due date and identical job sizes under a maximum allowable tardiness constraint was considered byMönch et al. (2006). They proposed a hybrid genetic

algorithm for the problem. Zhao et al. (2006) considered a common due window scheduling problem with batching on a single machine. They developed a polynomial time algorithm for minimizingthe penalty of total weighted earliness and tardiness. Mönch and Unbehaun (2007) developed three decomposition heuristics to minimize the earliness and tardiness on parallel burn-in ovens with unit job size and a common due date.Recently, Li et al. (2015) extended the problem of minimizing the earliness and tardiness to the case of non-identical job sizes. They proposed a hybrid genetic algorithm for the problem.

Concluding from the literature review, a few research considered the problem of minimizing the earliness and tardiness of all the jobs with non-identical job sizes. Considering the industrial relevance of this problem, it is obvious that there are rooms for developing heuristic and metaheuristic algorithms for the research problem.

3- Problem formulation

The single BPM problem with non-identical job sizes to minimize the total earliness and tardiness of jobs is considered in this research. There is a set of $n$jobs which are available for processing on a batch processing machine. Each job$j = 1, \dots, n$ is characterized by its processing timep_j and sizes_j. All the jobs have a common and loose or nonrestrictive due date$d$, which is greater than or equal to the makespan of the given set of jobs, so we assume that$d \geq \sum_j p_j$. The machine can process a group of jobs as a batch as long as the total size of the batch is less than or equal to the machine capacity B. Jobs cannot be split across the batches. Once the processing of a batch is initiated, it cannot be interrupted and other jobs cannot be introduced into the machine until the processing is completed.The processing time of a batch$b$ is determined by the longest processing time among the jobs in the batch. Based on the standard classification scheme for scheduling problems (Graham et al., 1979), the above problem can be noted as $1|p-batch, s_j \leq B, d_j = d| \sum(E_j + T_j)$.

The decision variables and the binary mixed integer linear programming (BMILP) formulation of the problem under study are as follows:

Decision variables:

x_{jb}= $\begin{cases} 1 & \text{If job } j \text{ is assigned to batch } b \\ 0 & \text{Otherwise} \end{cases}$

C_b: Completion time of batch b

ET_b: Absolute deviation of the completion time of batch b from the due date d

ET_j: Absolute deviation of the completion time of job j from the due date d

P_b: Processing time of batch b

The model:

Minimize $\sum_{j=1}^{n} ET_j$ (1)

Subject to: $\sum_{b=1}^{n} x_{jb} = 1 \qquad j = 1, \dots, n$ (2)

$\sum_{j=1}^{n} s_j x_{jb} \leq B \qquad b = 1, \dots, n$ (3)

$P_b \geq x_{jb} p_j \qquad j = 1, \dots, n;\ b = 1, \dots, n$ (4)

$C_1 \geq P_1$ (5)

$$C_b \geq C_{b-1} + P_b \qquad b = 2, \dots, n \qquad (6)$$

$$ET_b \geq d - C_b \qquad b = 1, \dots, n \qquad (7)$$

$$ET_b \geq C_b - d \qquad b = 1, \dots, n \qquad (8)$$

$$ET_j \geq ET_b - M(1 - x_{jb}) \qquad j = 1, \dots, n;\ b = 1, \dots, n \qquad (9)$$

$$P_b, C_b, ET_b, ET_j \geq 0 \qquad j = 1, \dots, n;\ b = 1, \dots, n \qquad (10)$$

$$x_{jb} \in \{0,1\} \qquad j = 1, \dots, n;\ b = 1, \dots, n \qquad (11)$$

Minimizingthe total earliness and tardiness of jobs is expressed by equation (1)as the objective function. Constraint set (2) ensures that each job is assigned exactly to one batch. The total size of all the jobs in a particular batch cannot exceed the machine's capacity. Constraint set (3) is incorporated into the model for this reason. Constraint set (4) determines the processing time of each batch. Constraint sets (5) and (6) ensure that the completion time of each batch is greater than or equal to the completion time of its predecessor batch plus its processing time. The absolute deviation of the completion time of batch b from the due date d is determined by constraint sets (7) and (8). If job j is assigned to batch b, then $ET_j$is equal to ET_b. Constraint set (9) is incorporated into the model for this reason. In this constraint,M is a large enough constant. Since the value of earliness and tardiness of batches is less than sum of the processing time of jobs, we can set $M = \sum_{j=1}^{n} p_j$.Constraint sets (10) and (11) specify the type of decision variables.The number of variables and the number of constraints in this model are $n^2 + 4n$ and $2n^2 + 5n$, respectively.

4- An improved memetic algorithm

The research problem is shown to be NP-hard byBrucker et al. (1998).Thus,the research hasfocused on developing metaheuristicalgorithm for finding near optimal solutions for large-sized problems.Memetic algorithm (MA) is an evolutionary algorithm which combines genetic algorithm (GA) with local search procedures.MA was first introduced as a hybrid GA combined with an individual learning procedure for local refinement by Moscato (1989). Previous researchshows that MA has a good performance in scheduling and timetabling problems (Hart et al., 2005). In the proposed algorithm,called MA-VNS, a variable neighborhood search algorithm (VNS) is usedas local search procedure, to enhance the memetic algorithm.The detailsof theproposedMA-VNSalgorithm for the research problem is as follows:

4-1- Solution representation

In evolutionary algorithms, defining a proper solution representation strongly affects the computational effort in crossover, mutation, and local search procedures. InMA algorithm, solutions are represented as chromosomes.Different steps of the algorithm such as crossover and mutation are applied on chromosomes. Each chromosome in the proposed MA-VNS algorithm is defined as the sequence of jobs. So each permutation of digits 1 to $n$represents a chromosome or solution.

