Journal of Industrial and Systems Engineering
Vol. 13, No. 2, pp. 9-26
Spring (April) 2021

JISE

A lower bounding method for earliness and tardiness minimization on a single batch processing machine

Taha Keshavarz[1*]

[1]*Department of Industrial Engineering, Semnan University, Semnan, Iran*

taha_keshavarz@semnan.ac.ir

Abstract

In this research, the problem of scheduling a single batch processing machine with non-identical job sizes is considered. The objective is to minimize the total earliness and tardiness of all the jobs. A batch processing machine can process a group of jobs simultaneously as a batch as long as its capacity is not violated. The processing time of a batch is equal to the maximum processing time of all the jobs in the batch. Since the problem under study is shown to be NP-hard, a lower bounding method based on column generation is proposed. The proposed lower bound can be used for evaluating the performance of the heuristic and metaheuristic algorithms developed for the research problem. The computational experiments are designed to analyze the performance of the proposed lower bound. The results show that the column generation approach can considerably generates better lower bound than the best known lower bounding method in the literature.

Keywords: Batch processing machine, just-in-time, lower bound, column generation

1-Introduction

Batch processing systems are one of the most important production systems in practice. A batch processing machine (BPM) can process several jobs simultaneously. A batch of jobs referred to a group of jobs that are processed together on the machine. All the jobs in a batch have the same start and completion time on the machine. In parallel BPM, 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. Chemical processes performed in tanks or kilns and burn-in oven operations in semiconductor industries are the examples of such problems. A comprehensive literature review of scheduling semiconductor manufacturing operations is performed by Mönch et al. (2011).

The most studies on BPM are related to its application in semiconductor manufacturing. Heat-stress test of integrated circuit chips are done by using Burn-in oven machines. Burn-in ovens are batch processing machine and 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, efficient scheduling on burn-in ovens can considerably reduce the cost of production in semiconductor manufacturing. In high-tech manufacturing such as semiconductor manufacturing where the product life cycles are short, developing appropriate strategies to reduce the production costs is so essential.

Just-in-time (JIT) scheduling is another popular strategy for improving the efficiency and cost-effectiveness of a production system. A JIT strategy improves a production system by reducing in-process inventories.

*Corresponding author
ISSN: 1735-8272,

The earliness and tardiness penalties are 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. 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. In other words, two important classes of scheduling problems, BPM scheduling problems and JIT scheduling problems are investigated in this research. 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 wafers are 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 by Brucker et al. (1998), there are a few research that propose effective algorithms for the research problem in the literature. Some researchers proposed several heuristic and metaheuristic for the research problem. However, there is not a good lower bound on the optimal solution of the problem. To evaluate the performance of the heuristic and metaheuristic algorithms, a tight lower bound can be very helpful. This is a motivation to focus on developing a lower bounding method based on column generation approach for the research problem.

The rest of the paper is organized as follows: Related literature to the research problem is briefly provided in section 2. The characteristics of the problem and a mathematical formulation are presented in section 3. The details of the proposed column generation approach are introduced in section 4. The experimental design to evaluate the effectiveness of the proposed approach is reported in section 5. Finally, conclusions and directions for the future research are discussed in section 6.

2-Related literature

In recent years, scheduling problems related to the batch processing machines have been investigated extensively by many researchers. In this section, only the papers having the most similarities with our assumptions are reviewed; especially the papers investigating the single batch processing machine problem with non-identical job sizes.

Uzsoy (1994) investigated the single batch processing machine problem with non-identical job sizes and gave the complexity results for both makespan minimization and total flow time minimization for the first time. He developed some heuristics and a branch and bound algorithm for the research problem and showed that these procedures provide near optimal solutions. Several heuristics were proposed by Jolai and Dupont (1998) for minimizing the total flow time. The same problem was considered by Dupont and Jolai (1998) and various heuristics were developed to minimize makespan. Their computational experiments revealed the performance of these heuristics. A branch and bound algorithm was developed by Azizoglu and Webster (2000) to minimize the total weighted flow time on a single batch processing machine. They showed that the problem instances with less than 25 jobs can be solved in a reasonable amount of time.

Dupont and Dhaenens-Flipo (2002) considered the problem of makespan minimization on a single BPM and proposed a branch and bound procedure. Rafiee Parsa et al. (2010) proposed a column generation approach and a branch and price algorithm for the same problem. They showed that the branch and price algorithm has a better performance than the proposed algorithm by Dupont and Dhaenens-Flipo (2002). Chen et al. (2011) introduced definition of waste ratio of a batch and proposed a clustering algorithm to minimize makespan. They showed that their algorithm outperforms the previous algorithms. Lee and Lee (2013) developed construction-based heuristics and improvement-based heuristics to minimize the makespan. They showed that their suggested heuristics produce high quality solutions in most cases. S. Li et al. (2005) investigated the problem with job release times and provided an approximation algorithm for minimizing makespan. They proved that the worst-case ratio of the proposed approximation algorithm is $2+\varepsilon$. Zhou et al. (2014) considered the problem with dynamic job arrivals and proposed constructive heuristics for this problem.

Wang (2011) proposed a two-phase heuristic for the problem of minimizing total weighted tardiness. Malapert et al. (2012) developed a constraint programming approach to minimize the

maximum lateness. They decomposed the problem into two sub-problems. In the first sub-problem, an assignment of the jobs to the batches was determined. Then, minimizing the lateness of the batches was considered in the second sub-problem. A new neighborhood search algorithm to minimize the maximum lateness of the jobs was introduced by Cabo et al. (2015). Their computational experiments show the effectiveness of the proposed neighborhood search algorithm.

