A new mathematical formulation and a hybrid evolutionary algorithm for re-entrant flow-shop problem with release date

Reza Behmanesh[1] *, Kamran Kianfar*[2,*]

[1] Department of Industrial Engineering, Naghshejahan Higher Education Institute, Isfahan 8144167984, Iran

[2,*] Faculty of Engineering, University of Isfahan, Isfahan 81746-73441, Iran

Abstract

In this paper, we address the weighted multi-objective re-entrant flow-shop scheduling problem considering release dates in order to minimize makespan, total completion time, total tardiness, maximum idle time, and number of tardy jobs. Each job is taken into account with deterministic processing times, and release dates. The flow-shop comprised of two workshops in whose jobs are entered to the main workshop and after the first part of the processing, they are transferred to the second workshop and after this stage, the jobs are returned to the main workshop for the last part of the processing. We model the problem by a new mixed integer programming based on formulating sum of idle time as a new concept. Moreover, a hybrid evolutionary algorithm is proposed based on some dispatching rules, ant colony optimization, and genetic algorithm. The performance of the proposed algorithm on some test instances is compared to the mixed integer linear programming model as well as the state-of-the-art algorithms called genetic algorithm, tabu search, bio-geography based optimization, and artificial bee colony. The computational experiments show that our proposed approach outperforms other algorithms and the results indicate efficiency and capability of the proposed algorithm in comparison with the traditional algorithms.

Keywords: Re-entrant flow-shop scheduling, Idle time, Release date, Makespan, Tardiness, Hybrid evolutionary algorithm, Mixed integer linear programming.

1. Introduction

Production scheduling as a decision making process is found not only in manufacturing industries but also in service systems [1]. Meanwhile, efficient and optimal production schedules lead to key improvement in cost reduction and productivity. In several types of research in the scope of the simple flow-shop scheduling problem, it is assumed that jobs are available at the time of scheduling (i.e., a parameter naming release date/ready time/arrival time is defined that is assumed to be zero); however, in real practice, for example for service systems such as operating room scheduling, emergency cases as jobs are not available in time zero and we need to consider unequal release date as an additional parameter in problem. In the literature, scheduling under

[1] Phone number : +98-9133177893
Email: rezaehs@gmail.com

[2] Phone number : +98-9132050419 , +98-31-37934041
Email: k.kianfar@eng.ui.ac.ir

release date with zero value is called static scheduling, while scheduling with an unequal release date is called dynamic scheduling [1,2]. Moreover, the machines may be idle when all jobs are not ready for processing and the release date of jobs are not assumed to be the same. In this scheduling case, the total idle time will incur the cost to the system and it would be better to consider the total idle time as a performance indicator in the model.

On the other hand, there are real cases in service systems like operating room scheduling in which, the frozen section is removed from the patient in the operating room (main workshop) and then is transferred to the laboratory (second workshop) for experiments. After carrying out experiments by technicians, the surgeon will continue the last stage of the processing in the operating room. As it is observed in this system, scheduling the surgical cases may be done in a two-machine re-entrant flow-shop environment since the main workshop and second workshop are considered as the first and second machine, respectively. In other real case in production system like electronic industries for PCB, and semiconductor manufacturing, the re-entrant flow-shop is also applicable [3]. In this case, the components like chips, resistors, and transistors, are needed to be inserted on the top and bottom of the panel, and the process of the insertion (on the top and bottom) is done by the required resources twice [4]. The two-machine re-entrant flow-shop is seen in other production industries such as textile, and mirror manufacturing industries. For instance, there are coloring and drying machines for the dyeing process in the textile industry, in which the textile fabrics must be operated on each machine twice, or there are coating and drying machines for the plating process in the mirror production system, in which the mirrors should be processed on each machine twice [5]. Hence, the re-entrant flow-shop scheduling problem in both the scope of the service and manufacturing systems may be essential and attractive for practitioners and researchers. Therefore, we define a re-entrant flow-shop consist of two machines (dependent workshops) and this problem will be modeled and solved theoretically. In a re-entrant flow-shop with two machines, jobs are operated on each machine and each job must visit each machine two or more times [5]; however, in the current study jobs consist of three operations where a job is processed on the first machine twice and the second machine once. In this research, the following contributions to the literature are offered:

- A novel formulation is introduced to compute machine idle time and summation of idle times using new variables.
- A novel parameter namely, position-dependent release date as well as novel dependent workshops in re-entrant flow-shop problem are introduced.
- To formulate the problem, a new mathematical model is extended using a new parameter, a new variable of idle time, and weighted multi-objective. Besides, a novel hybrid swarm and evolutionary algorithm is proposed for the first time through combining bio-geography based optimization, artificial bee colony, and tabu search.

This paper deals with a multi-objective re-entrant flow-shop scheduling (MORFSS) problem in which all jobs are processed on two machines with unequal release dates. In this paper, we use workshops instead of machines since there is a machine in each workshop, and the parameters such as processing times and release dates are assumed deterministic. The contribution of the paper is in the field of operations research technique so that we define a novel concept for machine idle time and then the sum of idle time for each job is calculated. On the other hand, there are two workshops those the jobs are processed continuously.

The rest of this paper is structured as follows. Section 2 reviews the literature of re-entrant flow-shop scheduling. In section 3, the problem definition is stated and then a mathematical model for the problem is built in Section 4. Section 5 presents some dispatching rules as well as a new hybrid algorithm to achieve near optimal solutions. Section 6, provides illustrative examples and computational experiments, and lastly, the conclusion and some suggestions for future research are presented in section 7.

2. Literature review

There are many researches in the field of flow shop scheduling problem [6-12]; however, there are a few papers recently study the flow shop problem with the re-entrant system [13,14] while this system is applied in few real cases such as the PCB. In this section, we present various studies on re-entrant flexible flow-shop (RFSS) problem with several parameters such as sequence-dependent setup time, and release date. Common criteria in these works are maximum completion time (makespan), total completion time, total weighted completion time, total tardiness, total weighted tardiness, and integrating those as multi-objective problems. In the first of the literature review, we look at the structure of the problem and models which was extended by researchers. Then the applied algorithms for solving RFSS problem are reviewed. The studies on the RFSS problems in this section are divided into three classes; (i) single objective RFSS without considering the additive parameters such as sequence dependent setup time and release date, (ii) single objective RFSS with considering the aforementioned parameters, and (iii) MORFSS with/without considering the aforementioned parameters. Table 1 summarizes the studies in this field and presents them using their assumptions, objectives and solution methods.

Table1. Researches on RFSS problem up to 2021
[15,16]

In this subsection, the different models of RFSS are reviewed and classified into three aforementioned groups. In the lowest class, many works on single objective RFSS were done without using the important parameters such as sequence-dependent setup time or release date [17-19,5,20,21,4,22]. The authors have modeled the problem to minimize makespan and only difference between their works is applied in the solution method. Besides, there are other studies [23-25,20] in whose, the objectives such as total completion time, total weighted completion time, total tardiness, and total weighted tardiness are minimized to achieve an optimized schedule. As it is considered, a simple model of RFSS is tackled by using various algorithms to minimize different single objective. . In our research, the idle time of the machines and the total number of tardy jobs (TU) are considered in addition to total completion time (TC), total tardiness (TT), and makespan (C_{max}) as the objectives of the problem.

In the middle class, the RFSS model is extended using some complicated parameters such as sequence dependent setup time and there are a few studies in this scope. The single objective RFSS using time window is modeled to minimize makespan [26,27], besides, Waqas et al [28] addressed the RFSS problem to minimize makespan considering sequence-dependent setup time moreover in another study [29], the sequence dependent setup time is taking into account in modeling the problem to minimize total tardiness.

In the highest class, there are few researches in the field of the MORFSS. In the addressed MORFSS, total weighted tardiness is minimized in addition to the makespan and different algorithms are employed to tackle the problem [30-32]. On the other hand, El-Khouly et al. [33] and Moghaddam et al. [34] addressed the MORFSS and modeled that to minimize other objectives such as costs, total completion, and utilization. As it is indicated in the list of papers, the multi-objective problem using different complex parameters has not been studied and only in a research by Mousavi [35] MORFSS is modeled using sequence dependent setup time. On the other hand, there is no study that concentrate on several objectives (more than two objectives) using release dates and due-dates. To the best of our knowledge, the simultaneous optimization of makespan, total completion time, total tardiness, and tardy jobs for RFSS problem with release date has not been studied in the literature. Besides, a parameter, namely position (workshop) dependent release date is introduced to model MORFSS and there is no study that models this problem in two dependent workshops as we have done.

In addition to extension of the models for the RFSS, there are several approaches that have captured practitioners and researchers to solve this problem. Although, there are many researches employing exact methods such as branch and bound algorithm to tackle the RFSS [3,18,5,23,29,4] however, solving these problems have been attempted using evolutionary, and intelligence algorithms in order to reduce the computational time of algorithms [36,37]. For instance, genetic and hybrid evolutionary algorithms [26,19,16,31,21,24,32,25,38], hybrid tabu search with evolutionary algorithm [39], particle swarm optimization and hybrid with genetic [40], hybrid simulated annealing [17,13] and hybrid ant colony optimization [27] have been employed in the literature for solving RFSS or MORFSS problems. As it is obvious in the literature, GA and hybrid of this algorithm with other approaches have been applied more than other meta-heuristic algorithms to solve the problem. Moreover, this evolutionary algorithm is known as a powerful algorithm with the structure of complex combinatorial problems. Therefore, hybrid of meta-heuristic algorithms can be taken into account as an effective method to solve these problems, and a hybrid algorithm of bio-geography based optimization, artificial bee colony and tabu search is proposed for solving MORFSS problem. To the best of our knowledge, there is no study in the literature of MORFSS problem that employs the proposed algorithm as we did. Accordingly, the solution technique that we develop for this problem is novel, too.

3. Problem definition

Our strategy divides the re-entrant flow-shop with two workshops into two single machine scheduling problems. The flow-shop includes two workshops, so that a job is processed on the first workshop and then, after completion time on the first workshop, it is transferred to the second workshop; after that, the job is processed on the first workshop again to be finalized. Therefore, this problem is divided into two single-workshop problems so that the sequence of the jobs in the second workshop is dependent to the sequence of the jobs in the first workshop. The outline of the model is displayed in Fig. 1.