4-2- Initial population

The initial population can be generated from various methods. Sequencing rules such as the longest processing time (LPT), and the shortest processing time (SPT) can be used for generating the initial population. So, in the proposed algorithm two of the initial chromosomes are generated based on the LPT and SPT orders. For the rest chromosomes, the jobs are arranged in a random sequence. The number of chromosomes within the population through generations is presented by *pop_size*.

4-3- Selection strategy for recombination

At each generation, the algorithm selects a few chromosomes from the population for recombination and generating new chromosomes. The chromosomes that are combined to produce new ones are called parents and the newly generated ones are called offspring. In this researcha binary tournament selection strategy is used for choosing a pair of parents.In the binary tournament selection,

two chromosomes are randomly chosen with equal probability from the population. Then, the best chromosome of these two ones is considered as the first parent. The process is repeated in the same wayto select the second parent.

4-4-Crossover operator

Combining chromosomes for generating new ones is called crossover. In this research, two point order crossover is used for combining two parents to produce new offspring. Assume that two chromosomes are selected for recombination. Initially two random numbers are uniformly generated to determine the piece of the first parent that will be copied to the offspring. Then, this piece is copied to the same position of the offspring. The rest of offspring is completed according to the second parent. The digits that do not appear in the copied piece from the first parent complete the empty alleles of offspring by considering their order in the second parent.Apossible combination by applying order crossover is presented in Figure 1.The copied piece from the first parent is highlighted. Since the first parent transfers more information to offspring, the first parent is considered as the chromosome with better fitness between two selected chromosomes for crossover.

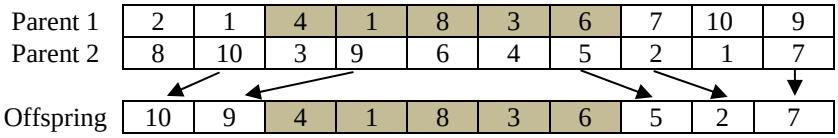

Figure 1. Order crossover operator

The number of newly generated chromosomes through crossover procedure is dependent on the*crossover rate*. Crossover rate is a number which is evaluated as a ratio of number of newly generated ones to *pop_size*. If the population size is 50 and the crossover rate is 0.6, then at each generation of the algorithm 50×0.6=30 new chromosomesare produced through crossover procedure. In this case, 30 pairs of chromosomes should be selected for applying crossover operator on them.

4-5- Mutation operator

Mutation procedures are used in evolutionary algorithms for diversification. In this research, after generating new chromosomes through crossover procedure,a few of them are selected to go through swap mutation operator. In swap mutation procedure, two alleles are selected and their positions are swapped as presented in Figure 2. In this case, the third and fifth allelesare randomly selected and then swapped.

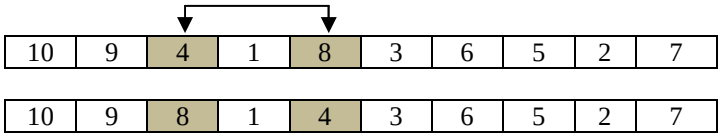

Figure 2. Mutation operator

Mutation rate determines the number of offspring that should go through mutation procedure at each generation of the algorithm. For instance, if mutation rate is set to 0.02 then at each generation, mutation would be applied on 2% of offspring.

4-6- Variable neighborhood search

In MA, each new offspring go through a local search procedure to find a new solution with a better fitness.A local search is performed and the neighborhood with the best objective function value is chosen.In this research,variable neighborhood searchis appliedto enhance the quality of solutions.

Most local search heuristics use only one neighborhood structure. However, a local optimum with respect to one neighborhood structure is not necessarily a local optimum with respect to another structure. The variable neighborhood search uses different neighborhood structures and switches between them while searching for a better solution. VNS employsa systematic change of the neighbor within the search space to find the global optimum or to get a better local optima. In the proposed

MA-VNS, three different neighborhood structures are used as illustrated in Figure 3, i.e. Adjacent interchange ($\mathcal{N}_1$), Swap ($\mathcal{N}_2$) and Insertion ($\mathcal{N}_3$)neighborhood structures.

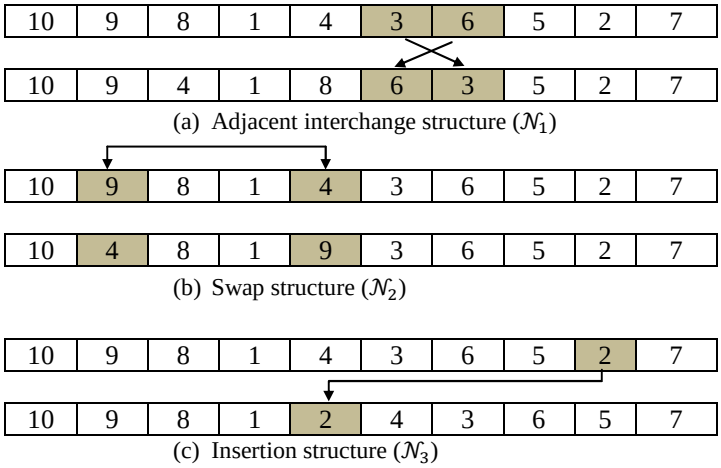

Figure 3. Neighborhood structures in the VNS

In the first neighborhood,the position of two adjacent alleles are interchanged. In the swap neighborhood structure, the position of two non-adjacent allelesare changed whereas in the insertion, one allele is removed from its position and inserted in another position.