There are several research efforts focused on developing various metaheuristic algorithms for the single batch processing machine. A simulated annealing algorithm for single BPM problem with non-identical job sizes was proposed by Melouk et al. (2004) to minimize the makespan. Kashan et al. (2006) developed two different genetic algorithms based on different representation schemes. They showed that the performance of the proposed algorithms have a superior performance than the simulated annealing approach proposed by Melouk et al. (2004). Investigating the bi-criteria scheduling problem of minimizing the makespan and maximum tardiness was considered by Kashan et al. (2010). They proposed two multi-objective genetic algorithms based on different encoding schemes. Xu et al. (2012) proposed an ant colony optimization algorithm for the single BPM problem with dynamic job arrivals to minimize the makespan. They introduced a definition of waste and idle space for batches. Damodaran et al. (2013) developed a Greedy Randomized Adaptive Search Procedure to minimize the makespan. They compared their proposed approach with the algorithms proposed by Melouk et al. (2004) and Damodaran et al. (2006). Jia and Leung (2014) formulated makespan minimization as a problem of minimizing the wasted space and presented an improved ant system algorithm. Al-Salamah (2015) proposed an artificial bee colony approach to minimize the makespan. Rafiee Parsa et al. (2016) developed a max-min ant system to minimize total completion time. Recently, Rafiee Parsa et al. (2019) designed a hybrid neural network approach for the same problem.

Considering non-regular objective functions, Brucker et al. (1998) proved that all the batch scheduling problems with due date related criteria are NP-hard. Hence, the minimization of earliness and tardiness is also NP-hard. Qi and Tu (1999) proposed a dynamic programming algorithm for the earliness and tardiness minimization problem. They considered identical job sizes and the same processing times for all the jobs and assumed jobs have a distinct due date. Mönch et al. (2006) considered the earliness and tardiness minimization problem with a common due date and identical job sizes under a maximum allowable tardiness constraint. They proposed a hybrid genetic algorithm for the problem. Zhao et al. (2006) developed a polynomial time heuristic algorithm for minimizing the total weighted earliness and tardiness when there is a common due window for the jobs. Several heuristic algorithms were developed to minimize the earliness and tardiness on parallel burn-in ovens by Mönch and Unbehaun (2007). They assumed that jobs have unit job size and a common due date. Z. Li et al. (2015) considered the case of non-identical job sizes and proposed a hybrid genetic algorithm for the problem with a common due date. Polyakovskiy et al. (2017) investigated the JIT batch scheduling problem with two dimensional bin packing constraints. They proposed a constraint programming based heuristic and an agent based modeling heuristic. Rafiee Parsa et al. (2017) proposed exact and heuristic algorithms for JIT scheduling in a batch processing system. They considered a common and loose due date and developed a mixed integer linear programming for the problem. They also proposed several heuristics and an exact algorithm. Ogun and Alabas-Uslu (2018) developed three different mathematical models for minimization of total earliness and tardiness of customer orders to provide on-time completion of customer orders and also, to avoid excess final product inventory. Jia et al. (2020) developed a new history-guided multi-objective evolutionary algorithm for a multi-objective scheduling problem on parallel batching machines with three objectives, the minimization of the makespan, the total weighted earliness/tardiness penalty and the total energy consumption, simultaneously.

Concluding from the literature review, there are a few research that considered the single BPM problem with minimization of the total earliness and tardiness with non-identical job sizes. In addition, there is no efficient lower bounding method for this problem. The valid and tight lower bounds can be used for evaluating different heuristic and metaheuristic algorithms developed for the research problem. Investigating on this problem is both academically interesting and practically important. In addition, the results can provide insights that can be used to solve more complicated batch scheduling problems more effectively.

3-Problem description

Consider a single batch processing machine with n jobs to be processed. Each job j is available at time 0 and has a processing time (p_j) and a corresponding size (s_j). The capacity of batch processing machine is C and the sum of the size of jobs in each batch cannot exceed C. Processing of a batch cannot be interrupted after it is started, and other jobs cannot be inserted into the machine until processing is completed. The processing time of a batch is given by the longest processing time among the jobs within the batch. In other words, the batch processing machine is considered to be a parallel batch processing machine. The completion time of the jobs in a batch is equal to the completion time of the batch. 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 goal is to find the best schedule of jobs in order to minimize the total earliness and tardiness. Based on the standard classification scheme for scheduling problems (Graham et al., 1979), the problem can be noted as $1|p-batch, s_j \leq C|\sum(E_j + T_j)$. 1 refers to single machine scheduling, $p-batch$ denotes the parallel batch processing assumptions, $s_j \leq C$ indicates that there are non-identical job sizes and the machine capacity is C, and $\sum(E_j + T_j)$ demonstrates the objective function. Rafiee Parsa et al. (2017) proposed a binary mixed integer linear programming model for the research problem. The parameters, decision variables and the mathematical model are as follows:

Parameters and notations:

n : Number of jobs

j : Index used for jobs

b : Index used for batches

s_j : Size of job j

p_j : Processing time of job j

C : Capacity of machine

d : Common loose due date

M : A large positive number

Decision variables:

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

Dependent variables:

P_b : Processing time of batch b

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

The model:

$$\text{Minimize} \quad \sum_{j=1}^{n} ET_j \tag{1}$$

Subject to:

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

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

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

$$C_1 \geq P_1 \qquad (5)$$

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

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

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

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

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

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

Minimizing the 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 positive number. 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-Lower bounding method – the column generation approach

The column generation approach has been known as an efficient method for solving linear and integer programming problems with huge number of variables. The details of the column generation approach can be found in Barnhart et al. (1998), Wilhelm (2001), and Wilhelm et al. (2003).

