International Journal of Industrial Engineering Computations 10 (2019) 331–348

Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

A hybrid genetic-gravitational search algorithm for a multi-objective flow shop scheduling problem

T.S. Lee[a*], Y.T. Loong[a] and S.C. Tan[b]

[a]*Faculty of Engineering and Technology, Multimedia University, Melaka, Malaysia*
[b]*Faculty of Information Science and Technology, Multimedia University, Melaka, Malaysia*

C H R O N I C L E

Article history:
Received September 17 2018
Received in Revised Format
January 12 2019
Accepted February 27 2019
Available online
February 27 2019

Keywords:
Dispatching rules
Multi-objective flow shop scheduling
Genetic algorithm
Gravitational Search algorithm

A B S T R A C T

Many real-world problems in manufacturing system, for instance, the scheduling problems, are formulated by defining several objectives for problem solving and decision making. Recently, research on dispatching rules allocation has attracted substantial attention. Although many dispatching rules methods have been developed, multi-objective scheduling problems remain inherently difficult to solve by any single rule. In this paper, a hybrid genetic-based gravitational search algorithm (GSA) in weighted dispatching rule is proposed to tackle a scheduling problem by achieving both time and job-related objectives. Genetic algorithm (GA) is used to select two appropriate dispatching rules to combine as a weighted multi-attribute function, while the GSA is used to optimize the contribution weightage of each rule in each stage of the flow shop. The results show that the proposed algorithm is significantly better than the traditional dispatching rules and the rules allocation algorithm. The proposed algorithm not only improved the quality of the schedule in multi-objective problems but also maintained the advantages of traditional dispatching rules in terms of ease of implementation.

1. Introduction

Production scheduling is the process of allocating limited resources, which include the manpower, machines or utilities, with respect to various products, in a limited time (Pinedo, 2008). This process involves a search for job order and job sequencing to obtain an optimal schedule with the highest utilization and efficiency. Flexible flow shop is one of the major manufacturing system configurations which combine the conventional flow shop and parallel machine system. This type of scheduling system can be defined as multiprocessor of flow shop with parallel machines (Jungwattanakit, 2008). A great number of semiconductor industries implement flexible flow shop in a wide range of production processes in order to gain flexibility in scheduling and control. Flexible flow shop scheduling offers the flexibility of production and job sequencing with more than one machine in a single stage. The duplication of the machines in certain stages can enhance the overall capacities, improve flexibility and reduce bottlenecks in some productions (Khalouli et al., 2010). However, many semiconductor industries are facing high tardiness and high makespan scheduling problem. It will leave a significant impact on

* Corresponding author Tel.: +606 252 3363; Fax: +606 231 6552
E-mail: tslee@mmu.edu.my (T.S. Lee)

2019 Growing Science Ltd.
doi: 10.5267/j.ijiec.2019.2.004

manufacturing costs. Usually industrial scheduling problems are aimed to achieve several objectives. However, single objective function lumps all different targets into one and aims to obtain the "best" solutions. This type of optimization usually cannot provide a set of alternative solutions with conflicting objectives. However, multi-objective optimization provides a set of compromised solutions and it is more realistic in reality. From the literature review (Ruiz & Vazquenz-Rodriguez, 2010), most of the dispatching rules related studies are single objective. However, single objective might not be sufficient to describe the time-related indicators (such as makespan, completion time, flow time, etc) and job-related targets (such as tardiness and earliness). Apart from these targeted objectives, other objectives such as workload (Pérez & Raupp, 2014) and machine effectiveness (Li, 2014) are also considered for optimization. Hence, multi-objective system is one step forward to the real application. In this study, a multi-objective function is proposed to optimize both time-related and job-related objectives.