Initially the search is performed in the first neighborhood structure to find a better solution. If a movecauses improvement in the fitness of chromosome, it is confirmed andthe search continues on theimproved chromosome. Otherwise, the search switches to the next neighborhood structure.The neighbors are examined one by one for a solution with better fitness in the neighborhood structures. The switching of neighborhoods prevents the search being stuck at the local minimum. When there is no better solution found in the first neighborhood structure, it can be a local minimum. But when the neighborhood changes, it is probable that a better solution can be found and thus the local minimum is skipped.The search terminates after a fixed number of iterations. The implementation of VNS is described as follows:

Step I: Set $x \leftarrow$ Initial solution, $t \leftarrow 0$ (Iteration number), $l \leftarrow 1$ (Structure number).
StepII: Generate a neighbor solution x_1 of x using the structure$\mathcal{N}_l$.
StepIII: Compare fitness of x_1 with x; If $fitness(x_1) \leq fitness(x)$, then $x \leftarrow x_1$; else $l \leftarrow l + 1$.
StepIV: $t \leftarrow t + 1$; If $t = t_{max}$ (the maximum number of iterations), then stop and return x;otherwise, go to step V.
Step V: If $l = 4$, then $l \leftarrow 1$, and go to Step II; else go to Step II.

4-7- Updating the population

At each generation of evolutionary algorithm, after generating new chromosomes through crossover and mutation steps the population is updated. In this research,ifthe fitness ofthe newly generated chromosome is better thanthe second parent that participated in crossover, then the second parent is replaced with this offspring. This mechanism increases the rate of improvement of the best solution. However, it reduces the diversification of the algorithm. To overcome this trap, the algorithm restarts from another region in the search space randomly.

After replacing new offspring with their parent if at least one of the new offspring ischosen to enter the population, the algorithm is continued with the updated population; otherwise,the population should be regenerated. For this purpose, the algorithm replaces all the current chromosomes with new randomly generated ones except the best elite chromosome that is obtained so far. Every time regenerating the population occurs the algorithm restarts its process from another region in the search space.

4-8- Calculating fitness of a chromosome

The fitness of a chromosome is equal to the total earliness and tardiness of all the jobs. For calculating the fitness of a chromosome, first the jobs are assigned to the batches based on the job sequence in the chromosome and then the batches are processed on themachine in order to minimize the total earliness and tardiness of all the jobs.

The first-first (FF) procedure which is commonly used for the bin-packing problem(Coffman et al., 1997), is adapted for the research problem to form the batches. The adaptive FF procedurecan be described as follows: The first unassigned job from the sequence is assigned to the first available batch with enoughresidual capacity. If none of the existing batches can accommodate the job, then a new batch is created, and the job is assigned to this new batch. This procedure is repeated until all the jobs from the sequence are assigned to a batch.

For computing the optimal schedule that minimizes the total earliness and tardiness of all the jobs for a given set of formed batches, we propose an effective dynamic programming algorithm (DP). The aim of the DP algorithm is to find a schedule of the given batches in such a way that the total earliness and tardiness is minimized.The details of the DP algorithm are described as follows.

Let $\varphi = \{B_1, B_2, \dots, B_m\}$ be a given set of formed batches, where m is the number of batches and B_b is a batch containing a subset of jobs for $b = 1,2,\dots,m$. Suppose that the batches are sorted in abatch weighted shortest processing time (BWSPT) order, i.e.,$\frac{P_1}{n_1} \le \frac{P_2}{n_2} \le \cdots \le \frac{P_m}{n_m}$, where P_b and n_b denote the processing time and the number of jobs processed in batch b, respectively.

Let $ET_b(s)$ be the minimum cost to schedule batches $b, b+1, \dots, m$ given that batches $1,2, \dots, b-1$ are scheduled and the sum of processing time of the batches that are scheduled early or on time is s. The state variable is defined only by s because by knowing$s$, the sum of processing time of the batches that are scheduled tardy can be determined. Since the processing time of all the previously scheduled batches is $\sum_{a=1}^{b-1} P_a$, the sum of processing time of the previously scheduled tardy batches is $\sum_{a=1}^{b-1} P_a - s$. The recursive relation is as follows:

$$ET_b(s) = min \left\{ n_b s + ET_{b+1}(s + P_b)\,, \qquad n_b \left(\sum_{a=1}^{b} P_a - s \right) + ET_{b+1}(s) \right\} \tag{12}$$

and the boundary conditions are:

$$ET_{m+1}(s) = 0, \quad s = 0, 1, \dots, \sum_{a=1}^{m} P_a \tag{13}$$

The optimal schedule is determined at $ET_1(0)$ and the corresponding schedule can be found by backtracking. Note that the first and second expression in the recursive relation (16) is the cost of scheduling batch b early and late, respectively.To determine the time complexity of the algorithm,note that b is $O(m)$ and s is$O(\sum_{a=1}^{m} P_a)$. Thus, the recursive relation is used $O(m \sum_{a=1}^{m} P_a)$ times. Consequently, the time complexity of the proposed DP algorithm is$O(m \sum_{a=1}^{m} P_a)$.

The steps of the proposed MA-VNS algorithm are presented as a flowchart in Figure 4.