In this section, a column generation approach is proposed for finding the lower bound of the research problem. The reformulated mathematical model, and the details of the column generation approach are discussed in the following sections.

4-1-Dantzig-Wolf decomposition model

A decomposition formulation for the research problem by reformulating it as a set-partitioning Master Problem by using Dantzig–Wolfe decomposition is proposed. In the set partitioning formulation, m positions are considered for scheduling the batches and each column corresponds to a set of jobs assigned to a batch that placed in a defined position. Since the number of batches may be less than the number of positions, it is assumed that empty batches are placed in the positions with no batch. The following parameters and decision variables beside the defined ones in Section 3 are required to present this model:

Parameters:

B_i	$i = 1, \dots, m$	The set of all feasible batches in position i
K_i	$i = 1, \dots, m$	Number of all feasible batches in position i, i.e. $\lvert B_i \rvert$
b_i^k	$i = 1, \dots, m; k = 1, \dots, K_i$	The k^{th} feasible batch in position i
P_i^k	$i = 1, \dots, m;\ k = 1, \dots, K_i$	The processing time of the k^{th} feasible batch in position i
C_i^k	$i = 1, \dots, m;\ k = 1, \dots, K_i$	The completion time of the k^{th} feasible batch in position i
E_i^k	$i = 1, \dots, m;\ k = 1, \dots, K_i$	The earliness of the k^{th} feasible batch in position i
T_i^k	$i = 1, \dots, m;\ k = 1, \dots, K_i$	The Tardiness of the k^{th} feasible batch in position i

$$x_{ij}^k = \begin{cases} 1 & \text{If job } j \text{ is assigned to the } k^{\text{th}} \text{ feasible batch in position } i \\ 0 & \text{Otherwise} \end{cases} \qquad \begin{matrix} i = 1, \dots, m;\ k = 1, \dots, K_i \\ j = 1, \dots, n \end{matrix}$$

Decision variables:

$$\lambda_i^k = \begin{cases} 1 & \text{If batch } b_i^k \in B_i \text{ is selected in posiotion } i \\ 0 & Otherwise \end{cases}$$

P_i	$i = 1, \dots, m$	The processing time of the batch in position i
C_i	$i = 1, \dots, m$	The completion time of the batch in position i
E_i	$i = 1, \dots, m$	The earliness of the batch in position i
T_i	$i = 1, \dots, m$	The tardiness of the batch in position i

Each column corresponds to a feasible batch in a position. A feasible batch b_i^k is defined by the jobs belonging to it. In other words, $b_i^k = \{x_{ij}^k, i = 1, \dots, m;\ j = 1, \dots, n\}$ where x_{ij}^k define a feasible assignment of jobs to a batch by considering the machine capacity. The master problem can be written as follows:

$$\text{Min } Z^{MP} = \sum_{k=1}^{K_m} \sum_{i=1}^{m} \sum_{j=1}^{n} \lambda_i^k x_{ij}^k \left(E_i^k + T_i^k\right) \tag{12}$$

$$s.t.$$

$$\sum_{k=1}^{K_i} \lambda_i^k C_i^k \geq \sum_{k=1}^{K_{(i-1)}} \lambda_{(i-1)}^k C_{(i-1)}^k + \sum_{k=1}^{K_i} \lambda_i^k P_i^k \qquad i = 2, \dots, m \tag{13}$$

$$\sum_{k=1}^{K_i}\sum_{i=1}^{m} x_{ij}^{k}\lambda_i^k = 1 \qquad j = 1,\dots,n \qquad (14)$$

$$\sum_{k=1}^{K_i} \lambda_i^k = 1 \qquad i = 1,\dots,m \qquad (15)$$

$$\lambda_i^k \in \{0,1\} \qquad i = 1,\dots,m;\ k = 1,\dots,K_i \qquad (16)$$

Minimization of total earliness and tardiness as the objective function is presented by equation (12). Constraint set (13) ensures that the completion time of a batch in position i is greater than or equal to the completion time of the batch in position $(i-1)$ plus its processing. This constraint set deals with the completion time of two consecutive batches. Constraint set (14) ensures that each job is assigned exactly to one batch selected for one position. Constraint set (15) is the convexity constraint and ensure that only one batch is selected for each position.

There are so many feasible batches for each position, so explicitly considering all of them into the model is not possible. Therefore, column generation approach is used to solve the linear programming relaxation of the master problem by adding new batches with negative reduced costs to the model as needed. Initial batches are generated by using the ETFF-LPT algorithm proposed by Rafiee Parsa et al. (2017) for the research problem. The linear programming master problem, which includes only a subset of all possible batches, is called the restricted linear master problem (RLMP) and can be written as below:

$$\text{Min } Z^{RLMP} = \sum_{k=1}^{K_m}\sum_{i=1}^{m}\sum_{j=1}^{n} \lambda_i^k x_{ij}^k \left(E_i^k + T_i^k\right) \qquad (17)$$