Dispatching rules method is popular due to its less computational time and is able to provide good solution (Chen et al., 2013). Dispatching rules (also known as, scheduling policies and prioritization rules) are used to prioritize jobs that are queued for processing on a machine (Ruiz & Vazquenz-Rodriguez, 2010). The performance of the scheduling planning highly depends on the dispatching rule that is used (EL Bouri & Amin, 2015). Every dispatching rule is only effective for certain performance criterion/objectives. This raises difficulties when multi-objectives scheduling problem is defined. In such a multi-criteria environment, selection of appropriate dispatching rules which could best satisfy the given performance criteria becomes a scheduling challenge. In recent decades, different artificial intelligent optimization methods are proposed to aid in selection of appropriate rules in the flow shop (Li, 2014). From literature, intelligent approach is able to solve different scheduling application effectively (Ribas et al., 2010). Hence, in this study, an enhancement method in dispatching rules is proposed and compared with the conventional dispatching rules and common rules allocation method. A hybrid genetic based gravitational search algorithm (GSA) in weighted dispatching rule is proposed for the bi-objective flow shop scheduling problem. Genetic algorithm (GA) is used to select two appropriate dispatching rules to combine as a weighted multi-attribute function with prioritize index to trade with the multi-criteria environment, while the GSA is used to optimize the contribution weightage of each rule in each stage of the flow shop based on the selected rules and the prioritize index of the objectives functions. GSA is selected in this study because of its ability to find near global optimum solution which differs from other nature inspired algorithms (Kumar & Sahoo, 2014). Therefore, the proposed hybrid algorithm is flexible in various objectives with different prioritize index. It can maintain the easiness of implementation and low computational effort of the traditional dispatching rules as well.

The remainder of this paper is organized as follows: The objectives functions and system constraints are described in second section. Next, the artificial intelligent approach in solving scheduling application is discussed followed by the methodology framework for the proposed hybrid Genetic based Gravitational Search Algorithm in weighted dispatching rules. Then, the results of the proposed algorithm are discussed and lastly a conclusion is presented.

2. Multi-Objective Scheduling Problems

In the real world application, objective functions of industrial scheduling problems are commonly divided into different aspects such as job-related, time-related and qualitative-related objectives. Among all, time-related and job-related objectives are adopted in this study. The multi-objective function has been presented in Eq. (3) with insertion of an objective prioritize index, λ.

2.1 Notations

To describe the objective functions, the problem constraints and the algorithm, certain notations are introduced as follows:

a_i^d	Acceleration of agent *i* in dimension *d*
C_{mean}	Mean of excess time
C_j	Completion time of job *j*
C_{jk}	Completion time of job *j* in stage *k*
C_{m_jk}	Completion time of job *j* on machine *m* in stage *k*
d_j	Due date of job *j*
dim	Dimension of the agent
$F_{ij}^d(t)$	The force acting on mass *i* from mass *j* at a specific time *t*
f_{obj}	Objective function
fit_i	Fitness value of the agent *i*
G_j	Boolean value for job *j*, 1 represents tardy job, 0 represent not a tardy job
$G(t)$	The gravitational constant at time *t*
h	Number of machine per production line
IT_{jk}	Idle time of job *j* in stage *k*
j	Job index
k	Number of stage
L	Location of the X agent
M_{aj}	The active gravitational mass
M_{pj}	The passive gravitational mass
M_{ii}	The inertial mass of the i^{th} agent
m	Machine index
m_j	Machine index with respect to job *j*
n	Number of jobs
N	Number of X agent
η_{mean}	Mean of tardy jobs with respect to total number of jobs
p	Population size
P_{jk}	Processing time of job *j* in stage *k*
q	Number of genes
R_{ij}	The Euclidean distance between agents *i* and *j*
ST_{m_jk}	Start time of job *j* on machine *m* in stage *k*
t	Time
T_j	Tardiness
U_{mk}	Boolean variable, 0 if job *j* is the first job on machine *m* at stage *k* and >0 otherwise
V_i^d	Velocity of agent *i* in dimension *d*
w	Total number of stages
X_j^d	Location of agent *j* in dimension *d*
λ	Contribution weight to objective function, C_{mean} and η_{mean}
ε	A small constant

2.2 Objective function and problem constraints

A set of *n* jobs has to be processed in a flexile flow shop (FFS) production setting. The FFS consists of a set of $k \geqslant 2$ stages or machine centers. At least one of these stages include more than one machine. At each stage, each machine can process one job at a time. There is no specific assumption on the similarity of the machines at each stage. A job consists of several operations to be performed by one machine on each stage. The job *j* to be performed at the *k*th stage requires $P_{jk} \geqslant 0$ units of time (processing time) and can start only after the completion of the job from previous stage according to the stage sequence of this job.