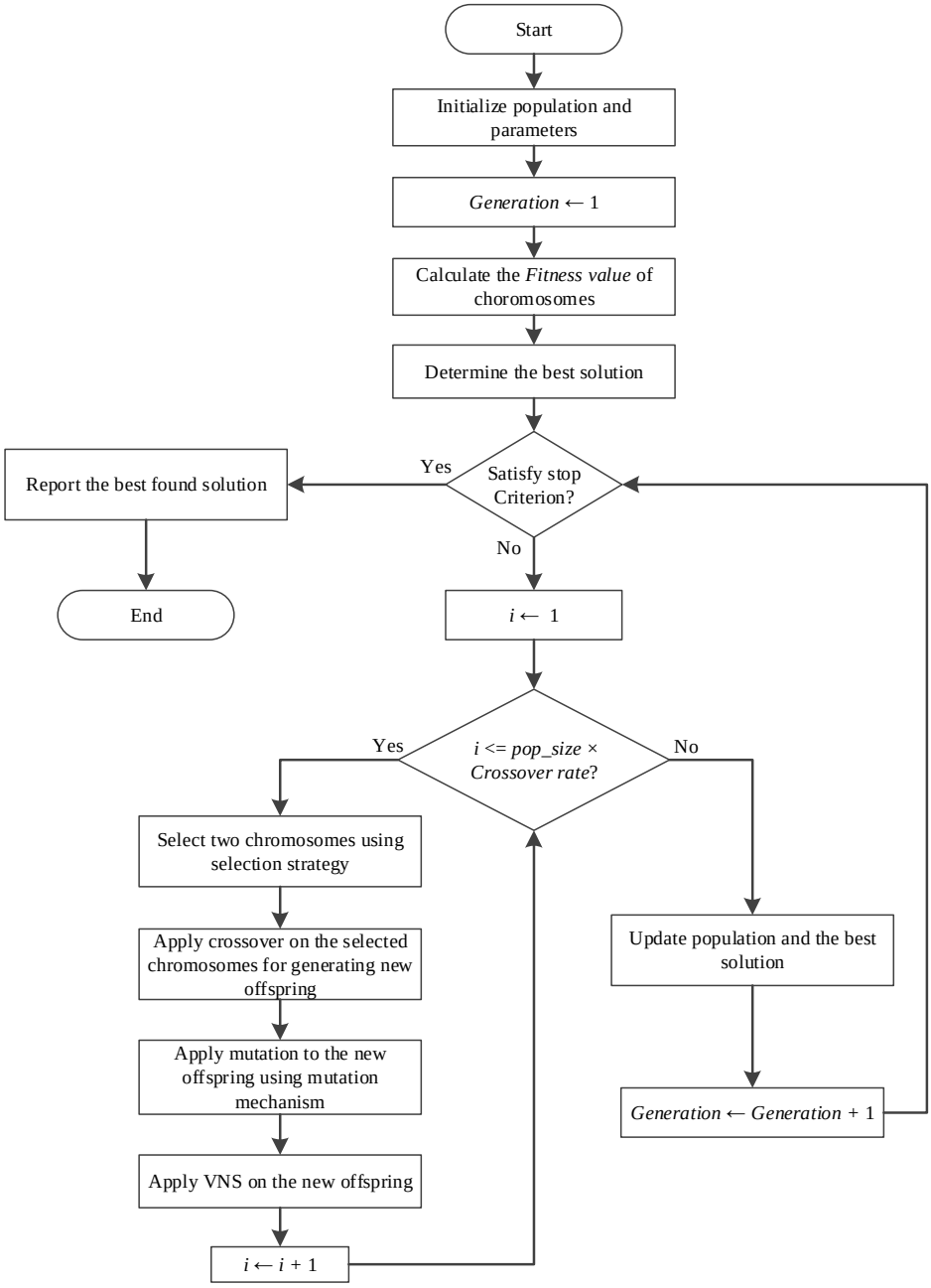

Figure 4. The flow chart of the proposed MA-VNS algorithm

5- Computational experiments

In this section, the specification of the test instances is described in the first part. Tuning the parameters of the MA-VNS algorithm is discussed in the second part. Finally, the performance of the proposed algorithm is compared with CPLEX and the best available metaheuristic algorithmin the literatureproposed by Li et al. (2015). ILOG CPLEX 12.0, a commercial optimization software, is used to solve the mathematical model. All the algorithms are coded in MATLAB 16.0 and executed on a PC with2.3 GHz processor and 2 GB RAM.

To evaluate the efficacy of the proposed algorithms, the test problems generated by Li et al. (2015) are used. Several parameters may affect the results, such as the number of jobs, the size and the processing time of jobs.The specifications of the instances are presented in Table 1.

Table 1. Specifications of the test problems

Parameters	**Levels**
Number of jobs (n)	20, 40, 60, 80, 100, 200
Job processing time (p_j)	Discrete uniform [1, 10] and Discrete uniform [1, 50]
Job sizes (s_j)	Discrete uniform [1, 30] and Discrete uniform [15, 35]
Due date (d)	$\sum_{j=1}^{n} p_j$
Machine capacity (B)	40

5-1- Parameters tuning

The performance of the MA-VNSalgorithm is generally sensitive to the setting of the parameters that influence the search mechanism and the convergence rate. A preliminary study on the performance of the MA-VNS showed that some parameters such as Population size (*pop_size*), Crossover rate ($\mathcal{C}$), Mutation rate ($\mathcal{M}$), andthe number of iterations in VNS (t_{max}) affect the performance. To tune the proper value for these parameters, we conducted a statistical analysis based on Taguchi method (Taguchi, 1986). Taguchi method attempts to increase the robustness of the system by minimizing the variation of the output results. Several computational experiments are conducted for determining the proper values of the parameter levels. Consequently, there are four main factors and three levels for each parameter values. The factors and their levels are presented in details in Table 2.