$s.t.$

$$\sum_{k=1}^{K_i} \lambda_i^k C_i^k \geq \sum_{k=1}^{K_{(i-1)}} \lambda_{(i-1)}^k C_{(i-1)}^k + \sum_{k=1}^{K_i} \lambda_i^k P_i^k \qquad i = 2,\dots,m \qquad [\gamma_i] \qquad (18)$$

$$\sum_{k=1}^{K_i}\sum_{i=1}^{m} x_{ij}^{k}\lambda_i^k = 1 \qquad j = 1,\dots,n \qquad [\beta_j] \qquad (19)$$

$$\sum_{k=1}^{K_i} \lambda_i^k = 1 \qquad i = 1,\dots,m \qquad [\alpha_i] \qquad (20)$$

$$0 \leq \lambda_i^k \leq 1 \qquad i = 1,\dots,m;\ k = 1,\dots,K_i \qquad (21)$$

Define γ_i, β_j, and α_i as the dual variables corresponding to constraint sets (18), (19), and (20), respectively. The dual problem of RLMP is presented to facilitate the presentation of the sub-problems. Sub-problems are used to identify if there is any column to add to RLMP to improve the objective function value. The dual of RLMP is generated as the following:

$$\text{Max } Z^{Dual} = \sum_{i=1}^{m} \alpha_i + \sum_{j=1}^{n} \beta_j \qquad (22)$$

$s.t.$

$$-\gamma_2 C_1^k + \alpha_1 + \sum_{j=1}^{n} x_{1j}^k \beta_j \leq \sum_{j=1}^{n} x_{1j}^k \left(E_1^k + T_1^k\right) \qquad k = 1,\dots,K_1 \qquad (23)$$

$$\gamma_i\left(C_i^k - P_i^k\right) - \gamma_{i+1}C_i^k + \alpha_i + \sum_{j=1}^{n} x_{ij}^k \beta_j \le \sum_{j=1}^{n} x_{ij}^k \left(E_i^k + T_i^k\right) \quad (24)$$

$$i = 2, \dots, m-1;\ k = 1, \dots, K_i$$

$$\gamma_m\left(C_m^k - P_m^k\right) + \alpha_m + \sum_{j=1}^{n} x_{mj}^k \beta_j \le \sum_{j=1}^{n} x_{mj}^k \left(E_m^k + T_m^k\right) \quad (25)$$

$$k = 1, \dots, K_m$$

$$\gamma_i \ge 0 \qquad i = 2, \dots, m \quad (26)$$

$$\alpha_i, \beta_j \text{ unresterticted} \qquad i = 1, \dots, m; j = 1, \dots, n \quad (27)$$

To simplify representing the sub-problems, define $\gamma_1 = 0$ and $\gamma_{m+1} = 0$. In order To find the batch with the most negative reduced cost in the i^{th} position $(i = 1, \dots, m)$, the value of $\underset{1 \le k \le K_i}{Min} (\gamma_{i+1} - \gamma_i)C_i^k + \gamma_i P_i - \sum_{j=1}^{n} x_{ij}\beta_j + \sum_{j=1}^{n} x_{ij}\left(E_i^k + T_i^k\right) - \alpha_i$ should be determined. It is equivalent to solve the following sub-problem:

$$\text{Min } Z^{SPi} = (\gamma_{i+1} - \gamma_i)C_i + \gamma_i P_i - \sum_{j=1}^{n} x_{ij}\beta_j + \sum_{j=1}^{n} x_{ij}(E_i + T_i) - \alpha_i \quad (28)$$

$s.t.$

$$\sum_{j=1}^{n} s_j x_{ij} \le C \quad (29)$$

$$P_i \ge x_{ij} p_j \qquad j = 1, \dots, n \quad (30)$$

$$C_i \ge P_i \quad (31)$$

$$E_i \ge d - C_i \quad (32)$$

$$T_i \ge C_i - d \quad (33)$$

$$C_i \ge 0,\ P_i \ge 0,\ E_i \ge 0,\ T_i \ge 0 \quad (34)$$

$$x_{ij} \in \{0,1\} \qquad j = 1, \dots, n \quad (35)$$

There are m sub-problems with similar structure and every sub-problem corresponds to a position. The non-linear term $x_{ij}(E_i + T_i)$ is appeared in the objective function of the sub-problems. To linearize the sub-problems, set $x_{ij}(E_i + T_i) = e_j + t_j$. The sub-problems can be rewritten as follows:

$$\text{Min } Z^{SPi} = (\gamma_{i+1} - \gamma_i)C_i + \gamma_i P_i - \sum_{j=1}^{n} x_{ij}\beta_j + \sum_{j=1}^{n} \left(e_j + t_j\right) - \alpha_i \quad (36)$$

$s.t.$

$$\sum_{j=1}^{n} s_j x_{ij} \le C \quad (37)$$

$$P_i \ge x_{ij} p_j \qquad j = 1, \dots, n \quad (38)$$

$$C_i \ge P_i \quad (39)$$

$$E_i \ge d - C_i \quad (40)$$

$$T_i \ge C_i - d \quad (41)$$

$$e_j \ge E_i - M(1 - x_{ij}) \qquad j = 1, \dots, n \quad (42)$$

$$t_j \ge T_i - M(1 - x_{ij}) \qquad j = 1, \dots, n \quad (43)$$

$$C_i \ge 0,\ P_i \ge 0,\ E_i \ge 0,\ T_i \ge 0 \quad (44)$$

$$x_{ij} \in \{0,1\}, e_j \ge 0,\ t_j \ge 0 \qquad j = 1, \dots, n \quad (45)$$

If the optimal value of at least one of the sub-problems are negative, there are batch(es) that can be added to the master problem and improve the objective function value. The process of solving sub-problems and adding new bathes to RLMP continues until all sub-problems have positive optimal objective function values.