Fig. 1. The schematic model of re-entrant flow-shop

The first workshop (main-shop) with one machine is considered as the first single machine scheduling problem and the second workshop (sub-shop) with one machine is taken into account

as the second single machine scheduling problem. Processing time introduces the time that each job J_i is processed on the machine and release date determines the availability of jobs. Processing time of a job is divided into three parts; (i) the first part in which the job is processed in the main-shop, (ii) the second part in which the job is processed in the sub-shop, (iii) the final part in which job is processed in the main-shop again and the processing is completed. These three parts are denoted by p_a, p_2, and p_b in Fig. 1. All the processing times are constant, but the second step of processing in the sub-shop makes a new single machine scheduling problem considering dependent release dates for jobs. A set J of n jobs $\left(J_1, J_2, \ldots J_i, \ldots, J_n\right)$ with the processing times (p_{ia}, p_{ib}), due-dates (d_i) and release dates (r_i) are sequenced in the main-shop to minimize makespan (also called schedule length or maximum completion time), sum of completion times, total tardiness, maximum idle time, and tardy jobs in the first problem. For the second part, after the completion of the first processing on the main-shop, each job with processing time p_{i2} and position dependent release date is transferred to the sub-shop to minimize total completion time in the second problem. The position dependent release dates are calculated based the completion time of each job in the first part of the first problem. Several assumptions are adopted to define MORFSS as follows:

1. The sequence of jobs on two workshops is assumed to be the same (permutation flow-shop).
2. Preemption is not allowed.
3. The number of jobs is fixed.
4. The priority of all jobs is assumed to be the same.
5. All the data including the processing times and release dates are deterministic.
6. All jobs have three processing parts (two processes in the main-shop and one in sub-shop).

4. Mathematical modeling

In this part, we formulate a mixed integer linear programming (MIP) model and describe its notation, objectives and constraints. The sets/indices, parameters, and decision variables used in the mathematical model are described as follows. The problem is represented as $n/1,1/r_i/Obj$ so that (1,1) denotes the two workshops in the problem.

Sets and indices

J :	$\{1,2,\ldots,n\}$ $J_i, J_j \in J$	Set of jobs
W :	$\{1,2\}$ $w \in W$	Set of workshops

Parameters

$p_i^{1a}, p_i^{1b}, p_i^2$:	Processing time of job *i* in the first workshop (part a and part b), and the second workshop
d_i :	Due-date of job *i*
r_i :	Release date of job *i*
M :	A large positive number
n :	Total number of jobs

Decision variables

S_i^1, S_i^2 :	The start time of job *i* in the first and second workshop
C_i^1, C_i^2 :	The completion time of job *i* in the first and second workshop
T_i :	The tardiness of job *i*
$U_i\left(Binary\,variable\right)$:	Equals 1 if job *i* is tardy, 0 otherwise
α_{max} :	Total idle time for the main-shop
C_{max} :	Makespan
$z_{ij}\left(Binary\,variable\right)$:	Equals 1 if job *i* is sequenced before job *j*, 0 otherwise

Parameter r_i and the specific schedule of jobs may construct a new variable that is called idle time of machine for job i and is notated by I_i. The idle time of the machine is the period in which no job is processed by machine in the course of production, and it may be carried out at the start, and middle of the schedule. Therefore, idle time prolongs makespan and completion times. The machine idle time is classified into two types of nature and forced; nature idle time is the result of release date constraint, while forced idle is a consequence of schedule constructed [41]. Thus, forced idle time is sequence dependent, and it may be minimized by altering the sequence of jobs in the schedule. It should be noted that variable I_i is not calculated based on a similar equation for all jobs because this is sequence dependent. In this section, idle time is formulated based on the new concept, i.e., the sum of idle time for the job. There are three states when job J_j is released after J_{j-1} according to the following Gantt charts (Fig. 2). In the following, the notation $\left[j\right]$ is related to the job in the j^{th} position of the schedule. For example, $I_{[j]}$ denotes the idle time before the job in position j.

Fig. 2. All states of idle time after job releasing

Sum of machine idle times before job $J_{[j]}$, called $SIT_{[j]}$, is obtained according to three states, respectively shown in Fig. 2-a to Fig. 2-c: (1) job $J_{[j]}$ is released during the processing of job $J_{[j-1]}$ which means $r_{[j]} < C_{[j-1]}$; (2) job $J_{[j]}$ is released at completion time of processing $J_{[j-1]}$ where $r_{[j]} = C_{[j-1]}$,and (3) as job $J_{[j]}$ is released after completion time of processing $J_{[j-1]}$ where $r_{[j]} > C_{[j-1]}$. To calculate sum idle times before $J_{[j]}$, we can subtract $p_{[j-1]}$ from $r_{[j]}$ considering that $SIT_{[j]}$ is the partial sum of idle times before $J_{[j]}$. In *states 1* and *2*, the machine would not be idle for $J_{[j]}$, while in the *state 3* there is idle time for the machine before the start of the processing time of $J_{[j]}$. In other words, in *states 1* and *2*, sum idle times before $J_{[j]}$ is equal to sum idle times before $J_{[j-1]}$, while in the *state 3*, this variable is increased by idle time before processing $J_{[j]}$.

Therefore, $\max\left(SIT_{[j-1]}, r_{[j]} - p_{[j-1]}\right)$ is formulated in order to obtain $SIT_{[j]}$. Assuming that *states 1* or *2* occurs, $\max\left(SIT_{[j-1]}, r_{[j]} - p_{[j-1]}\right) = SIT_{[j-1]}$, and in the case of *state 3* we have $\max\left(SIT_{[j-1]}, r_{[j]} - p_{[j-1]}\right) = r_{[j]} - p_{[j-1]}$. According to the Gantt chart of (Fig. 3), the sum of idle time for each job is defined as equations (1)-(4).

Fig. 3. General Gantt chart

$$SIT_{[1]} = I_{[1]} = r_{[1]} \tag{1}$$

$$SIT_{[2]} = I_{[1]} + I_{[2]} = \max\left\{r_{[1]}, r_{[2]} - p_{[1]}\right\} \tag{2}$$

$$SIT_{[3]} = I_{[1]} + I_{[2]} + I_{[3]} = \max\left\{r_{[1]}, r_{[2]} - p_{[1]}, r_{[3]} - p_{[2]} - p_{[1]}\right\} \tag{3}$$

$$SIT_{[n]} = \sum_{i=1}^{n} I_{[i]} = \max\left\{r_{[1]}, r_{[2]} - p_{[1]}, \ldots, r_{[n]} - \sum_{i=1}^{n-1} p_{[i]}\right\} \tag{4}$$

Here, $I_{[i]}, r_{[i]}$ and $p_{[i]}$ represent idle time of machine before job on position i, release date of job on position i, and processing time of job on position i. The idle time for job on position j is determined as the following:

$$I_{[j]} = \sum_{i=1}^{j} I_{[i]} - \sum_{i=1}^{j-1} I_{[i]} \quad \forall\, j = \{2, \ldots, n\} \tag{5}$$

In Eq. (5), we have $I_{[j]} \geq 0$ and there is two status; if $I_{[j]} = 0$, it means machine is not idle from the completion time of job $J_{[j-1]}$ to the start time of job $J_{[j]}$, however if $I_{[j]} > 0$, that means machine is idle between the completion time and the start time of jobs $J_{[j-1]}$ and $J_{[j]}$ that equals to $I_{[j]}$. The total idle time for a machine, denoted by α_{max}, is an important variable for modeling the problem and is obtained as the following equation.

$$\alpha_{max} = \sum_{i=1}^{n} I_{[i]} \tag{6}$$

In this section, we provide an example for describing the novel formulation of idle times. In this research, this approach is just applied to calculate the idle time of second machine in RFSS and idle time of the first machine is not considered. The processing times and release dates of the jobs are given in Table 2 and the sequence $A \rightarrow B \rightarrow C \rightarrow D$ is selected to calculate the value of $I_{[i]}$.

Table2. The parameters of the example

$$I_{[1]} = r_{[1]} = r_A = 2$$

$$I_{[1]} + I_{[2]} = \max\{r_A, r_B - p_A\} = \max\{2,3\} = 3$$

$$I_{[1]} + I_{[2]} + I_{[3]} = \max\{r_A, r_B - p_A, r_C - p_B - p_A\} = \max\{2,3,-6\} = 3$$

$$I_{[1]} + I_{[2]} + I_{[3]} + I_{[4]} = \max\{r_A, r_B - p_A, r_C - p_B - p_A, r_D - p_C - p_B - p_A\} = \max\{2,3,-6,4\} = 4$$

$$I_A = I_{[1]} = 2, I_B = I_{[2]} = 1, I_C = I_{[3]} = 0, I_D = I_{[4]} = 1$$

The sequence solution $A \rightarrow B \rightarrow C \rightarrow D$ is displayed in Fig. 4 where the idle times are inserted.

Fig. 4. Sequence of A-B-C-D

In the example, the release date of job D equals 16 and is higher than the completion time of the previous job C (15). Therefore $\alpha_{max} = \sum_{i=A}^{D} I_{[i]}$ is equal to 4 and it includes of the idle times of jobs A, B, C, and D i.e. 2, 1, 0, and 1 respectively. As $\sum_{i=1}^{n} p_i$ is equal to 18, we have $C_{max} = 22$.

According to the above discussion, we need a new free variable α_i to calculate partial terms inside Eq. (4) for example term $\{r_{[i]} - \sum_{j=1}^{i-1} p_{[j]}\}$ before processing job i according to the position of that job is considered as free variable α_i.

Lemma. The variable $\alpha_{[j]} = r_{[j]} - \sum_{i=1}^{j-1} p_{[i]}$ is necessary to be taken into account as a free variable (positive/negative).

Let consider term $\left\{ r_{[j]} - \sum_{i=1}^{j-1} p_{[i]} \right\}$ for $J_{[j]}$. Also, the sum of the idle times before job $J_{[j]}$ is assumed to be zero. There are three states for its release date in the workshop to be processed. First state occurs if $J_{[j]}$ is released before the completion time of job $J_{[j-1]}$, it means $r_{[j]} < \sum_{i=1}^{j-1} p_{[i]}$ and therefore value of term $\alpha_{[j]}$ is negative. In the second state, job $J_{[j]}$ is released exactly on the completion time of $J_{[j-1]}$, it means $r_{[j]} = \sum_{i=1}^{j-1} p_{[i]}$ and therefore the value of term $\alpha_{[j]}$ equals to zero. In the last

condition, job $J_{[j]}$ is released to process after the completion time of job $J_{[j-1]}$ which means $r_{[j]} > \sum_{i=1}^{j-1} p_{[i]}$ and the value of term $\alpha_{[j]}$ become positive.