Table 2. Levels of main factors

A: Population size (*pop_size*)	B: Crossover rate ($\mathcal{C}$)	C: Mutation rate ($\mathcal{M}$)	D: Number of iterations in VNS (t_{max})
A(1): 20	B(1): 0.2	C(1): 0.01	D(1): 10
A(2): 50	B(2): 0.4	C(2): 0.1	D(2): 20
A(3): 100	B(3): 0.8	C(3): 0.5	D(3): 30

The orthogonal array $L_9(3^4)$ is the best fittest design for factors' levels. 20 randomly generated test instances are solved by all 9 predefined scenarios, shown in Table 3. Totally, 180 runs should be executed for the design. The response variable of the design is calculated based on the objective function of the MA-VNS algorithm.

Table 3. Factors levels of orthogonal array$L_9(3^4)$

Scenario no.	Factors levels			
	A	B	C	D
1	A(1)	B(1)	C(1)	D(1)
2	A(1)	B(2)	C(2)	D(2)
3	A(1)	B(3)	C(3)	D(3)
4	A(2)	B(1)	C(2)	D(3)
5	A(2)	B(2)	C(3)	D(1)
6	A(2)	B(3)	C(1)	D(2)
7	A(3)	B(1)	C(3)	D(2)
8	A(3)	B(2)	C(1)	D(3)
9	A(3)	B(3)	C(2)	D(1)

The robustness of the algorithm is measured by the means of Signal-to-Noise (S/N) ratio. The S/N ratio is computed as $-10\,log(ET)^2$. For minimization problems, the larger value of S/N ratio leads to the smaller variation of the output results. Since the test instances have different sizes and therefore

different objective functions, the size effect should be eliminated. So, the objective function value of each test instance is changed to the relative percentage deviation (RPD). RPD is calculated as $(ET - ET_{best})/\,(ET_{worst} - ET_{best})$, where ET is the total earliness and tardiness value obtained from the MA-VNS algorithm for the test instance. ET_{best}andET_{worst} are respectively the best and worst total earliness and tardinessvalue obtained for the test instance from different scenarios. Therefore, the S/N ratio for each scenario is calculated by $(S/N)_r = -10\log\left(\frac{1}{20}\sum_{i=1}^{20} RPD_{ir}\right)^2, r = 1,2,\dots,9$.

Figure5represents the results of Taguchi experiment. For each factor, the best level is the one with smaller RPD and higher S/N ratio. Based on the Taguchi results, the best levels are A(2), B(3), C(2), and D(3). The best setting of parameters is given in Table 4.