4-2-Solving the sub-problems for generating improving columns

It is not necessary that the sub-problems be solved optimally during the intermediate iterations of column generation approach and finding an improving column is adequate. Thus, a metaheuristic is used to solve the sub-problems during early iterations. When the metaheuristic is unable to find a suitable column for all sub-problems, the sub-problems are solved optimally. At the end of the column generation approach, all sub-problems must be solved optimally to make sure that the optimal solution of the node is found. After solving the sub-problems using the metaheuristic, if at least one of the sub-problems have a negative objective function, new columns are added to the RLMP. By solving updated RLMP, new dual values are obtained to update sub-problem objective functions. In this research, memetic algorithm is used for solving sub-problems heuristically. The details of the proposed memetic algorithm are briefly described in the following sub-sections.

4-2-1-Solution representation

Solution representation is one of the most important aspects of the metaheuristic algorithms. Every solution to sub-problems determines that which jobs are assigned to the batch. Hence, binary representation can be used for determining that a job is assigned to the batch or not. A chromosome can be defined as $Q = [a_1, a_2, \dots, a_n]$ in which $a_j \in \{0,1\},\ j = 1, \dots, n$. In this notation, $a_j = 1$ means that job j is assigned to the batch, and $a_j = 0$ denotes that it is not assigned. The fitness or objective function value of a chromosome is calculated based on the objective function value of the sub-problem.

4-2-2-Selection strategy for recombination

There are various methods for selecting chromosomes for crossover. In this research, the random tournament selection strategy is used. In this selection strategy, first two chromosomes are selected randomly from the population. Then, the random number r is drawn from continuous uniform distribution $U[0,1]$. If r is greater than 0.5, then the better chromosome is selected and the worse chromosome is selected otherwise. The better chromosome is the one with better sub-problem objective function. For selecting two parents and applying the crossover operator, the tournament strategy should be executed twice (once for every parent).

4-2-3-Crossover and mutation operators

In order to perform reproduction operations and to generate new solutions, it is necessary to use operators such as crossover and mutation. The crossover operator generates new solutions by combining the properties of the parents. Two-point crossover is used in this research. The mechanism of this operator is that after the selection of two parents, two points are randomly selected in the parents, and all the genes between these two points are transferred from one parent to the new offspring, and the rest of the genes are completed according to the another parent. An example of combination by applying two-point crossover for a problem with 10 jobs is presented in Figure 1.

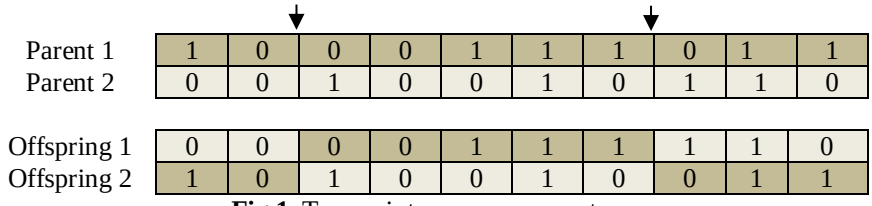

Fig 1. Two-point crossover operator

Mutation operator plays an important role in providing population diversity. In this research, a uniform mutation operator is used. In this operator, two genes are randomly selected first, and the

amount of selected genes is reversed (from zero to one or from one to zero). Uniform mutation operator is demonstrated in figure 2.

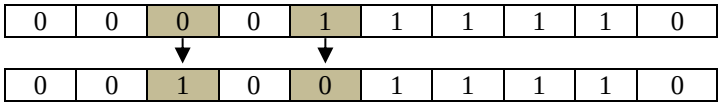

Fig 2. Uniform mutation operator

4-2-4-Local search procedure

Local search procedures rely on the neighborhood structures. The neighborhood structure indicates the relationship between solutions in the solution space. The neighborhood of a solution can be defined as all the solutions that can be achieved by performing any of the possible changes according to the neighborhood structure. In this study, a neighborhood is generated by changing a gene from one to zero or from zero to one. A hill climbing algorithm for this kind of neighborhood structure is as follows: start with an initial chromosome and perform a change on the chromosome, i.e. change a gene from one to zero or from zero to one. A change is accepted whenever it reduces the value of the objective function. But if the change increases the value of the objective function, it will be rejected and the chromosome will not change. After a successful change, this cycle continues on the improved chromosome. After a thorough check, local search will end if all possible changes are accomplished without achieving any improvement, otherwise another cycle of changes will run on the improved chromosome.

4-2-5-Updating the population

After generating new chromosomes through crossover and mutation steps the population should be updated. An intensive strategy is used for accepting new chromosomes in this research. Only if the fitness of the newly generated chromosome is better than at least one of the participating parents in crossover, then the worse parent is replaced with the new 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 is chosen 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. A detailed flowchart of the proposed memetic algorithm for solving sub-problems is depicted in figure 3.