A MIP model is formulated for the considering problem with *n* jobs and two workshops as follows.

$$\min \sum_{i=1}^{n} C_i^1 + \sum_{i=1}^{n} C_i^2 + \alpha_{max} + \sum_{i=1}^{n} T_i + \sum_{i=1}^{n} U_i + C_{max} \quad (7)$$

$$\alpha_i \geq S_i^1 + p_i^{1a} - \sum_{j=1}^{n} z_{ji} p_j^2 \qquad i \in \mathbf{I} \quad and \; j \neq i \quad (8)$$

$$\alpha_{max} \geq \alpha_i \qquad i \in \mathbf{I} \quad (9)$$

$$C_j^2 - M.z_{ij} \leq S_i^2 \qquad i \in \mathbf{I} \quad and \; j \neq i \quad (10)$$

$$C_i^2 - M\left(1 - z_{ij}\right) \leq S_j^2 \qquad i \in \mathbf{I} \quad and \; j \neq i \quad (11)$$

$$S_i^2 \geq \alpha_{\mathrm{i}} + \sum_{j \neq i} z_{ji}.p_j^2 \qquad i \in \mathbf{I} \quad (12)$$

$$C_i^2 \geq p_i^2 + S_i^2 \qquad i \in \mathbf{I} \quad (13)$$

$$C_{max} \geq C_i^1 \qquad i \in \mathbf{I} \quad (14)$$

$$S_i^1 \geq r_i \qquad i \in \mathbf{I} \quad (15)$$

$$C_j^1 - M.z_{ij} \leq S_i^1 \qquad i \in \mathbf{I} \quad and \; j \neq i \quad (16)$$

$$C_i^1 - M\left(1 - z_{ij}\right) \leq S_j^1 \qquad i \in \mathbf{I} \quad and \; j \neq i \quad (17)$$

$$C_i^1 \geq p_i^{1b} + C_i^2 \qquad i \in \mathbf{I} \quad (18)$$

$$T_i \geq C_i^1 - d_i \qquad i \in \mathbf{I} \quad (19)$$

$$C_i^1 \leq d_i + M.U_i \qquad i \in \mathbf{I} \quad (20)$$

$$S_i^1, S_i^2, C_i^1, C_i^2, T_i \geq 0 \quad (21)$$

$$z_{ij} \; , U_i \in \{0,1\} \quad (22)$$

In the above model, Eq. (7) states the objective function which minimizes the total completion time in each workshop, the idle time of the machine in sub-workshop, total tardiness, tardy jobs in the main workshop, and the makespan of the main workshop that should be minimized.

The constraints (8-13) are related to the sub-workshop and the constraints (14-20) are related to the main workshop. Inequality (8) reflects the same partial term related to Eq. (4) according to the position of job i. The total idle time of the second machine during production is specified by equation (9). Constraints (10) and (11) make sure that the completion time of job i in sub-workshop is before the start time of job j in the same workshop if $z_{ij}=1$ and vice versa. Relation (12) guarantees that processing of job j in sub-workshop must be started after its release date based on its position. Inequality (13) ensures that the difference between start time and completion time of job i in sub-workshop is greater than processing time of that job.

Constraint (14) calculates the makespan and constraint (15) guarantees that the start time of a job in the main workshop must be higher than its release date in the same shop. Equations (16) and (17) make sure that the completion time of job i in the main workshop is before the start time of job j in the same workshop if $z_{ij}=1$ and vice versa. Constraint (18) enforces that difference between start time and completion time of job i in the main workshop should not be less than processing time of that job. The tardiness of jobs is specified by inequality (19), and constraint (20) determines the jobs with tardiness in the main workshop. Lastly, relations (21-22) define the type of decision variables. The multi-objective function is obtained based on the simple additive weighting (SAW) method where the weights of all normalized objectives are the same and weighted multi-objective function is calculated as following equation:

$$z=\frac{\sum C_1-\sum C_1^{min}}{\sum C_1^{max}-\sum C_1^{min}}+\frac{\sum C_2-\sum C_2^{min}}{\sum C_2^{max}-\sum C_2^{min}}+\frac{\alpha_{max}-\alpha_{max}^{min}}{\alpha_{max}^{max}-\alpha_{max}^{min}}$$
$$+\frac{\sum T-\sum T^{min}}{\sum T^{max}-\sum T^{min}}+\frac{\sum U-\sum U^{min}}{\sum U^{max}-\sum U^{min}}+\frac{C_{max}-C_{max}^{min}}{C_{max}^{max}-C_{max}^{min}} \quad (23)$$

The notations obj^{min} d obj^{max} in the above equation describe the minimum and maximum possible values for each objective so that we can obtain the normalized objectives. To calculate the minimum and maximum of the objectives, firstly, the model was formulated by each objective as a single objective problem and then, the optimal (minimum) value for each objective was obtained separately. For the maximum values, an upper bound was generated by running the BBO algorithm on the single objective model for each objective separately.

5. The proposed hybrid evolutionary algorithm

The RFSS is an NP-hard problem [34,13]; on the other hand, the release date parameter is considered in our problem and it increases the complexity of the problem. So, the problem in hand is considered as a strongly NP-hard problem in re-entrant flow-shop environment. In this section, we present a new hybrid algorithm including bio-geography based optimization [42], artificial bee colony [43], and tabu search [44] as a new solution to solve the considering problem. Moreover, some dispatching rules such as shortest processing time (SPT), and earliest release date (ERD) are

employed in order to improve the performance of the hybrid algorithm since each dispatching rule gives us an optimal solution for one of the objectives. The shortest processing time rule optimizes the sum of the completion times in a single machine schedule. Also, it is indicated that the makespan is minimized in a polynomial time by sorting the jobs based on increasing order of release dates (earliest release date rule) [45] and this rule is prone to minimize makespan for a single machine scheduling problem with release date.

To improve the performance of the algorithm and to obtain better solutions in saving time, the characteristics of three aforementioned algorithms are considered so that the obtained solutions of bio-geography based optimization after termination conditions are taken into account as initial solutions for artificial bee colony, and then the best solutions after stopping criterion is considered as the initial solution for tabu search. Furthermore, shortest processing time as well as earliest release date rules are applied to scheme representation not only to generate better initial solutions but also to save the computational time. Scheme representation plays an important role in saving the computational time for achieving the better populations, because generating infeasible solutions and needing extra time for repairing infeasible solutions may be tackled by applying an efficient and effective solution representation.

Random key-based encoding scheme is adopted to represent feasible schedules and to generate them after performing the operators of the algorithms on solutions. In this scheme, random numbers are generated in a range and based on uniform distribution, and then the position of each job on workshops is decoded via sorting the random numbers. Random-key mechanism was firstly introduced by Bean [46] in a genetic algorithm to solve a sequencing problem. It must be pointed out that the encoding scheme for our problem is a suitable approach to generate feasible solutions. The random-key solution will not be infeasible after using the different operators of the algorithm, since the encoded solution is decoded to the sorted integer numbers and this presents a feasible solution for the problem in hand. According to the assumptions, the sequence of each job on the first workshop and the second workshop is the same and therefore, the scheme represents the sequence of jobs on both workshops. For example, in the random key encoding scheme {0.1, 0.5, 0.4, 0.6, 0.3}, each random number occupies a position i.e. 0.5 occupies the second position or 0.3 occupies the fifth position. After sorting this scheme, the solution is encoded via sequence {1, 5, 3, 2, 4}, and this shows that job 1 takes the first position for processing and then successively followed by jobs 5, 3, 2 and 4. To evaluate the final solution (schedule), the fitness of the solution is calculated as the weighted normalized objective function.

A near-optimal sequence for minimizing the fitness function (23) is constructed according to the pseudo code of the *Algorithm 1* that is presented as follows. The stopping criteria is checked based on the maximum iteration (Max_It) as an important parameter of the algorithm. This parameter is set based on the convergence of the decoded schedules in generations.

Algorithm 1. Constructing sequence by using ERD and SPT rules in BBO+ABC+TS

---

1. Input: Jobs' characteristics like processing times, due dates, and release dates

2. Initialization:

Parameter setting: $nPop$, Max_It, $\mu, \lambda, P_{mutation}$, $Tabu_{Tenure}$, $Tabu_{list}$, $Tabu_{tuner}$, and $Aspiration_{criteria}$

Generate a schedule by sorting jobs based on ERD rule so that $r_{[1]} \le r_{[2]} \le \ldots \le r_{[n]}$

Calculate summation P_2 of jobs in the sub-shop, and P_{1a} and P_{1b} of jobs in the main-shop
Generate a schedule by sorting jobs based on SPT rule
Generate $nPop - 2$ schedules randomly

Put all $nPop$ schedules (generated solutions) in BBO as initial habitats

3. Apply BBO until the stopping criteria are met and Record the elites

For each solution

If generated random parameter is less than λ for each solution (habitat)

Emigration: determine emigration probability and select a source habitat and migrate the solution based on operating in (Fig. 5)

ElseIf generated random parameter is less than $P_{mutation}$ for each solution (habitat)

Mutation: mutate each non-elite habitat with a random parameter based on operating in (Fig. 5)

End if

Combine new solutions from Emigration or Mutation with kept habitats as next iteration

Evaluation: evaluate each habitat according to the multi-objective function (z)

End for

4. Apply ABC by feeding the last population of BBO until the stopping criteria are met, and **Record the elites**

For each solution (output of BBO)

New solution: update the position of solution (recruited and onlooker bee) based on acceleration coefficient as shown in (Fig. 6)

Evaluation: evaluate each solution according to the multi-objective function (z)

End for

5. Apply TS by feeding the best solution of ABC until stopping criteria are met and **Record the best**

For the decoded solution (output of ABC)

Do action: do swap, insertion, and inversion (Fig. 7)

Tabu Memory: add movement on tabu list and check aspiration criteria (the best solution)

Evaluation: evaluate each solution according to the multi-objective function (z)

End for

6. Return set the final schedule and objective function

In BBO, two operators are carried out to generate new habitats including emigration and mutation. The BBO emigration strategy is considered as an adaptive process, in which the existing habitats or old migrants are modified by a recombination strategy. In this strategy, the difference between the position of the source habitat and the old migrant is added to the position of the old migrant, and then the new position is calculated. A habitat can change suddenly due to random events in the mutation process by recombination of the random vector and old migrant [42]. These operators are shown in Fig. 5.