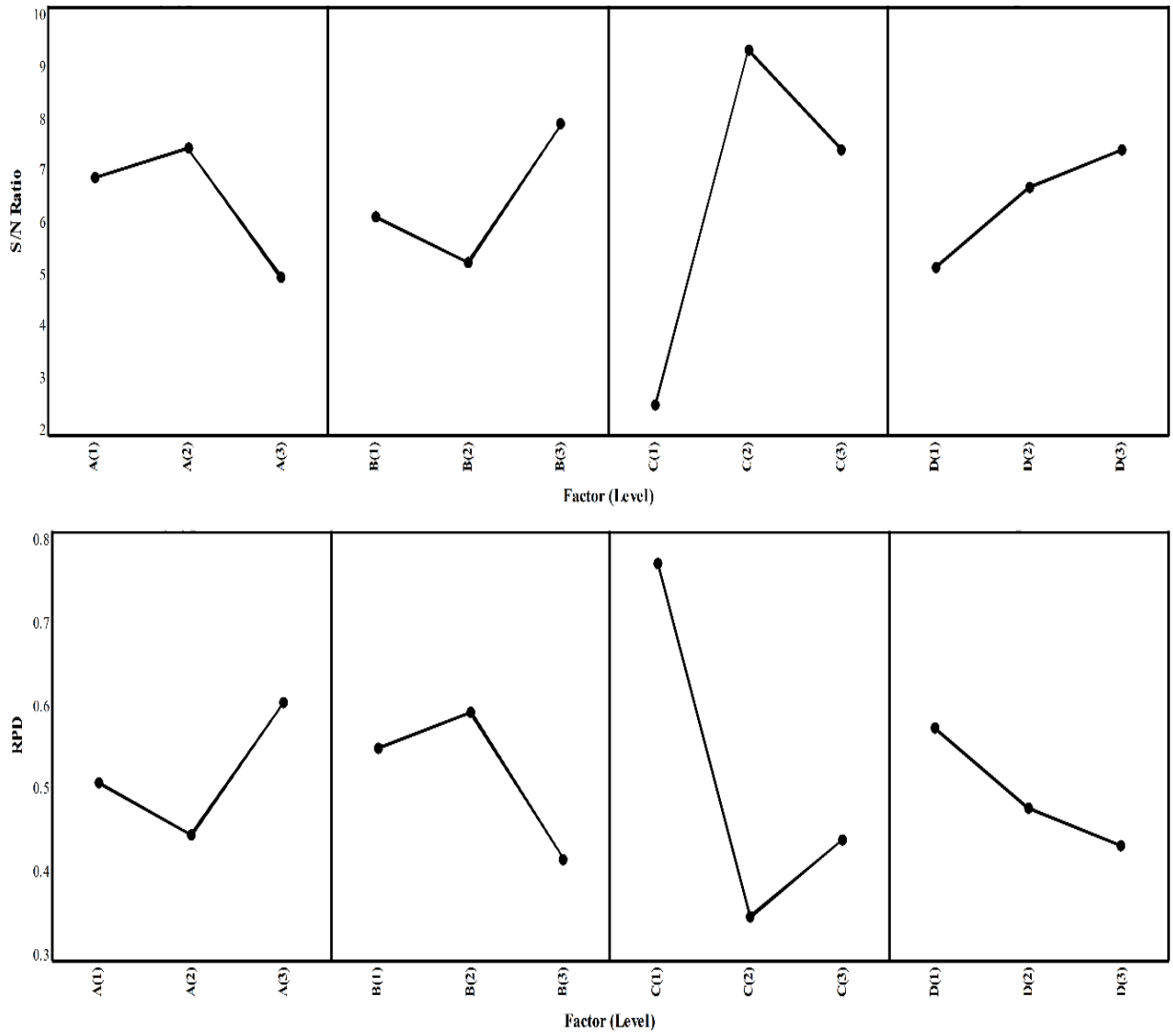

Figure 5. The results of Taguchi experiment

Table 4. Taguchi results: selected levels of factors

A: Population size (pop_size)	B: Crossover rate ($\mathcal{C}$)	C: Mutation rate ($\mathcal{M}$)	D: Number of iterations in VNS (t_{max})
A(2): 50	B(3): 0.8	C(2): 0.1	D(3): 30

5-2- Evaluation of BMILP model and MA-VNS algorithm on small-sized instances

In this section, the results of the proposed algorithm are compared with the solutions of the binary mixed integer modelpresented in Section 3. The model is solved by using CPLEX. Since the research problem is NP-hard, the optimal solution for medium and large sized problems cannot be found in a reasonable time. CPLEX did not converge even after running for several hours, so it was terminated after 24 hours and the best found solution is used for comparison.

Problems with $n = 10$, 20, and 40 are considered and for each level of n, a random problem instance in each category of job sizes and job processing times are generated.The number of variables and the number of constraints in the mathematical model for different number of jobs are presented in Table 5.

Table 5. Number of variables and constraints in the mathematical model

No. jobs	No. Binary variables	No. Total variables	No. Constraints
10	100	140	250
20	400	480	900
40	1600	1760	3400
n	n^2	$n^2 + 4n$	$2n^2 + 5n$

The results obtained from MA-VNS and CPLEX are presented in Table6.The last column represents the optimality gap (GAP) of CPLEX. A large GAP implies that CPLEX requires much more time to converge to an optimal solution.

Table 6. Comparison of MA-VNS and CPLEX

p_j	s_j	n	MA-VNS/CPLEX	Gap (%)
$U[1,10]$	$U[1,30]$	10	1.0000	0.00
		20	0.9469	4.23
		40	0.9472	7.38
	$U[15,35]$	10	1.0000	0.00
		20	0.9669	3.62
		40	0.9175	8.59
$U[1,50]$	$U[1,30]$	10	1.0000	0.00
		20	1.0000	4.71
		40	0.9151	12.01
	$U[15,35]$	10	1.0000	0.00
		20	0.9299	13.48
		40	0.9491	9.36

As can be seen from Table 6, MA-VNS and CPLEX find the optimal solutions for all the instances with 10 jobs. However, the computational time required by MA-VNS is extremely less than CPLEX. For the instances with 20 and 40 jobs, CPLEX could not find the optimal solution within 24 hours. The optimality gap infers that CPLEX needs more time to converge to an optimal solution. For these instances, the solutions obtained from MA-VNS are equal or better than the best solutions reported by CPLEX. This results demonstrate the verification of the proposed algorithm.

5-3- Evaluation of the MA-VNS algorithm

For evaluating the efficiency of MA-VNS on large-sized instances, it has been compared with the proposed algorithm developed by Li et al. (2015), called GAMARB. For each combination of the number of jobs, job sizes, and job processing times,10 instances are tested, for a total of240. All the

instances are solved by each of the algorithms. Li et al. (2015)used 1000 generation as the stopping criterion of GAMARB,we also use the same stopping criterion to fairly compare the algorithms. The performance comparison between MA-VNS and GAMARB is representedin Table 7.

Table 7. Comparing solution quality of MA-VNS and GAMARB

p_j	s_j	n	MA-VNS	GAMARB	MA-VNS/ GAMARB
$U[1,10]$	$U[1,30]$	20	160.5	163.9	0.9795
		40	561.8	584.8	0.9607
		60	1205.3	1291.7	0.9331
		80	2124.0	2340.9	0.9074
		100	3534.6	3968.3	0.8907
		200	13730.6	15523.6	0.8845
				Average	**0.9260**
	$U[15,35]$	20	296.2	300.3	0.9864
		40	1201.5	1220.6	0.9844
		60	2525.6	2573.2	0.9815
		80	4294.6	4376.4	0.9813
		100	7058.3	7193.4	0.9812
		200	27159.7	27762.1	0.9783
				Average	**0.9822**
$U[1,50]$	$U[1,30]$	20	656.6	675.0	0.9728
		40	2716.4	2841.2	0.9561
		60	5813.0	6219.3	0.9347
		80	10072.4	10859.8	0.9275
		100	15202.2	16520.4	0.9202
		200	61873.3	67275.5	0.9197
				Average	**0.9385**
	$U[15,35]$	20	1303.6	1316.1	0.9905
		40	5409.3	5477.0	0.9876
		60	11865.9	12023.4	0.9869
		80	18812.0	19114.3	0.9842
		100	31972.3	32565.0	0.9818
		200	127355.6	129968.0	0.9799
				Average	**0.9852**