Minimization optimization is used in this study to seek for a schedule that minimizes a combination of mean excess time (time-related objective) and the mean of tardy jobs (job-related objective). Let the mean excess time of all the jobs be C_{mean}, then

$$C_{mean} = \frac{\sum_{j=1}^{n} \frac{C_j - \sum_{k=1}^{w} P_{jk}}{\sum_{k=1}^{w} P_{jk}}}{n}, \quad (1)$$

where C_j represents the total completion time of the job j and $\sum_{k=1}^{w} P_{jk}$ is the sum of processing time of job j from stage 1 to stage w. Moreover, let $G_j = 1$ if due date for job j is smaller than the completion time C_j of job j, otherwise $G_j = 0$. The mean of tardy jobs (η_{mean}) is defined as

$$\eta_{mean} = \frac{\sum_{j=1}^{n} \frac{G_j}{n}}{n} \quad (2)$$

Thus, the minimization objective function value is defined by

$$\min \quad \lambda C_{mean} + (1 - \lambda)\eta_{mean} \quad (3)$$

where $0 \leq \lambda \leq 1$. λ denotes the weight (or relative importance) given to C_{mean} and η_{mean}.

The general FFS scheduling problem is described as follows. For the first stage, Eq. (4) gives the completion time of the first job on each machine, i.e., completion time of job j at stage 1 is equal to the processing time of job j at stage 1. When it is not the first job of the machine, the completion time of job j is the summation of the processing time of job j and the previous completion time in the same machine. Eq. (5) denotes the idle time is equal to zero at stage 1. Eq. (6) and Eq. (7) denote the idle time and completion time at the second stage onwards. The idle time is equal to zero when it is the first job processing in the machine. While the idle time for the second job onwards at stage k, is the difference in time of the previous completion time of the machine m corresponding to job j at the current stage and the completion time of job j in previous stage. Eq. (7) determines the completion time of job j at stage k; while the first job on the machine is equal to the summation of the processing time of job j at stage k, the completion time of the machine m corresponding to job j at the current stage and the idle time. In contrast, when the job is not the first job in the machine, the completion is equal to the summation of the processing time of job j at stage k, completion time of the machine m at the previous stage, the idle time and the time difference between the completion time of previous job in the same machine m and the completion time of machine m in the previous stage.

This differs from some exisiting methods in the literature (e.g. Choi & Wang, 2012; Wang & Choi, 2012; Wang & Choi, 2014), since the idle time is used in this study to calculate the completion time in second stage onwards, while the modelling developed by Choi and Wang (2012) uses the maximum time to determine the completion time. There is a literature using job ready time to calculate the completion time in second stage (Kim et al., 2007). However, the waiting time might not be considered while the job is in queue. In the literature discussed by Jungwattanakit (2008), the waiting time calculated by the setup time plus the processing time and a very big constant to force the second job follow the first job by at least the processing time of first job and the setup time. However, the condition where the first job is in the middle of processing is not being considered.

For stage 1 (k=1),

$$C_{jk} = \begin{cases} P_{jk} & , \quad if\ U_{mk} = 0 \\ P_{jk} + C_{m_{(j-h)}(k)} & , \quad if\ U_{mk} > 0 \end{cases} \quad (4)$$

$$IT_{jk} = 0 \quad (5)$$

For stage 2 onwards (k>1),

$$IT_{jk} = \begin{cases} 0 & , \ if\ U_{mk} = 0\ OR\ IT_{jk} < 0 \\ C_{(j-h)k} - C_{m_j(k-1)} & , \ if\ U_{mk} > 0 \end{cases} \quad (6)$$

$$C_{jk} = \begin{cases} P_{jk} + C_{m_j(k-1)} + IT_{jk} & , \ if\ U_{mk} = 0 \\ P_{jk} + C_{m_j(k-1)} + IT_{jk} + \left[C_{(j-h)k} - C_{m_j(k-1)}\right] & , \ if\ U_{mk} > 0 \end{cases} \quad (7)$$