Fig. 5. An illustration of the operators of BBO on schedule solution

In the ABC algorithm, the onlooker and employed bees are sent to the food sources and the new position of the bees is calculated based on the displacement. In this approach, a source site (bee) is selected randomly and the difference between the old position of the bee and the source site is added to the old position of the bee for calculation of the new bee's position [43]. The operator of updating position of bees is displayed in Fig. 6.

Fig. 6. An illustration of the operator of ABC on schedule solution

In TS, the neighborhood of a schedule is created according to a seed schedule by applying some actions and movements that are not listed in tabu memory. However, if a movement satisfies aspiration criteria, the constructed neighborhood may be selected as a good schedule. The movements and procedures of the algorithm are displayed in the following example including insertion, swapping, and inversion. If the content of the cell in the scheme is considered as job sequence J={J(1),…J(n)}, so J(i) is a job that has *i*th position in the schedule, and the neighborhood is defined as the set of all schedules created by applying movements. In insertion action, the position of a job is exchanged from the current position to another position. In swapping action, the positions of two jobs chosen randomly are exchanged. In Inversion action, the part of jobs sequence is chosen and then their positions are inversed [47]. Figure 7 presents all movements on a scheme representation. For example, the position of job #8 is exchanged from 2 to 5 in the new neighborhood by insertion, the positions of job #8 and job #3 are exchanged in the new neighborhood by swap, and the positions of jobs #8, #4, #1, and #3 are inversed from 2, 3, 4, and 5 to 5, 4, 3, and 2 in the new neighborhood by inversion operator.

Fig. 7. An illustration of the movements of TS on schedule solution

It is noted that, the simple algorithms such as ABC, BBO, and TS are developed for the mentioned problem according to the same structure of the algorithms that are integrated in the proposed algorithm, and moreover, GA is developed for the problem by using random key representation scheme and basic operators similar to mutation and one-point crossover.

6. Computational experiments

To evaluate the proposed algorithm, we generate random jobs including processing time, due dates, and release dates. In this simulation, we consider three test sample cases. These cases are classified into small, medium, and large sizes which are different in processing times, the number of jobs, the due dates, and the release dates. The first case consists of four various instances or problems and other cases include three problems. Cases category and their specifications are shown in Table 3. As it is observed, the problems of each case are different in the number of jobs (column 3), the processing time (column 4), and the release date (column 5). While generating data for times, we considered ranges in which values are represented by an interval, and it means that the given values are chosen randomly within this range. Each problem case is solved 10 times so that 100 samples are provided to evaluate the proposed algorithm. Besides, due date for each

job i is generated based on the following equation [48]. The parameter k in this equation is set to 1 in order to obtain shorter due dates.

$$d_i = r_i + k.p_i \tag{24}$$

Table3. Test cases and structure

All algorithms are coded in MATLAB software and ran on an Intel Core i5-7200U, 3.18 GHz personal computer with 4 GB of RAM. Moreover, the MIP model was coded in GAMS *v* 24.1 and run by solver CPLEX *v*12.5 in GAMS *v*24.1. Firstly, we applied the proposed hybrid algorithm and the MILP model on small cases of Table 3 to validate our approach. All algorithms are repeated 10 times and then the mean of the objective function found by the proposed algorithm is compared to the results of MILP. The algorithm is validated in comparison with MILP model as shown in Table 4. As it is indicated, the proposed algorithm is able to obtain 99.996% of the average optimal solution in 1.52% of the average time of the exact algorithm (solver CPLEX in GAMS).

Table 4 The validation of the BBO+ABC+TS algorithm

Here, the impact of changes in processing times (p with blue color), release dates (r with red color), and due dates (d with gray color) on the objective value is analyzed. The sensitivity analysis of objective value versus increasing and decreasing the parameters on problems 1 and 2 is displayed in Fig. 8. The horizontal axis represents the changes of the parameters (0.9 means the parameter is decreased by 10% and 1.1 means the parameter is increased by 10%) and the vertical axis shows the changes of the objective. The changes of release date and due date have low effect on the objective with small increments, while the changes of processing time have high effect on the objective in comparison with other parameters. Besides, with increasing the size of the problem, the impact of the processing time on objective is increased while the impact of the due date as well as release date is decreased.

Fig. 8. Sensitivity analysis of multi-objective versus parameters

All the algorithms were repeated 10 times for each instance in order to compare five algorithms. The relative percent deviation (RPD) measure [49] was applied to homogenize all data because the obtained objective values of problems are heterogeneous. The RPD value of each objective is obtained according to the following equation:

$$RPD_{ij} = \frac{obj_{ij} - min_j\left(obj_{ij}\right)}{min_j\left(obj_{ij}\right)} \tag{25}$$

where the problem is denoted by i, and j is introduced as the notation of the algorithm; for example, obj_{ij} is correspondent to i^{th} problem that is solved by j^{th} algorithm . The different parameters of the proposed algorithm effect on the diversification and intensification and hence, setting parameters has key role on the performance of the algorithm. The important parameters of the proposed algorithm include maximum iteration, number of population, μ,λ, mutation rate, Tabu_list, and Tabu_tuner. Therefore, several parameter values of the algorithms were set based

on the observations (quality of the solutions and computational time). All the parameters were set using trial and error, and after several trials the best parameters were determined. The evaluation criterion for determining the best level of the parameters is the quality of the solution with saving the time. To determine the best level in each case, the largest problem is selected. In this strategy applied to set the parameters, parameter *Max_It* was set first and its value was fixed, then the *nPop* was set for each algorithm. This strategy was also applied to set mutation rate and *Tabu_tuner* for TS. The mutation rate and *Tabu_tuner* were set on 0.1 and 3, respectively for all problems; and since, μ (emigration rate) and λ (immigration rate) are formulated based on population ($\mu = Uniform(1, nPop)$ and $\lambda = 1 - \mu$), setting these parameters is subjected to the number of population. The final parameters of each algorithm for the three cases are displayed in Table 5.

Table 5. Setting parameters of algorithms

In order to compare the five algorithms for solving large-scale instances, we ran the algorithms using data of Table 3. The results of the normalized experiments based on RPD and computational time of algorithms are indicated in Tables 6-7. According to the results, the proposed algorithm obtains more number of optimum solutions with saving time in comparison with the-state-of-the-art algorithms. Moreover, the bar-charts indicate the comparison between five algorithms for each problem. Bar-charts of the multi-objective (RPD) of the algorithms on each problem is shown in Fig. 9 and bar-charts of computational time of the algorithms on each problem is displayed in Fig. 10. It is indicated that the behavior of all the algorithms on the small category of the problems are almost the same. However, tabu search and the proposed algorithm present better results on problem 4 in comparison with other algorithms. For the medium and large-size problems, the proposed algorithm outperforms other algorithms not only in finding near optimal solutions but also in saving the computation times.

Fig. 9. Bar-charts of the objective (RPD) of the algorithms for each problem

Fig. 10. Bar-charts of the computational time of the algorithms for each problem

The hybrid algorithm utilizes the advantages of tabu search, artificial bee colony, and bio-geography-based optimization, at the same time. Bio-geography-based optimization generates better results on medium and large problem instances in comparison with three basic algorithms optimally and timely; also, artificial bee colony performs in saving time on large problem instances when it is compared with tabu search and genetic algorithm. On the other hand, tabu search gets to better results in comparison with artificial bee colony and genetic algorithm on problem 8 as well as small and medium-size groups. So, tabu search and bio-geography-based optimization are powerful algorithms to find near optimal solutions and artificial bee colony saves time to find solutions. The combination of these methods and dispatching rules is promising to find better solutions in less time.

Figure 11 displays convergence plot of all applied algorithms for the medium case example. The red line describes the best solution obtained by ABC at each iteration, the black line states the best

generations of BBO, the yellow line indicates the best of GA, the violet line shows the best of TS, and the best solutions of the proposed algorithm are shown by green line. As it is observed, the solutions obtained by the proposed algorithm have significant difference in comparison with those of other algorithms.

Fig. 11. Convergence plot of all algorithms for medium case (Problem 6)

Table 6 The RPD value of multi-objective (mean and standard deviation) of the solutions of the algorithms

Table 7 The Computational Time (mean and standard deviation) of the solutions of the algorithms

Here, the ANOVA test is applied to verify whether the convergence values of algorithms are different significantly. The result of the ANOVA for comparison of five algorithms is presented in Table 8, and it is indicated that convergence values of algorithms are different. Furthermore, both Tamhane's comparison and Dunnett T3's comparison were applied in ANOVA to determine the relationship between algorithms for finding the better algorithm. The homogeneity of variances was rejected according to Levene test at 0.05 significance level, and so the Dunnett T3 and Tamhane tests were carried out to show significance between different results at 0.05 significance level. The results of the comparison tests for objective values and computational time values are presented in Tables 9-10 and as it is obvious that our proposed algorithm outperforms others significantly. As a consequence, we infer that the proposed approach is a promising meta-heuristic algorithm to provide good solutions for solving the problem because of its better exploration in comparison with other algorithms.

Table 8 ANOVA test for comparison of multi-objective and run time of algorithms

Table 9 Comparison Tests for five algorithms in the convergence objective value

Table 10 Comparison Tests for five algorithms in the run time value

Figure 12 presents the median and inter-quartile range (IQR) values of the algorithms on the test problems. The size of each rectangle displays the IQR. The short line at the end of each rectangle indicates the maximum and minimum values and the median is represented by the short line in each rectangle. As it is indicated, BBO+ABC+TS occupies the lowest position in the graphs (a-objective, b-computational time) as compared to other algorithms. Furthermore, the proposed algorithm's rectangle occupies the smallest area, and this indicates that this has the smallest degree of variance. Therefore, the proposed algorithm outperforms others by finding better solutions in less computational time.