From Table 7, it can be observed that MA-VNS has a better performance compared to GAMARBin all categories of problem instances. The performance of the proposed algorithm for problems with small job sizes is more significant than the case of large job sizes. For the problems with large job sizes, two algorithms have almost the same performance and obtain solutions with similar quality. Since when the size of jobs are larger, more jobs are processed individually, and thus the feasible search space is much smaller compared to the case of small job sizes. This leads to a reduction in the performance differences between two algorithms. The performance comparison between MA-VNS and GAMARB for different job processing time shows that, job processing time has no significant impact on the solution quality. It also can be observed that the performance differences between two algorithms become more considerable by increasing the number of jobs. The results for different categories of job processing times and job sizes are depictedin Figures 6 and 7.

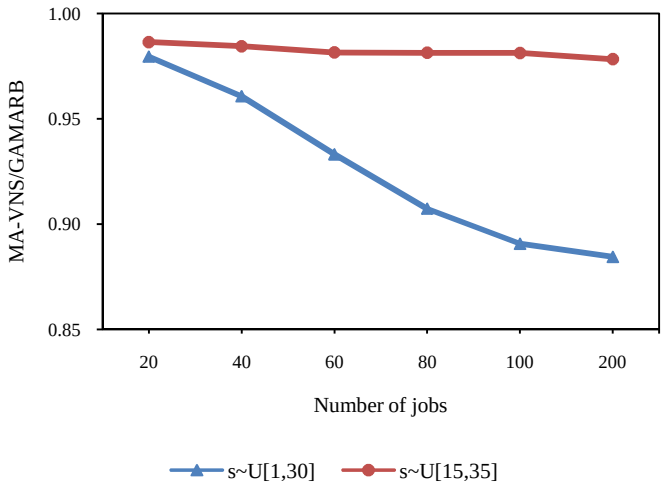

Figure 6. Comparison theperformance of algorithms for instances with $\boldsymbol{p_j \sim U[1, 10]}$

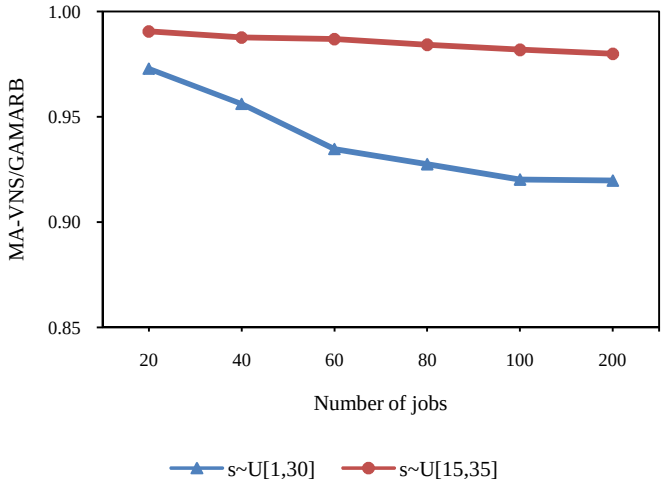

Figure 7. Comparison theperformance of algorithms for instances with $\boldsymbol{p_j \sim U[1, 50]}$

The average computational time of MA-VNS and GAMARB is shown in Table 8. The results show that there is no significant difference between two algorithms in terms of computational time.The two algorithms need more time as the number of jobs increases. Finally, we can conclude that MA-VNS provides better solutions than GAMARB but need approximately the same computational time.

Table 8. Computational time for the MA-VNS and GAMARB

p_j	n	$s_j \sim U[1,30]$		$s_j \sim U[15,35]$	
		MA-VNS	GAMARB	MA-VNS	GAMARB
$U[1,10]$	20	1.6	1.8	2.3	2.1
	40	8.1	7.9	11.4	10.2
	60	16.4	17.6	17.6	19.7
	80	30.8	31.6	32.8	33.0
	100	25.4	27.6	37.5	35.2
	200	60.12	57.4	85.6	81.8
$U[1,50]$	20	2.7	2.1	2.0	2.3
	40	10.4	9.4	10.2	9
	60	18.7	20.3	22.3	24.8
	80	27.8	26.2	42.7	39.1
	100	55.7	52.9	54.2	53
	200	94.2	92.5	97.6	98.8

6- Conclusions

The minimization of total earliness and tardiness on a single batch processing machine is investigated in this research. A mathematical model for the research problem is proposed. An improved memetic algorithm, which combines evolutionary algorithmswith variable neighborhood search procedure is presented.Computational experiments show that theproposed MA-VNS has a superior performance than the best available metaheuristic algorithm existing inthe literature i.e., the GAMARB algorithm byLi et al. (2015).Developinglower boundingmethods for evaluating the performance of metaheuristic algorithmsisan interesting extension and opportunity for the future research.It's also possible to relax the common due date assumption. Consideringminimization of earliness and tardiness on batch processing systems in theother machine environments such as flowshop and parallel machines, is another possible future research topic.

References:

Al-Salamah, M. (2015). Constrained binary artificial bee colony to minimize the makespan for single machine batch processing with non-identical job sizes. *Applied Soft Computing*, 29, 379-385.