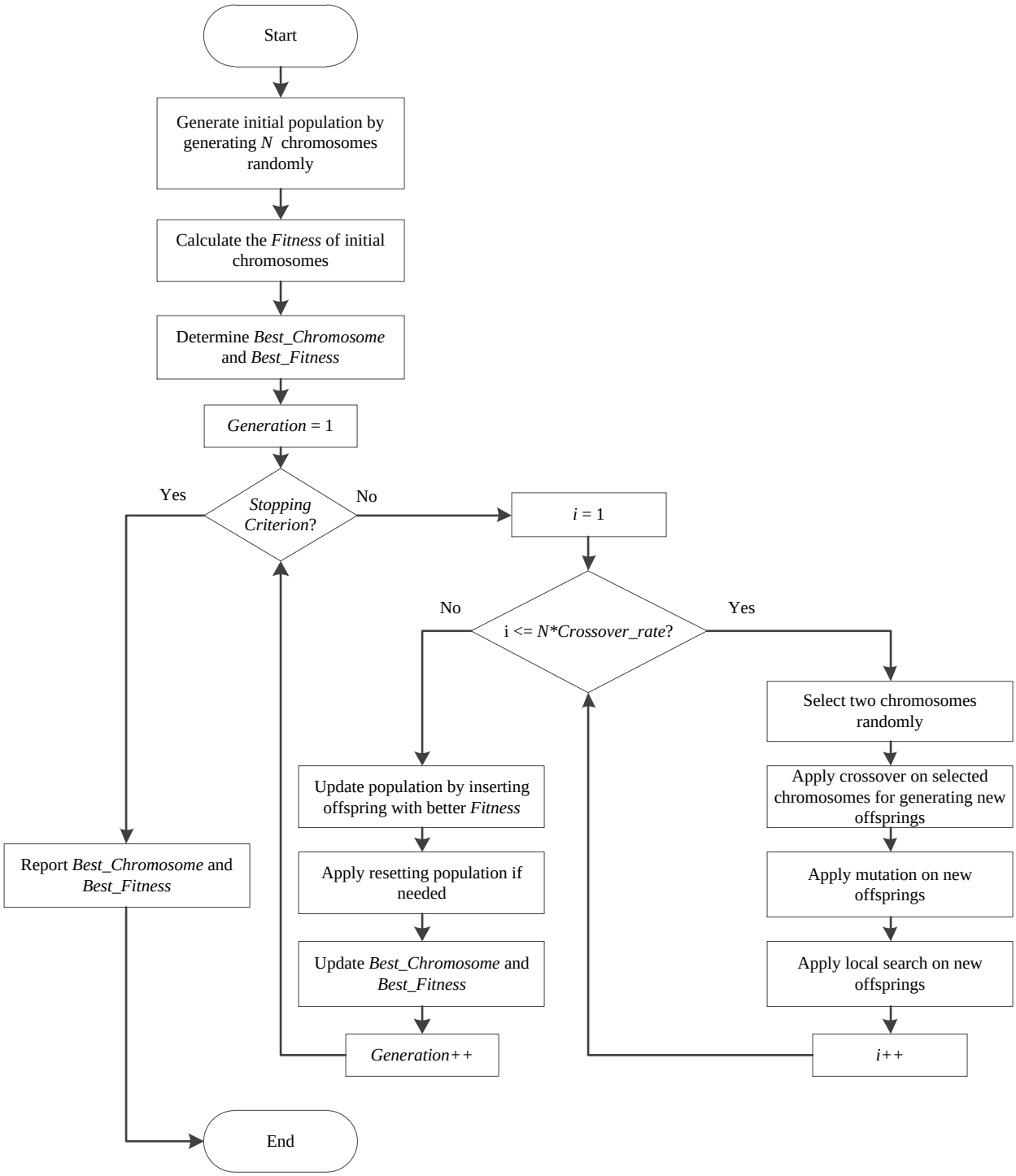

Fig 3. The flowchart of the proposed algorithm for solving sub-problems

5-Computational results

In this section, after describing the specifications of the random test instances, the performance of the proposed column generation approach is analyzed. ILOG CPLEX 12.0, a commercial optimization software, is used to solve the mathematical model. The proposed column generation approach is also coded with visual C++ 2008 using ILOG CPLEX (version 12.1) concert technology. All the algorithms are executed on a laptop with 2.53 GHz processor and 4 GB RAM.

Rafiee Parsa et al. (2017) generate a set of test instances for the $1|p-batch, s_j \leq C|\sum(E_j + T_j)$ problem. These test instances are used for computational experiments in this research. The specifications of the test instances are as follows:

- The number of jobs ranges from 10 to 200 (n =10, 20, 40, 60, 80, 100, 200).
- The capacity of machine B is assumed to be 40 in all the experiments.
- The job sizes are generated from a discrete uniform distribution in four categories:
    - Combination of small and large jobs (Discrete uniform [1, 40]).
    - Medium jobs (Discrete uniform [10, 20]).
    - Large jobs (Discrete uniform [10, 30]).
    - Small jobs (Discrete uniform [1, 10]).
- The processing time p_j of a job is drawn from a uniform random distribution with lower and upper limit 1 and 100.

The presented mathematical model in Section 3 can be used to solve the small sized instances optimally. CPLEX as an optimization software is employed to solve the mathematical model. Problems with 10 number of jobs are considered and 10 instances are generated randomly in each category of job sizes. Then the optimal solution (opt) is compared with the lower bound (lb) obtained by the proposed column generation approach. The results of this experiment are shown in table 1. The average percentage error of the column generation approach is reported in the last column. As can be seen from the results, the proposed lower bound is very close to the optimal solution in the small-sized instances.

Table 1. Comparison the results of CPLEX and CG for small-sized instances

Job size	**CPLEX (opt)**	**Column Generation (lb)**	$\mathbf{100 \times (opt - lb)/opt}$
[1, 40]	470.9	470.9	0%
[10, 20]	388.0	388.0	0%
[10, 30]	555.8	555.8	0%
[1, 10]	146.8	143.7	2%

To evaluate the performance of the proposed lower bounding method in the large sized instances, it is compared with the best known lower bounding method in the literature. Rafiee Parsa et al. (2017) proposed a lower bound for the research problem through a relaxation. They relax the constraint that jobs must be processed as a batch. It means that the batch processing machine is replaced by parallel identical unit-capacity machines. In other words, this corresponds to the relaxation of the constraint that all the jobs within a batch should be completed simultaneously. In their method, a batch processing machine with the capacity B is replaced by B parallel identical unit-capacity machines. Hence, at most B jobs can be processed at the same time.