Fig. 12. Box-plot of Multi-objective (a) and computational time (b) for algorithms

7. Conclusions

In this research, we addressed the re-entrant flow-shop scheduling problem inside dependent workshops under dynamic scheduling to minimize makespan, total completion time, total tardiness, maximum idle time, and tardy jobs. In this problem, each job is considered with deterministic processing times, release date, and due date. A new hybrid evolutionary algorithm called BBO+ABC+TS using earliest release date and shortest processing time rules was proposed to solve this problem. To illustrate our methodology, we provided three case test data that each case comprises of three problems with different size. Our proposed algorithm is evaluated by two performance factors of computational time and quality of the objective function generated. The performance of the proposed approach was validated by a MIP model and it is indicated that the proposed algorithm is capable to achieve 99.996% of the average optimal solution in 1.52% of the average time of the exact algorithm (solver CPLEX in GAMS). The proposed algorithm was compared to the-state-of-the-art algorithms such as BBO, ABC, TS, and GA. It was observed that our proposed algorithm outperforms others by results and discussions.

There are two strong points in our work including the extension of the RFSS problem with considering some objectives as important performance indicators of the scheduling problem simultaneously, and developing the hybrid meta-heuristic algorithm to solve the extended problem. In the extended model, not only the producer-oriented criteria such as total completion time, makespan, and idle time are taken into account, but also the customer-oriented criteria such as total tardiness, and tardy jobs are considered. Optimizing these objectives in real-world cases leads to increasing the satisfaction of both the customers and the management system. For instance, in the surgical case scheduling problems, the collaborative system of the laboratory and operating theater can be considered as RFSS, in which the patients are sequenced for the surgical operation in the operating room and the laboratory operations as two machines of the flow shop. The optimization of total tardiness in this real case increases the satisfaction of the patients, while the satisfaction of the management system is enhanced by optimizing the total completion time as well as idle time. According to the sensitivity analysis of multi-objective versus parameters in Fig. 8, it is observed that the objective value is exacerbated with decreasing the due date because two objectives such as tardy jobs and total tardiness are increased. Therefore, it is necessary for a production manager to determine high due dates before the scheduling in re-entrant flow-shop to minimize the objective function. On the other hand, the objective value is worsened with increasing the both release date and processing time since some objectives such as total completion time, makespan, and idle time are increased. It is essential for a production manager to control processing times as well as release dates to minimize the objective function before the scheduling in re-entrant flow-shop.

We suggest some directions as opportunities for the future research in this area. To construct a robust schedule, it would be essential to consider uncertain processing times and release dates. It means that we can provide fuzzy data instead of crisp data to illustrate the proposed algorithm for solving the problem. Applying Pareto-based many objective evolutionary algorithms may be novel and powerful methods to solve Pareto based MORFSS. At last, if some assumptions are relaxed, the new complex problem is generated. For example, the preemption may be allowed, or the setup

time of the jobs may be assumed to be sequence dependent, or the priority of jobs may be considered.

References

1. Pinedo, M.L. "Scheduling: Theory, Algorithms, and Systems". p. 670. Springer International Publishing (2016).
2. Baker, K.R. "Introduction to sequencing and scheduling". pp. ix, 305 p. Wiley New York, (1974).
3. Che, A., Chabrol, M., Gourgand, M., et al. . "Scheduling multiple robots in a no-wait re-entrant robotic flowshop", *International Journal of Production Economics*, **135**(1), pp. 199-208 (2012).
4. Pan, J.-H. and Chen, J.S. "Minimizing makespan in re-entrant permutation flow-shops", *Journal of the Operational Research Society*, **54**(6), pp. 642-653 (2003).
5. Choi, S.W. and Kim, Y.D. "Minimizing makespan on a two-machine re-entrant flowshop", *Journal of the Operational Research Society*, **58**(7), pp. 972-981 (2007).
6. Bootaki, B. and Paydar, M.M. "On the n-job, m-machine permutation flow shop scheduling problems with makespan criterion and rework", *Scientia Iranica*, **25**(3), pp. 1688-1700 (2018).
7. Abbaszadeh, N., Asadi-Gangraj, E. and emami, s. "Flexible flow shop scheduling problem to minimize makespan with renewable resources", *Scientia Iranica*, pp. - (2019).
8. Gholami, H.R., Mehdizadeh, E. and Naderi, B. "Mathematical models and an elephant herding optimization for multiprocessor-task flexible flow shop scheduling problems in the Manufacturing Resource Planning (MRPII) system", *Scientia Iranica*, **27**(3), pp. 1562-1571 (2020).
9. Shao, Z., Shao, W. and Pi, D. "Effective heuristics and metaheuristics for the distributed fuzzy blocking flow-shop scheduling problem", *Swarm and Evolutionary Computation*, **59** p. 100747 (2020).
10. Wang, G., Gao, L., Li, X., et al. . "Energy-efficient distributed permutation flow shop scheduling problem using a multi-objective whale swarm algorithm", *Swarm and Evolutionary Computation*, **57** p. 100716 (2020).
11. Baskar, A. and Xavior, M.A. "New idle time-based tie-breaking rules in heuristics for the permutation flowshop scheduling problems", *Computers & Operations Research*, **133** p. 105348 (2021).
12. Ozcan, B., Yavuz, M. and Fıglalı, A. "A data mining-based solution method for flow shop scheduling problems", *Scientia Iranica*, **28**(2), pp. 950-969 (2021).
13. Wu, C.-C., Liu, S.-C., Cheng, T.C.E., et al. . "Re-Entrant Flowshop Scheduling With Learning Considerations to Minimize The Makespan", *Iranian Journal of Science and Technology, Transactions A: Science*, **42**(2), pp. 727-744 (2018).
14. Rifai, A.P., Kusumastuti, P.A. and Mara, S.T.W. "MULTI-OPERATOR HYBRID GENETIC ALGORITHM-SIMULATED ANNEALING FOR REENTRANT PERMUTATION FLOW-SHOP SCHEDULING", *ASEAN Engineering Journal*, **11**(3), pp. 109-126 (2021).
15. Huang, J.D., Liu, J.J., Chen, Q.X., et al. . "Minimizing makespan in a two-stage flow shop with parallel batch-processing machines and re-entrant jobs", *Engineering Optimization*, **49**(6), pp. 1010-1023 (2017).
16. Geng, K., Ye, C., Cao, L., et al. . "Multi-Objective Reentrant Hybrid Flowshop Scheduling with Machines Turning on and off Control Strategy Using Improved Multi-Verse Optimizer Algorithm", *Mathematical Problems in Engineering*, **2019** p. 2573873 (2019).
17. Amin Naseri, M.R., Tasouji Hassanpour, S. and Nahavandi, N. "Solving Re-entrant No-wait Flow Shop Scheduling Problem", *International Journal of Engineering*, **28**(6), pp. 903-912 (2015).
18. Chen, J.-S. and Chao-Hsien Pan, J. "Integer programming models for the re-entrant shop scheduling problems", *Engineering Optimization*, **38**(5), pp. 577-592 (2006).

19. Chen, J.-S., Pan, J.C.-H. and Lin, C.-M. "A hybrid genetic algorithm for the re-entrant flow-shop scheduling problem", *Expert Systems with Applications,* **34**(1), pp. 570-577 (2008).
20. Jing, C., Tang, G. and Qian, X. "Heuristic algorithms for two machine re-entrant flow shop", *Theoretical Computer Science,* **400**(1), pp. 137-143 (2008).
21. Lin, D., Lee, C.K.M. and Ho, W. "Multi-level genetic algorithm for the resource-constrained re-entrant scheduling problem in the flow shop", *Engineering Applications of Artificial Intelligence,* **26**(4), pp. 1282-1290 (2013).
22. Xu, J., Yin, Y., Cheng, T.C.E., et al. . "A memetic algorithm for the re-entrant permutation flowshop scheduling problem to minimize the makespan", *Applied Soft Computing,* **24** pp. 277-283 (2014).
23. Choi, S.-W. and Kim, Y.-D. "Minimizing total tardiness on a two-machine re-entrant flowshop", *European Journal of Operational Research,* **199**(2), pp. 375-384 (2009).
24. Lin, D., Lee, C.K.M. and Wu, Z. "Integrating analytical hierarchy process to genetic algorithm for re-entrant flow shop scheduling problem", *International Journal of Production Research,* **50**(7), pp. 1813-1824 (2012).
25. Xu, J., Lin, W.-C., Wu, J., et al. . "Heuristic based genetic algorithms for the re-entrant total completion time flowshop scheduling with learning consideration", *International Journal of Computational Intelligence Systems,* **9**(6), pp. 1082-1100 (2016).
26. Chamnanlor, C., Sethanan, K., Chien, C.-F., et al. . "Re-entrant flow shop scheduling problem with time windows using hybrid genetic algorithm based on auto-tuning strategy", *International Journal of Production Research,* **52**(9), pp. 2612-2629 (2014).
27. Chamnanlor, C., Sethanan, K., Gen, M., et al. . "Embedding ant system in genetic algorithm for re-entrant hybrid flow shop scheduling problems with time window constraints", *Journal of Intelligent Manufacturing,* **28**(8), pp. 1915-1931 (2017).
28. Waqas, U., Geilen, M., Kandelaars, J., et al. . "A re-entrant flowshop heuristic for online scheduling of the paper path in a large scale printer", *Design, Automation & Test in Europe Conference & Exhibition (DATE),* pp. 573-578 (2015)
29. Jeong, B. and Kim, Y.-D. "Minimizing total tardiness in a two-machine re-entrant flowshop with sequence-dependent setup times", *Computers & Operations Research,* **47** pp. 72-80 (2014).
30. Kim, H. and Lee, D.H. "Heuristic algorithms for re-entrant hybrid flow shop scheduling with unrelated parallel machines", *Proceedings of The Institution of Mechanical Engineers Part B-journal of Engineering Manufacture - PROC INST MECH ENG B-J ENG MA,* **223** pp. 433-442 (2009).
31. Lee, C.K.M., Lin, D., Ho, W., et al. . "Design of a genetic algorithm for bi-objective flow shop scheduling problems with re-entrant jobs", *The International Journal of Advanced Manufacturing Technology,* **56**(9), pp. 1105-1113 (2011).
32. Mousavi, S.M., Mahdavi, I., Rezaeian, J., et al. . "An efficient bi-objective algorithm to solve re-entrant hybrid flow shop scheduling with learning effect and setup times", *Operational Research,* **18**(1), pp. 123-158 (2018).
33. El-Khouly, I.A., El-Kilany, K.S. and El-Sayed, A.E. "Modelling and simulation of re-entrant flow shop scheduling: An application in semiconductor manufacturing", *International Conference on Computers & Industrial Engineering,* pp. 211-216 (2009)
34. Moghaddam, A., Yalaoui, F. and Amodeo, L. "A Genetic-based Algorithm to Find Better Estimation of Non-Dominated Solutions for a Bi-objective Re-entrant Flowshop Scheduling Problem with Outsourcing", *IFAC Proceedings Volumes,* **45**(6), pp. 1383-1388 (2012).
35. Mousavi, S.M., Mahdavi, I., Rezaeian, J., et al. . "Bi-objective scheduling for the re-entrant hybrid flow shop with learning effect and setup times", *Scientia Iranica,* **25**(4), pp. 2233-2253 (2018).
36. Behmanesh, R., Zandieh, M. and Hadji, M. "Multiple Resource Surgical Case Scheduling Problem: Ant Colony System Approach", *Economic Computation and Economic Cybernetics Studies and Research,* **54** pp. 251-268 (2020).
37. Behmanesh, R., Zandieh, M. and Hadji Molana, S.M. "The surgical case scheduling problem with fuzzy duration time: An ant system algorithm", *Scientia Iranica,* **26**(3), pp. 1824-1841 (2019).