Fig. 1 shows an example of a scheduling problem consisting of 6 jobs flow in 3 stages production lines with 3 parallel machines at each stage. C_j represents the total completion time of job j while C_{jk} is the completion time in stage k. $C_{m_j(k-1)}$ is the completion time of job j in stage k-1 corresponding to machine m which is similar to $C_{j(k-1)}$. Additional parameter m is added in the symbol because the job might not flow in parallel direction; hence we need to identify which machine is operating the job in the previous stage. For example, when job 4 in stage 2 machine m_{22} is evaluated, the completion time $C_{m_j(k-1)}$ of job 4 in stage 1 is operated in m_{11}. Lastly, the $C_{(j-h)k}$ is the completion time of the machine before the next job. For example, the $C_{(j-h)k}$ of job 4 is the completion time of the machine m_{22} in the first round.

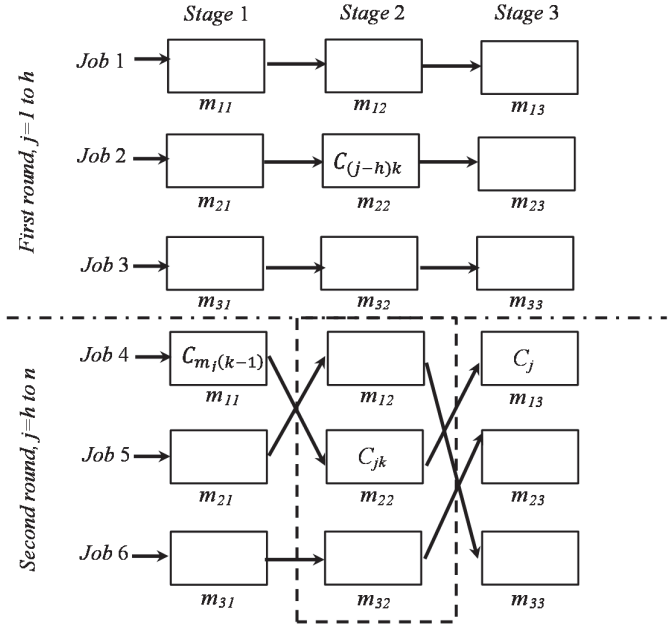

Fig. 1. Illustration of a sample problem in 3 stages production system

Constraint sets (8) and (9) determine the correct value of the tardiness (Tj). Constraint set (8) determines the tardiness is equal to the completion time of job j minus the due date D of job j. If the tardiness is larger than zero, the job is tardy $G_j = 1$; otherwise this job is not tardy $G_j = 0$. η_T is the total number of tardy jobs.

$$T_j \geq C_j - D_j \quad (8)$$

$$if\ T_j \geq 0\ , then\ G_j = 1 \quad (9)$$

$$\eta_T = \sum_{j=1}^{n} G_j \quad (10)$$

$$G_j \in \{0,1\} \quad (11)$$

Eq. (12) stipulates that each of the parallel machines at a stage takes equal time to process the same job (Wang & Choi, 2012). Eq. (13) requires the processing sequence of each stage to satisfy the processing time and ensures non-negative start time of job processing. Eq. (14) guarantees that each machine can process only one job at a time.

$$P_{jk_{m_1}} = P_{jk_{m_2}} = P_{jk} \quad ,(m_1, m_2) \in M_k \tag{12}$$

$$ST_{m_{1j}(k+1)} = P_{jk} + IT_{j(k+1)} \quad ,where\ ST_{m_jk} \geq 0 \tag{13}$$

$$\left[\left(ST_{m_{j_2}k} - ST_{m_{j_1}k}\right) \geq P_{jk_{m_1}}\right] \tag{14}$$