The results obtained from the proposed lower bounding method by Rafiee Parsa et al. (2017) that is based on relaxation (LBR) and the proposed lower bounding method in this research that is based on column generation (CG) approach are presented in table 2. For each combination of the number of jobs and the job sizes, 20 instances are considered, for a total of 480. All the instances are solved by LBR and CG algorithms to find a lower bound for the problem. The average lower bound values of

LBR and CG are reported in columns 3 and 4, respectively. The last column represents the average percentage gap between LBR and CG. The percentage gap is calculated as $100 \times (CG - LBR)/(LBR)$.

Table 2. Comparison the results of LBR and CG

Job size	n	LBR	CG	Gap (%)
[1, 40]	20	1468.4	1770.5	20.6
	40	6177.7	7546.5	22.2
	60	12649.2	17283.7	36.6
	80	20133.0	31269.3	55.3
	100	29878.7	48071.3	60.9
	200	100258.6	184153.3	83.7
	Average			**46.5**
[10, 20]	20	1317.5	1474.5	11.9
	40	4277.8	5423.7	26.8
	60	8905.9	11970.3	34.4
	80	14109.5	21757.9	54.2
	100	19718.2	32861.4	66.7
	200	76395.1	130137.7	70.3
	Average			**44.1**
[10, 30]	20	1922.1	2089.9	8.7
	40	6959.3	8176.2	17.5
	60	15646.5	18263.3	16.7
	80	28364.8	31272.9	10.3
	100	43895.7	50136.4	14.2
	200	178193.0	194812.7	9.3
	Average			**12.8**
[1, 10]	20	423.7	540.3	27.5
	40	1341.1	2186.9	63.1
	60	2658.2	4444.7	67.2
	80	4800.5	7999.4	66.6
	100	7253.7	12190.0	68.1
	200	26492.4	45630.6	72.2
	Average			**60.8**

The results show that the column generation approach increases the lower bound around 41% in average. It also can be observed that the performance differences between two algorithms become more considerable by increasing the number of jobs. The average percentage gap based on different category of job sizes is depicted in T 4. It is noteworthy that the differences between two methods for the instances with large job sizes, the category [10, 30], are not as the other categories. When the job sizes are larger, more jobs are processed individually, and thus the influence of batching on the solutions would not be significant. In other words, the solution space is not as large as the other categories. It leads to almost similar performance for two methods.

Since LBR is based on a simple relaxation and calculates the lower bound in one iteration, its computational time is less than one second. On the other hand, column generation is an iterative approach and needs more computational efforts. A time limitation of 300 seconds is considered for the column generation in this research. It should be mentioned that, it is obvious that column generation needs more computational time to find the lower bound. However, in finding the lower bound the quality of lower bound is more important than the speed of finding it.

Another way to analyze the efficiency of the proposed column generation approach is to compare it with the lower bound obtained by CPLEX. Since the research problem is NP-hard, CPLEX could not solve the mathematical model for large-sized instances in a reasonable time. However, it could provide a valid lower bound based on linear programming relaxation of the mathematical model after a time limitation. The quality of lower bound provided by the column generation approach can be compared with the standard lower bound that is obtained by CPLEX. Totally 48 random instances (2 in each category of job sizes and number of jobs) are solved by CPLEX and the lower bounds are reported after 3600 seconds. The average lower bound of CPLEX is compared with the average of proposed lower bounding method in table 3. The percentage gap is calculated as $100 \times (CG - CPLEX)/(CPLEX)$. The results show that the proposed lower bound is significantly better than the lower bounds provided by the CPLEX and so the proposed method outperforms CPLEX.

Table 3. Comparison the results of CG and CPLEX

Job size	n	CPLEX	CG	Gap (%)
[1, 40]	20	1928.1	1947.5	1.0
	40	7279.3	7999.3	9.9
	60	12942.0	16592.3	28.2
	80	20937.9	30393.7	45.2
	100	28304.4	43783.4	54.7
	200	97785.4	164080.6	67.8
	Average			**34.5**
[10, 20]	20	1253.3	1253.3	0.0
	40	4540.7	4935.5	8.7
	60	8618.6	11491.5	33.3
	80	13707.5	19582.2	42.9
	100	18977.4	31054.0	63.6
	200	56609.9	96803.0	71.0
	Average			**36.6**
[10, 30]	20	2110.8	2110.8	0.0
	40	8516.3	9157.3	7.5
	60	16656.1	18738.1	12.5
	80	22016.1	30960.1	40.6
	100	30773.7	49138.7	59.7
	200	91172.3	157798.3	73.1
	Average			**32.2**
[1, 10]	20	462.0	486.3	5.3
	40	1790.6	1946.3	8.7
	60	2977.5	3866.9	29.9
	80	5505.2	8879.3	61.3
	100	8370.9	15944.5	90.5
	200	22340.7	43805.4	96.1
	Average			**48.6**