38. Zhang, G. and Xing, K. "Differential evolution metaheuristics for distributed limited-buffer flowshop scheduling with makespan criterion", *Computers & Operations Research*, **108** pp. 33-43 (2019).
39. Chen, J.-S., Pan, J.C.-H. and Wu, C.-K. "Hybrid tabu search for re-entrant permutation flow-shop scheduling problem", *Expert Systems with Applications*, **34**(3), pp. 1924-1930 (2008).
40. Sangsawang, C., Sethanan, K., Fujimoto, T., et al. . "Metaheuristics optimization approaches for two-stage reentrant flexible flow shop with blocking constraint", *Expert Systems with Applications*, **42**(5), pp. 2395-2410 (2015).
41. Oyetunji, E.O. and Oluleye, A.E. "Minimizing Makespan on Single Machine with Release Dates", *International Journal of Advancements in Technology*, **2**(1), pp. 3-13 (2011).
42. Simon, D. "Biogeography-Based Optimization", *IEEE Transactions on Evolutionary Computation*, **12**(6), pp. 702-713 (2008).
43. Karaboga, D. "An Idea Based on Honey Bee Swarm for Numerical Optimization, Technical Report - TR06", *Technical Report, Erciyes University*, (2005).
44. Glover, F. "Future paths for integer programming and links to artificial intelligence", *Computers & Operations Research*, **13**(5), pp. 533-549 (1986).
45. Lawler, E.L. "Optimal Sequencing of a Single Machine Subject to Precedence Constraints", *Management Science*, **19**(5), pp. 544-546 (1973).
46. Bean, J.C. "Genetic Algorithms and Random Keys for Sequencing and Optimization", *ORSA Journal on Computing*, **6**(2), pp. 154-160 (1994).
47. Solimanpur, M. and Elmi, A. "A tabu search approach for cell scheduling problem with makespan criterion", *International Journal of Production Economics*, **141**(2), pp. 639-645 (2013).
48. Kaplan, A.C. and Unal, A.T. "A probabilistic cost-based due date assignment model for job shops", *International Journal of Production Research*, **31**(12), pp. 2817-2834 (1993).
49. Seidgar, H., Fazlollahtabar, H. and Zandieh, M. "Scheduling two-stage assembly flow shop with random machines breakdowns: integrated new self-adapted differential evolutionary and simulation approach", *Soft Computing*, 24(11), pp. 8377-8401 (2020).

Biography

Reza Behmanesh is currently an assistant professor in Naghshejahan Higher Education Institute, Isfahan, Iran. He received his BSc degree of mining engineering from Isfahan University of technology, Isfahan, Iran in 2001 and his MSc degree of Industrial engineering from Yazd University, Yazd, Iran in 2010. Also, he received his PhD degree of Industrial Engineering (Operation research and system engineering) from Science and Research Branch, Islamic Azad University, Tehran, Iran in 2017. He has three years job experience in EORC as a planner in maintenance workshop. His current research interests are swarm and evolutionary computations, scheduling, optimization techniques.

+989131860461
+983132730712
rezaehs@gmail.com
rezaehs@yahoo.com

r.behmanesh@khuisf.ac.ir

Kamran Kianfar is an Assistant Professor of Industrial Engineering in the Faculty of Engineering at University of Isfahan, Iran. He received his PhD from Department of Industrial and Systems Engineering at Isfahan University of Technology, Iran in 2014. He holds an MSc in Industrial Engineering from Amirkabir University of Technology, Iran and a BSc in Industrial Engineering from Isfahan University of Technology, Iran. His main areas of teaching and research interests include production planning and scheduling, combinatorial optimization, meta-heuristics and Supply Chain management.

+98-9132050419
+98-31-37934041
k.kianfar@eng.ui.ac.ir
kianfar.kamran@gmail.com

List of Tables

List of Figures

Table1. Researches on RFSS up to 2021

Authors	Year	Characteristics			Model		Objectives	Solution Method
		SDST[1]	TW[2]	RD[3]	SO[4]	MO[5]		
Chen et al	2006				●		C_{max}	Integer programming
Choi & Kim	2007				●		C_{max}	Branch & Bound
Choi & Kim	2009				●		C_{max}	Heuristics
Jing et al	2008				●		C_{max}	Heuristics
Chen et al	2008				●		C_{max}	Hybrid GA
Choi & Kim	2009				●		TT[6]	Branch & Bound
El-Khouly et al	2009					●	Utilization+SD[7]	Simulation
Kim & Lee	2009					●	TT+C_{max}	Heuristics
Lee et al	2011					●	TWT[8] +C_{max}	GA
Che et al	2012				●		Cycle Time	Heuristic
Moghadam et al	2012					●	TC+Cost	NSGA-II
Lin et al	2012				●		TWT	AHP+GA
Lin et al	2013				●		C_{max}	GA
Jeong & Kim	2014	●			●		TT	Branch & Bound
Xu et al	2014				●		C_{max}	MA
Chamnanlor et al	2014		●		●		C_{max}	HGA
Amin-Naseri et al	2015				●		C_{max}	SA+GA
Sangsawang et al	2015				●		C_{max}	GA+PSO
Waqas et al	2015	●			●		C_{max}	Heuristic
Xu et al	2016				●		TC[9]	GA
Chamnanlor et al	2017		●		●		C_{max}	GA+ACO
Huang et al	2017				●		C_{max}	Heuristics
Moussavi et al	2018	●				●	TT+C_{max}	GA
Wu et al	2018				●		C_{max}	SA+heuristics
Geng et al	2019					●	C_{max}+ TT + Energy	heuristics
Rifai et al	2021				●		TC+TT+C_{max}	SA+GA
Our Research				●		●	TU[10] +Idle time+TC+TT+C_{max}	BBO+ABC+TS

1- Sequence dependent setup times 2- Time window 3- Release date 4- Single objective
5- Multi objective 6- Total tardiness 7- Standard deviation 8- Total weighted tardiness
9- Total completion time 10- Total utility

Table2. The parameters of the example

Job i	A	B	C	D
p_i	5	4	3	6
r_i	2	8	3	16

Table3. Test cases and structure

Cases	Problem	Number of Jobs	Processing Time	Release Date
Case1	1	5	[1,15]	[1,50]
	2	10	[1,30]	[1,60]
	3	15	[1,30]	[1,60]
	4	20	[1,40]	[1,80]
Case2	5	30	[1,60]	[1,100]
	6	45	[1,80]	[1,150]
	7	60	[1,100]	[1,250]
Case3	8	80	[1,150]	[1,500]
	9	120	[1,300]	[1,1200]
	10	240	[1,500]	[1,1500]

Table 4. The validation of the BBO+ABC+TS algorithm

Problem	**GAMS (CPLEX)**	BBO+ABC+TS	GAP*	pOpt**	Run time (GAMS)	Run time (BBO+ABC+TS)	RCT***
1	**2.524231**	2.524231	0	1	1.12	0.06	0.05357
2	**4.670389**	4.670389	0	1	83	0.53	0.00638
3	**4.993787**	4.993787	0	1	934	0.47	0.00050
4	**5.3669**	5.36772	0.015	0.9	2258	0.65	0.00029
Avg.	---	**---**	**0.004**	**---**	**---**	**---**	**0.01519**

*. GAP= ($\frac{BBO,ABC,TS - GAMS}{GAMS}$ %)

**. The percent of optimum solutions in 10 runs

***. Relative Computational Time = ($\frac{BBO,ABC,TS}{GAMS}$ %)

Table 5. Setting parameters of algorithms

Case no.	Algorithms	(max-it , pop) for the largest problem in each case	Tabu_list [problems]*
Case 1	BBO	150, 80	-
	ABC	300, 50	-
	GA	250-150	-
	TS	20, 1	[23 – 95 – 218 – 390]
	BBO-ABC-TS	(45+5, 50)**	[23 – 95 – 218 – 390]
Case 2	BBO	300, 100	-
	ABC	500, 80	-
	GA	400-150	-
	TS	1, 35	[885 – 2003 – 3750]
	BBO-ABC-TS	(80+6, 40)	[885 – 2003 – 3750]
Case 3	BBO	300, 150	-
	ABC	700, 100	-
	GA	900, 200	-
	TS	1, 50	[6360 – 14340 – 57480]
	BBO-ABC-TS	70+1, 250	[6360 – 14340 – 57480]

* Four problems for the first case, and three problems for the second and the third cases
** 45+5 means 45 generations for BBO and ABC, and 5 iterations for TS

Table 6. The RPD value of multi-objective (mean and standard deviation) of the solutions of the algorithms