Brucker, P., Gladky, A., Hoogeveen, H., Kovalyov, M.Y., Potts, C., Tautenhahn, T. & Van De Velde, S. (1998). Scheduling a batch machine. *Journal of Scheduling*, 1, 31–54.

Cabo, M., Possani, E., Potts, C.N. & Song, X. (2015). Split–merge: Using exponential neighborhood search for scheduling a batching machine. *Computers & Operations Research*, 63, 125-135.

Chen, H., Du, B. & Huang, G.Q. (2011). Scheduling a batch processing machine with non-identical job sizes: a clustering perspective. *International Journal of Production Research*, 49(19), 5755-5778.

Cheng, B., Li, K. & Chen, B. (2010). Scheduling a single batch-processing machine with non-identical job sizes in fuzzy environment using an improved ant colony optimization. *Journal of Manufacturing Systems*, 29(1), 29-34.

Coffman, E.G., Garey, M.R. & Johnson, D.S. (1997). *Approximation algorithms for bin packing: a survey*. In: S.H. Dorit, Approximation algorithms for NP-hard problems (pp. 46-93): PWS Publishing Co.

Damodaran, P., Ghrayeb, O. & Guttikonda, M.C. (2013). GRASP to minimize makespan for a capacitated batch-processing machine. *The International Journal of Advanced Manufacturing Technology,* 68(1-4), 407-414.

Dupont, L. & Dhaenens-Flipo, C. (2002). Minimizing the makespan on a batch machine with non-identical job sizes: an exact procedure. *Computers & Operations Research,* 29(7), 807-819.

Dupont, L. & Jolai, F. (1998). Minimizing makespan on a single batch processing machine with non-identical job sizes. *European Journal of Automation Systems,* 32, 431-440.

Graham, R.L., Lawler, E.L., Lenstra, J.K. & Kan, A. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of discrete Mathematics,* 5, 287-326.

Hart, W.E., Krasnogor, N. & Smith, J.E. (2005). *Recent advances in memetic algorithms* (Vol. 166): Springer Verlag.

Jia, Z.-h. & Leung, J.Y.T. (2014). An improved meta-heuristic for makespan minimization of a single batch machine with non-identical job sizes. *Computers & Operations Research,* 46(0), 49-58.

Jolai, F. & Dupont, L. (1998). Minimizing mean flow times criteria on a single batch processing machine with non-identical jobs sizes. *International Journal of Production Economics,* 55(3), 273-280.

Li, Z., Chen, H., Xu, R. & Li, X. (2015). Earliness–tardiness minimization on scheduling a batch processing machine with non-identical job sizes. *Computers & Industrial Engineering,* 87, 590-599.

Malapert, A., Gueret, C. & Rousseau, L.-M. (2012). A constraint programming approach for a batch processing problem with non-identical job sizes. *European Journal of Operational Research,* 221(3), 533-545.

Mathirajan, M. & Sivakumar, A.I. (2006). A literature review, classification and simple meta-analysis on scheduling of batch processors in semiconductor. *The International Journal of Advanced Manufacturing Technology,* 29(9-10), 990-1001.

Mönch, L., Fowler, J.W., Dauzère-Pérès, S., Mason, S.J. & Rose, O. (2011). A survey of problems, solution techniques, and future challenges in scheduling semiconductor manufacturing operations. *Journal of Scheduling,* 14(6), 583-599.

Mönch, L. & Unbehaun, R. (2007). Decomposition heuristics for minimizing earliness–tardiness on parallel burn-in ovens with a common due date. *Computers & Operations Research,* 34(11), 3380-3396.

Mönch, L., Unbehaun, R. & Choung, Y.I. (2006). Minimizing earliness–tardiness on a single burn-in oven with a common due date and maximum allowable tardiness constraint. *OR Spectrum,* 28(2), 177-198.

Moscato, P. (1989). On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. *Caltech Concurrent Computation Program, C3P Report,* 826.

Potts, C.N. & Kovalyov, M.Y. (2000). Scheduling with batching: a review. *European Journal of Operational Research,* 120(2), 228-249.

Qi, X. & Tu, F. (1999). Earliness and tardiness scheduling problems on a batch processor. *Discrete Applied Mathematics,* 98(1), 131-145.

Rafiee Parsa, N., Karimi, B. & Husseinzadeh Kashan, A. (2010). A branch and price algorithm to minimize makespan on a single batch processing machine with non-identical job sizes. *Computers & Operations Research*, 37(10), 1720-1730.

Taguchi, G. (1986). *Introduction to quality engineering: designing quality into products and processes*: Asian productivity organization.

Uzsoy, R. (1994). Scheduling a single batch processing machine with non-identical job sizes. *International Journal of Production Research*, 32(7), 1615-1635.

Wang, H.-M. (2011). Solving single batch-processing machine problems using an iterated heuristic. *International Journal of Production Research*, 49(14), 4245-4261.

Xu, R., Chen, H. & Li, X. (2012). Makespan minimization on single batch-processing machine via ant colony optimization. *Computers & Operations Research*, 39(3), 582-593.

Zhao, H., Hu, F. & Li, G. (2006). *Batch scheduling with a common due window on a single machine*. In: Fuzzy Systems and Knowledge Discovery (pp. 641-645): Springer.

Zhou, S., Chen, H., Xu, R. & Li, X. (2014). Minimising makespan on a single batch processing machine with dynamic job arrivals and non-identical job sizes. *International Journal of Production Research*, 52(8), 2258-2274.