3. Artificial Intelligent Approach

In this study, Genetic Algorithm (GA) and Gravitational Search Algorithm (GSA) are used as enhancement method in dispatching rules to form a complete scheduling framework. Artificial intelligent approach has been used for this purpose in many ways. Korytkowski et al. (2013a) proposed a dispatching rules allocation method by introducing an evolutionary heuristic method. Korytkowski et al. (2013b) also proposed another heuristic method based on ant colony optimization to determine the suboptimal allocation of dynamic multi-attribute dispatching rules to maximize the performance of a job shop system. Ant colony optimization was also used to select heuristic rules for production and transportation scheduling (Tian et al., 2018). Jayamohan and Rajendran (2000) investigated the effectiveness of two approaches where one approach advocated the possibility of using different rules in various stages, while another approach suggested using the same rule at all stages of the flow shop. Nguyen et al. (2013) developed an iteratively dispatching rule by using genetic programming method. Xu (2013) proposed an immune algorithm to solve the scheduling problem in flexible flow shop based on several novel dispatching rules. From the literature, the dispatching rules method is still widely used due to the simplicity and effectively by different enhancement method. However, there are lacks of studies in multi-objective flexible flow shop problem. Hybridization of optimization algorithms helps in covering more areas of complex application. Hence, both Genetic Algorithm (GA) and Gravitational Search Algorithm (GSA) are selected as the optimization tools in this study due to their ability in searching near global optimum solution.

3.1 Genetic Algorithm (GA)

A GA starts with the creation of a population of randomly generated candidate solutions (called chromosomes). The first step in constructing the GA is to define an appropriate genetic representation (coding). The goodness of each chromosome in the population is measured using a fitness function. Fitness values are referred to determine which of the chromosomes are selected to produce offspring or survive into the next generation. If the optimization mode is minimization, the chromosome that is more fit (with a smaller objective function value) is selected as a parent chromosome to produce different offspring. The offspring from reproduction are then further perturbed by mutation (Spears, 2000). These cycles of selection, reproduction, mutation, and evaluation are repeated until the optimization criterion is reached (Simon, 2013; Liptak, 2005; Abraham et al., 2008). A wide range of application is successfully solved by using GA due to their simplicity and ease of operations characteristic (Goldberg, 1989). Jungwattanakit (2008) proposed an iterative GA-based method for bi-objectives problem by using constructive algorithm in population selection. There are great numbers of GA application in scheduling purpose; however, GA is used to obtain the complete schedule where the formulation and the decoding method would be complex. Hence, there are researchers who have proposed some hybrid algorithm by simplifying the formulation. For example, Morita and Shio (2005) proposed a new hybrid method using GA to improve the bound calculation. Rodriguez and Salhi (2005) used GA in the first stage and dispatching rules method in other stages for the FFS problem. However, those algorithms are usually implemented in single objective problems.

3.2 Gravitational Search Algorithm (GSA)

Gravitational Search Algorithm (GSA) is based on the Law of Gravity and Law of Motion (Rashedi et al., 2009; Eldos & Qasim, 2013). According to the Law of Gravity, lighter objects will be attracted towards heavier objects by gravitational forces. The heavier objects correspond to good solutions move

slower than lighter objects (Singh & Deep, 2015). Each object represents a solution and the algorithm is navigated by properly adjusting the gravitational and inertial mass. It starts with the generation of an initial population and followed by the evaluation of fitness for each individual (agent) in the population. From the evaluation, the agents' mass are updated. Then, the force, the velocity and the acceleration for that particular agent are calculated. The agents will move towards a new position (new solution). This process is repeated until the stopping criterion is achieved.

GSA has been widely used in mathematic area, data mining, pattern recognition and various engineering application due to its capability of reaching the optimum solution. However, there are only a handful of studies using GSA in scheduling purpose especially in flow shop production to obtain a complete schedule. GSA has been proven to outperform other nature inspired algorithms in terms of converging speed and local minima avoidance, and could generate better quality solution within shorter computational time and stable convergence characteristics (Sabri et al., 2013). The iteration method of GSA by changing the velocity and position of the agents with non-randomize method provides a better searching ability compared with some other algorithms. The structural of GSA also provides a clear and flexible problem representation where the environment of the problem can be simply understood. Therefore, GSA and GA are used in this study as part of the proposed framework by enhancing conventional dispatching rules method.

4. Proposed Genetic based Gravitational Search Algorithm in weighted Dispatching Rules (GA-GSA-WDR)

In this section, GA and GSA are introduced to replace the traditional dispatching rules prioritization method in flow shop scheduling problem. GA is used to select the appropriate rules combinations while GSA algorithm is used to optimize the weightage of the flow shop at each stage. The final schedule is constructed by using the mean of excess time and the mean of tardy jobs as the objectives function.