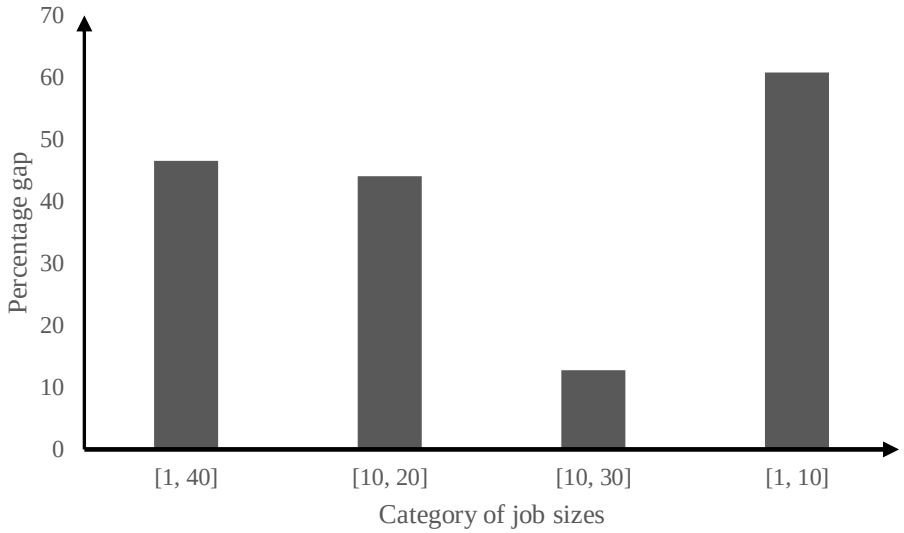

Fig 4. Comparison of average percentage gap based on category of job sizes

6-Conclusions

The minimization of total earliness and tardiness on a single batch processing machine is investigated in this research. A lower bounding method based on column generation approach is proposed. First, the problem reformulated as a Dantzig-Wolf decomposition model. Then, the master problem and the sub-problems are derived. An evolutionary algorithm is developed for solving the sub-problems and finding improving columns during early iterations. The computational results are conducted to analyze the performance of the proposed column generation approach. The results show that the proposed method can enhance the lower bound around 41% in average in comparison with the best known lower bounding method in the literature. Even though we have focused on instances where earliness and tardiness have the same importance, the proposed algorithms can be easily extended to handle other situations. Incorporating the proposed lower bound in a branch and bound tree and developing a branch-and-price algorithm for the research problem is an interesting extension and opportunity for the future research. It's also possible to extend the proposed method for other batch scheduling problems. Considering assumptions such as distinct due date and release time for jobs can help to make the problem more realistic.

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.

Azizoglu, M. & Webster, S. (2000). Scheduling a batch processing machine with non-identical job sizes. *International Journal of Production Research*, 38(10), 2173-2184.

Barnhart, C., Johnson, E.L., Nemhauser, G.L., Savelsbergh, M.W.P. & Vance, P.H. (1998). Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46(3), 316-329.

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.

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.

Damodaran, P., Kumar Manjeshwar, P. & Srihari, K. (2006). Minimizing makespan on a batch-processing machine with non-identical job sizes using genetic algorithms. *International Journal of Production Economics,* 103(2), 882-891.

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.

Jia, Z.-h., Gao, L.-y. & Zhang, X.-y. (2020). A new history-guided multi-objective evolutionary algorithm based on decomposition for batching scheduling. *Expert Systems with Applications,* 141, 112920.

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.

Kashan, A.H., Karimi, B. & Jolai, F. (2006). Effective hybrid genetic algorithm for minimizing makespan on a single-batch-processing machine with non-identical job sizes. *International Journal of Production Research,* 44(12), 2337-2360.

Kashan, A.H., Karimi, B. & Jolai, F. (2010). An effective hybrid multi-objective genetic algorithm for bi-criteria scheduling on a single batch processing machine with non-identical job sizes. *Engineering Applications of Artificial Intelligence,* 23(6), 911-922.

Lee, Y.H. & Lee, Y.H. (2013). Minimising makespan heuristics for scheduling a single batch machine processing machine with non-identical job sizes. *International Journal of Production Research,* 51(12), 3488-3500.

Li, S., Li, G., Wang, X. & Liu, Q. (2005). Minimizing makespan on a single batching machine with release times and non-identical job sizes. *Operations Research Letters,* 33(2), 157-164.

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.

Melouk, S., Damodaran, P. & Chang, P.-Y. (2004). Minimizing makespan for single machine batch processing with non-identical job sizes using simulated annealing. *International Journal of Production Economics,* 87(2), 141-147.

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.

Ogun, B. & Alabas-Uslu, Ç. (2018). Mathematical models for a batch scheduling problem to minimize earliness and tardiness. *Journal of Industrial Engineering and Management (JIEM),* 11(3), 390-405.

Polyakovskiy, S., Makarowsky, A. & M'Hallah, R. (2017). Just-in-time batch scheduling problem with two-dimensional bin packing constraints. In: Proceedings of the Genetic and Evolutionary Computation Conference (pp. 321-328).

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. & Husseini, S.M.M. (2017). Exact and heuristic algorithms for the just-in-time scheduling problem in a batch processing system. *Computers & Operations Research,* 80, 173-183.

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.

Rafiee Parsa, N., Karimi, B. & Moattar Husseini, S.M. (2016). Minimizing total flow time on a batch processing machine using a hybrid max–min ant system. *Computers & Industrial Engineering,* 99, 372-381.

Rafiee Parsa, N., Keshavarz, T., Karimi, B. & Moattar Husseini, S.M. (2019). A hybrid neural network approach to minimize total completion time on a single batch processing machine. *International Transactions in Operational Research,* n/a(n/a).

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.

Wilhelm, W.E. (2001). A technical review of column generation in integer programming. *Optimization and Engineering,* 2(2), 159-200.

Wilhelm, W.E., Damodaran, P. & Li, J. (2003). Prescribing the content and timing of product upgrades. *IIE Transactions,* 35(7), 647-663.

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.