Problem	ABC			BBO			GA			TS			BBO+ABC+TS		
	Mean	St. dev.	pOpt	Mean	St. dev.	pOpt	Mean	St. dev.	pOpt	Mean	St. dev.	pOpt	Mean	St. dev.	pOpt
P1	0.000	0.000	1.0	0.000	0.000	1.0	0.000	0.000	1.0	0.000	0.000	1.0	**0.000**	0.0	**1.0**
P2	0.000	0.000	1.0	0.070	0.140	1.0	0.000	0.000	1.0	0.033	0.080	1.0	**0.000**	0.0	**1.0**
P3	0.000	0.000	1.0	0.017	0.031	0.8	0.000	0.000	1.0	0.000	0.000	0.8	**0.000**	0.0	**1.0**
P4	0.034	0.074	0.5	0.009	0.015	0.7	0.106	0.272	0.8	0.000	0.000	0.9	**0.000**	0.0	**0.9**
P5	0.169	0.034	0.0	0.065	0.121	0.1	0.032	0.032	0.1	0.010	0.009	0.2	**0.004**	0.0	**0.5**
P6	0.768	0.154	0.0	0.112	0.132	0.0	0.112	0.079	0.0	0.088	0.053	0.0	**0.000**	0.0	**0.3**
P7	1.299	0.125	0.0	0.108	0.085	0.1	0.278	0.116	0.0	0.158	0.092	0.0	**0.000**	0.0	**0.3**
P8	2.064	0.224	0.0	0.056	0.047	0.0	0.247	0.147	0.0	0.180	0.062	0.0	**0.016**	0.0	**0.1**
P9	2.397	0.340	0.0	0.230	0.165	0.0	0.290	0.149	0.0	1.094	0.135	0.0	**0.000**	0.0	**0.1**
P10	5.639	0.283	0.0	0.101	0.070	0.0	0.766	0.064	0.0	2.611	0.695	0.0	**0.000**	0.0	**0.1**

Table 7. The Computational Time (mean and standard deviation) of the solutions of the algorithms

Problems	ABC		BBO		GA		TS		BBO+ABC+TS	
	Mean	St. dev.	Mean	St. dev.	Mean	St. dev.	Mean	St. dev.	Mean	St. dev.
P1	0.309	0.009	0.224	0.005	0.232	0.010	0.088	0.001	**0.060**	0.001
P2	1.313	0.055	1.558	0.034	1.237	0.010	0.718	0.007	**0.543**	0.024
P3	2.196	0.167	1.669	0.074	1.525	0.094	0.746	0.121	**0.475**	0.025
P4	4.242	0.124	2.213	0.153	5.144	0.066	1.474	0.078	**0.653**	0.068
P5	8.855	0.535	3.134	0.266	8.008	0.404	5.687	0.350	**2.709**	0.126
P6	10.456	0.132	5.939	0.123	10.174	0.222	20.945	0.210	**4.276**	0.100
P7	22.302	0.235	12.710	0.086	17.086	0.127	56.146	0.247	**11.865**	0.119
P8	35.258	0.677	34.144	3.447	45.236	0.761	134.75	0.906	**25.651**	0.378
P9	67.873	0.378	48.234	0.610	78.875	0.373	445.25	0.738	**20.874**	0.316
P10	131.05	0.583	70.045	0.163	174.32	3.174	305.47	3.759	**30.449**	1.116

Table 8. ANOVA test for comparison of multi-objective and run time of algorithms

***Hypothesis*:**	H_0: $\mu_{ABC} = \mu_{BBO} = \mu_{GA} = \mu_{TS} = \mu_{BBO+ABC+TS}$				
	H_1: Otherwise				
Source of variation	DF	SS	MS	F	*P*-value
Algorithm (Obj)	4	100.928	25.232	33.996	0.000
Error	495	367.392	0.742		
Total	499	468.319			
Result:	Reject H_0				
Algorithm (CT)	4	480011.024	120002.756	22.098	0.000
Error	495	2688067.883	5430.440		
Total	499	3168078.907			
Result:	Reject H_0				

Table 9. Comparison Tests for five algorithms in the convergence objective value

Algorithm (A)	Algorithm (B)	Mean Difference (A-B)	*P*-value		Result
			Tamhane	Dunnett T3	
BBO+ABC+TS	ABC	-1.23*	0.000	0.000	BBO+ABC+TS < ABC
BBO+ABC+TS	BBO	-.075*	0.000	0.000	BBO+ABC+TS < BBO
BBO+ABC+TS	GA	-.18*	0.000	0.000	BBO+ABC+TS < GA
BBO+ABC+TS	TS	-.41*	0.000	0.000	BBO+ABC+TS < TS
Result:			$\mu_{BBO+ABC+TS} < \mu_{Others}$		

*. The mean difference is significant at the 0.05 level

Table 10. Comparison Tests for five algorithms in the run time value

Algorithm (A)	Algorithm (B)	Mean Difference (A-B)	*P*-value		Result
			Tamhane	Dunnett T3	
BBO+ABC+TS	ABC	-18.63*	0.000	0.000	BBO+ABC+TS < ABC
BBO+ABC+TS	BBO	-8.23*	0.018	0.018	BBO+ABC+TS < BBO
BBO+ABC+TS	GA	-24.42*	0.000	0.000	BBO+ABC+TS < GA
BBO+ABC+TS	TS	-87.37*	0.000	0.000	BBO+ABC+TS < TS
Result:				$\mu_{BBO+ABC+TS} < \mu_{Others}$	