4.1 GA-GSA-WDR algorithm

In the proposed algorithm, the GA rules selection and GSA weightage optimization are combined to obtain a more flexible schedule. Two selected dispatching rules by using GA are combined and form a single weightage function shown in Eq. (15). These weightages (L_f) are optimized by GSA in each stage of the flow shop system based on the objective functions explained earlier in section 2.2. The optimized weightages for each stage is used to prioritize the jobs.

Compared with the recent rules allocation method discussed in section 3, the prioritize value is continuous in the proposed method. Rules allocation method provides a ranking at each stage by appropriate rules selection, however, every single dispatching rule only corresponds to a respective objective. For example, EDD is effective when tardiness objective is used. In this case, bi-objectives problem by combining both time-related and job-related objectives will become ineffective by using rules allocation method. Besides, the prioritize value obtain by rules allocation method is discrete, where the responding solution is minimum when the weightages of the objectives in bi-objectives are changing. The proposed method can be more flexible and solve the problem by optimizing the rules weightages to respond to the change of the objectives index. In other words, two stage optimization levels are used to find more compromised solutions for multi-objective scheduling problem. Fig. 2 illustrates the flow of the algorithm.

$$min, Z_j = L_f(Rule\ 1) + \left(1 - L_f\right)(Rule\ 2). \tag{15}$$

p population set of initial chromosomes (two selected rules in a single chromosome) are optimized by d dimension (number of stages) of i agents. The locations X in d dimension of i agents are corresponding to the optimal weightages. Two levels of iteration loops are repeated (shown in Fig. 2). The first loop

(Loop A) is in the GSA optimization to obtain the optimal weightage for each rules combination. Another loop (Loop B) is repeated by GA iteration (chromosome crossover and mutation) to obtain an optimal combination of rules until stopping criteria are achieved. This method can ensure the flexibility of the scheduling with respect to the objectives index, λ and maintain the room of improvement by introducing more and effective rules. The detailed steps are explained in the following section.

4.2 Problem identification of Case Study

A scheduling problem from a mechatronic manufacturing company is used to illustrate the flow of the framework. The company is located in Bangi, Malaysia. The company produces a variety of products using milling, turning and bending machines. A part of the productions were considered in this research work. A total number of 10 jobs were investigated in 3 stages with 3 parallel machines available in each stage. The 3 stages were the milling, turning and finishing processes. The products were produced in batches. The processing time for 10 jobs in each stage is listed in Table 1. The company is only practicing manual scheduling prepared by a production planner. The planner always refers to FIFO (first in first out) and EDD (earlier due date) to arrange the job sequence, neither heuristic nor algorithm is used in schedule planning. In this study, a new sequence of job is recommended by using proposed model, which is then compared with the job sequences obtained by applying GA and 7 dispatching rules.