*. The mean difference is significant at the 0.05 level

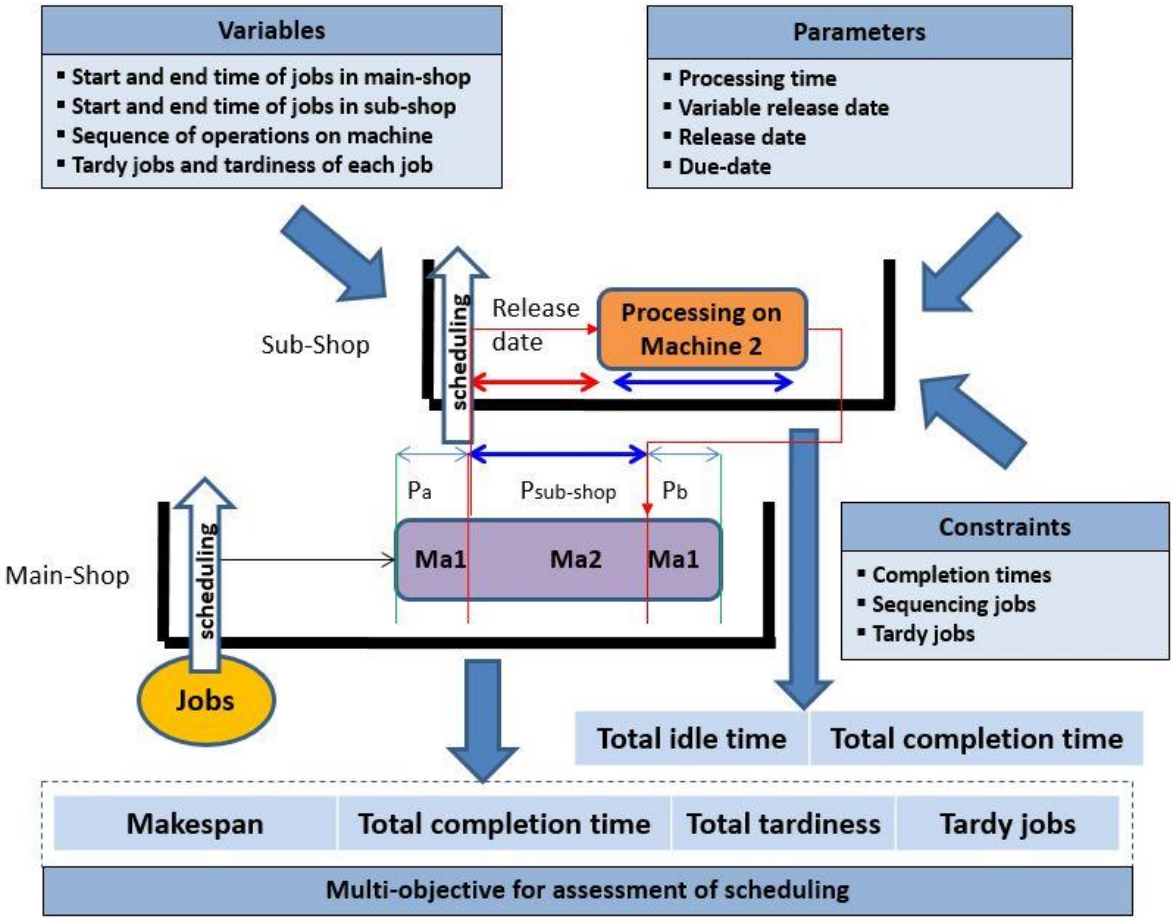


Fig. 1. The schematic model of re-entrant flow-shop

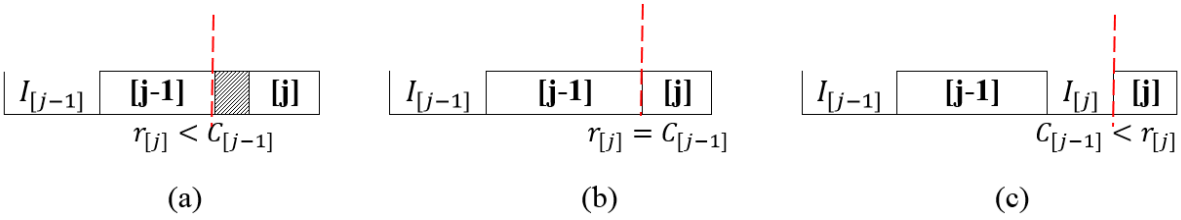


Fig. 2. All states of idle time after job releasing

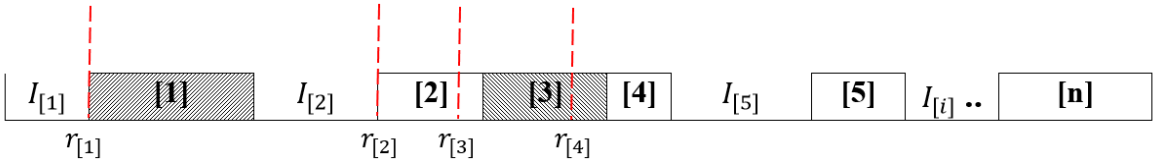


Fig. 3. General Gantt chart

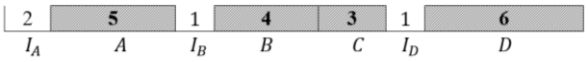


Fig. 4. Sequence of A-B-C-D

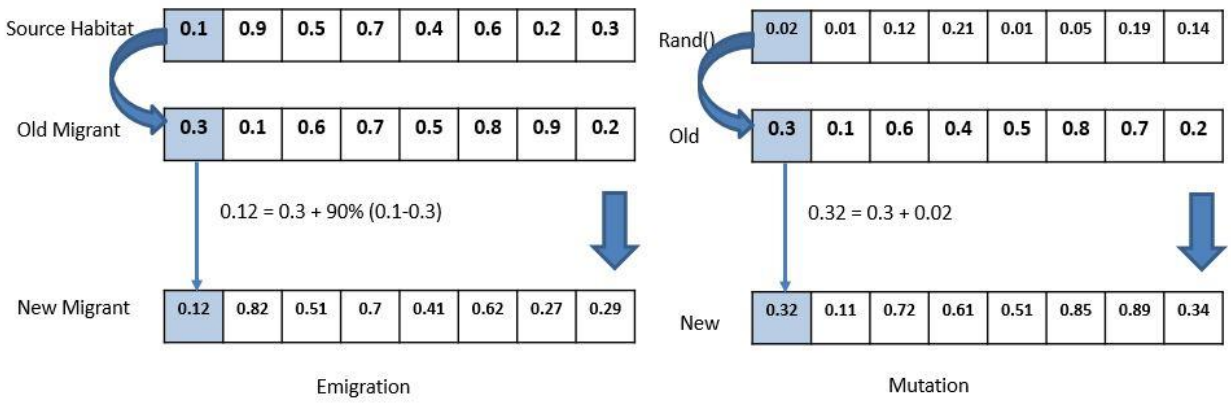


Fig. 5. An illustration of the operators of BBO on schedule solution

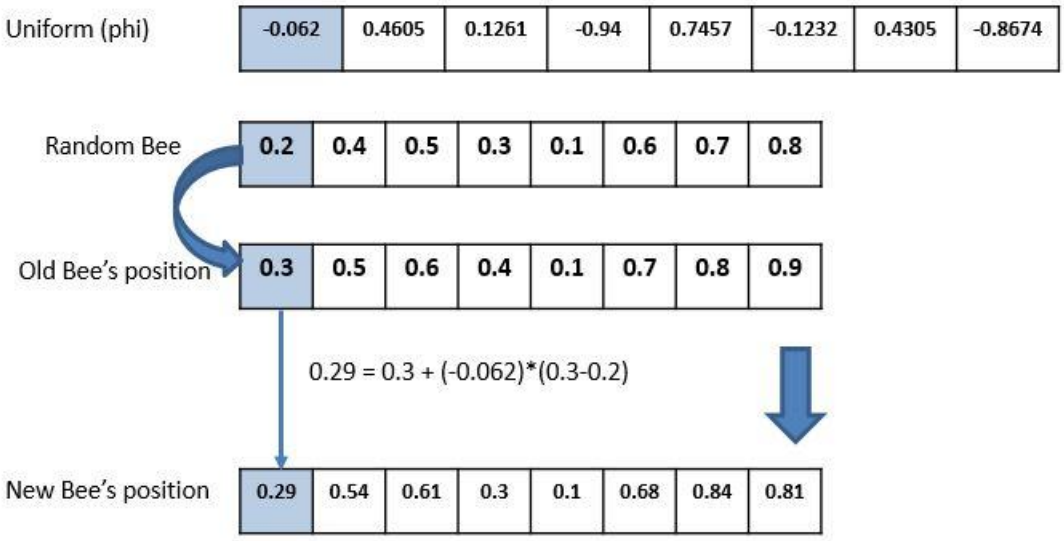


Fig. 6. An illustration of the operator of ABC on schedule solution

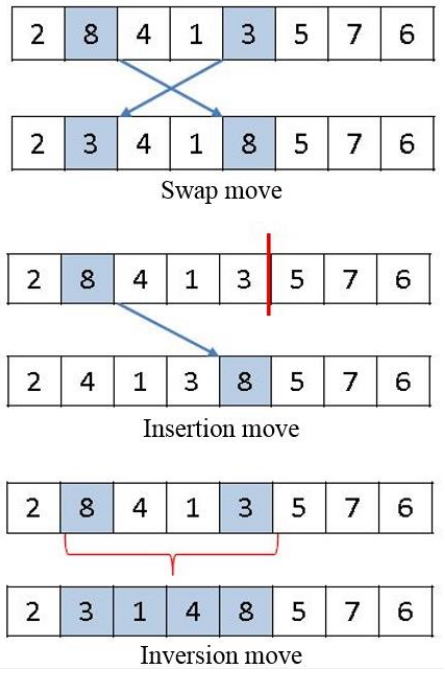
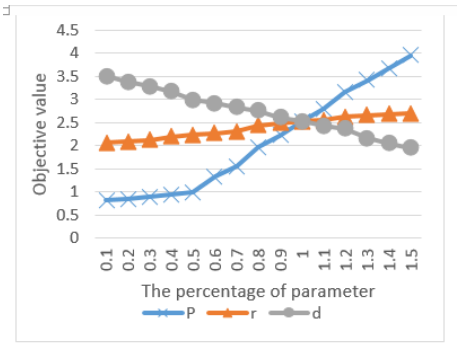


Fig. 7. An illustration of the movements of TS on schedule solution



(a) problem 1

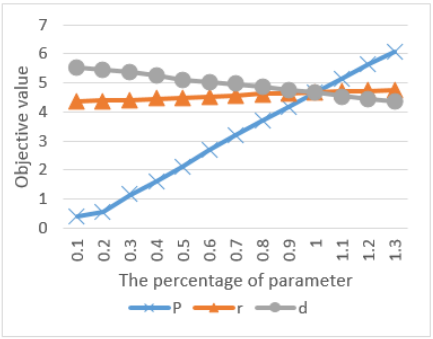


Fig. 8. Sensitivity analysis of multi-objective versus parameters

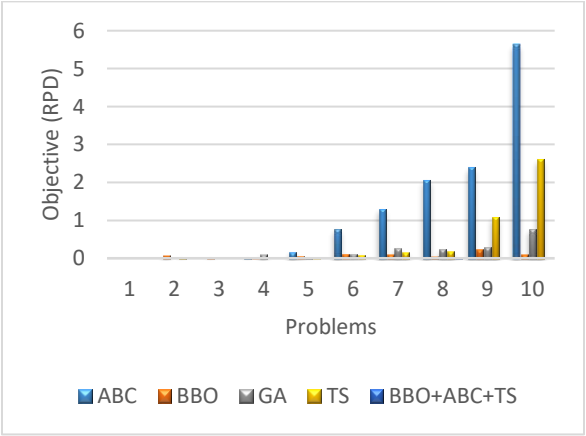


Fig. 9. Bar-charts of the objective (RPD) of the algorithms for each problem

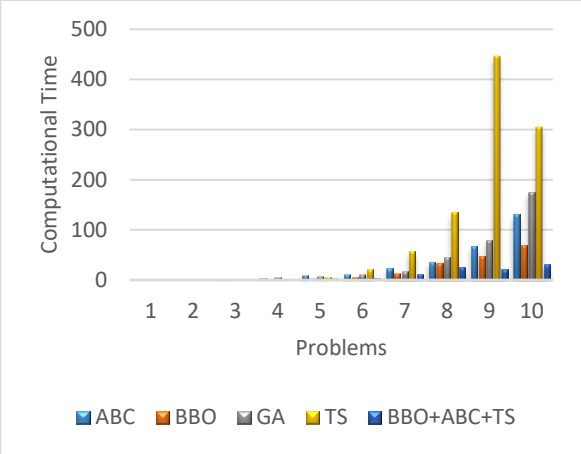


Fig. 10. Bar-charts of the computational time of the algorithms for each problem

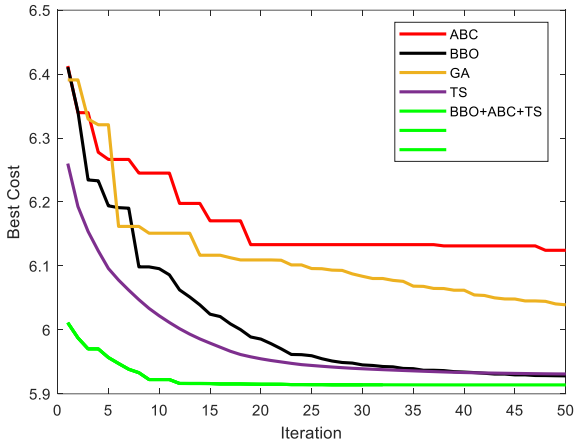
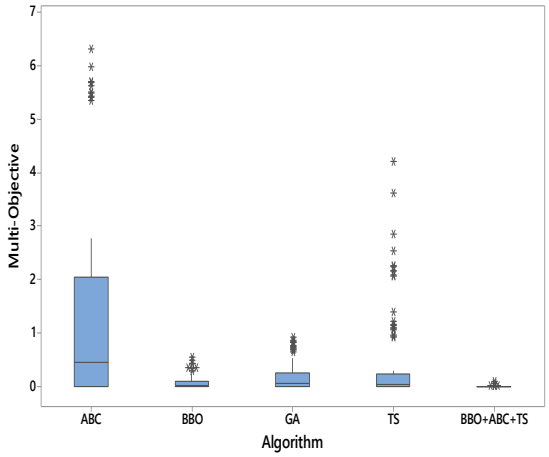
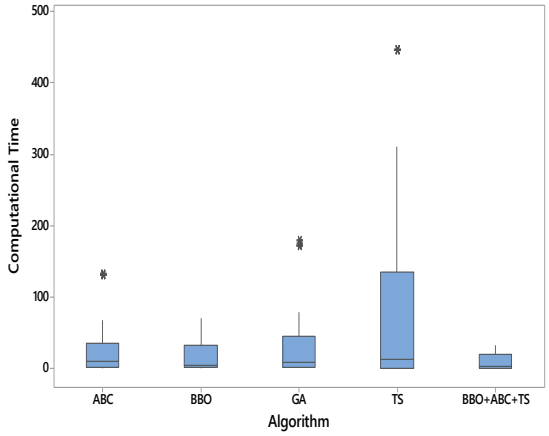


Fig. 11. Convergence plot of all algorithms for medium case (Problem 6)



(a)



(b)

Fig. 12 Box-plot of Multi-objective (a) and computational time (b) for algorithms