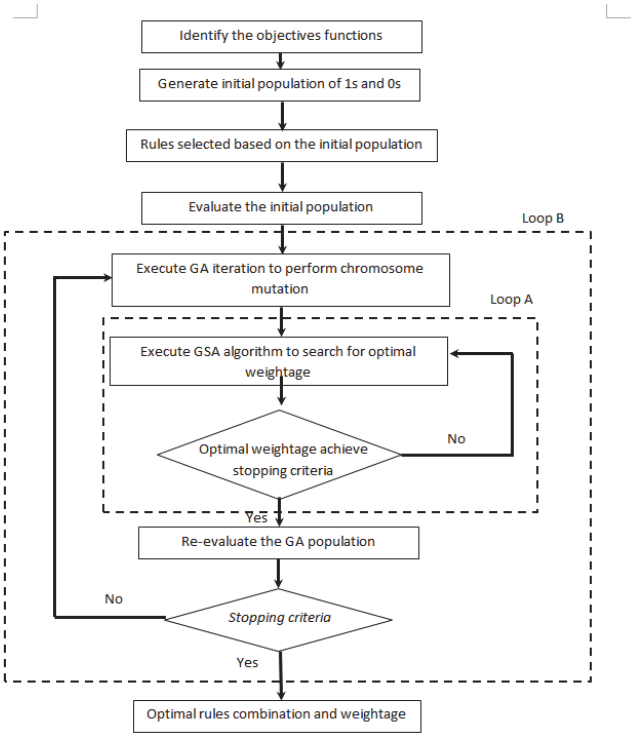

Fig. 2. The flow of the proposed algorithm

Table 1
Expected processing time for 10 jobs in 3 stages

		Stage 1 (Turning)	Stage 2 (3 Axis Milling)	Stage 3 (Surface Finish/Hear Treatment)
Job	Batch size	Expected Processing Time		
1	100	28	14.91	20.422
2	50	8.2	6	14.3
3	80	11.5	17	41
4	100	25.8	17.8	15.8
5	100	14.8	11	15.398
6	50	16	3.5	26.3
7	50	7	7	9
8	41	5.5	26.2	8.1
9	40	6.5	21.5	65.518
10	50	6.8	6.8	13.3

4.3 Rules Selection of Genetic Algorithm

In this study, a *p* population size is defined as the number of chromosomes (candidate solutions) with *q* number of genes (rules). Each chromosome comprises a series of binary codes where the sum of the codes must be equal to 2 (Fig. 3). In other words, only 2 rules will be selected from each chromosome in a population. The dispatching decision for each gene is independent of each other and the solution is only dependent on the number of candidate rules. The positions of genes with 1s represent the corresponding number of rules. Fig. 3 illustrates an example of the chromosome. This framework can be used to select more than 2 rules and different number of rules can be applied. In this study, 7 candidates dispatching rules are used for illustration and the details of these rules and their associated notation are shown in Table 2 (Korytkowski et al., 2013a,b; Joo et al., 2013). Some new effective rules are encouraged to be introduced in this framework.

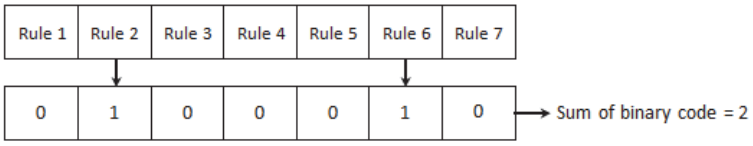

Fig. 3. Rules selection method by using GA

Table 2
Dispatching rules used in this study

No	Rule	Description
1	FIFO	FIFO stands for first in first out. This rule selects the first job which goes into the queue at the workstation buffer.
2	SPT	SPT stands for the shortest processing time. This rule selects the job that has the shortest processing time at the machine.
3	EDD	EDD stands for the earliest due date. This rule selects the job that has the earliest due date.
4	S/OPN	S/OPN stands for the minimum slack time per remaining operation. This rule selects the job with the least slack per remaining number of operations.
5	S/RPT	S/RPT stands for slack per remaining process time. This rule allocates the allowance time for operations according to the ratio of slack to the remaining processing time.
6	CT	The ratio of the current time to the sum of the remaining processing time and total processing time.
7	DDC	The ratio of the remaining time before the due date to the total completion time.

Table 3 shows randomly generated binary chromosomes with respect to the rules selected. The weightages is pre-set at 0.5 for each stage. The weightages will then be optimized by GSA after the initial evaluation. Therefore, a basic function as Eq. (3) is formed. 1:1 objectives index is used throughout the example which means that the time-related and job-related objectives are equally important.

Table 3
Initial populations

	Binary Chromosomes	Rules selected	Weightages
Pop 1	0 0 0 1 0 0 1	S/OPN +DDC	0.5 : 0.5 : 0.5
Pop 2	1 0 1 0 0 0 0	FIFO + EDD	0.5 : 0.5 : 0.5
Pop 3	0 1 0 0 0 1 0	SPT + CT	0.5 : 0.5 : 0.5
Pop 4	0 0 0 0 0 1 1	CT + DDC	0.5 : 0.5 : 0.5
Pop 5	0 1 0 0 1 0 0	SPT + S/RPT	0.5 : 0.5 : 0.5
Pop 6	1 0 0 0 0 1 0	FIFO + CT	0.5 : 0.5 : 0.5
Pop 7	0 0 0 1 1 0 0	S/OPN + S/RPT	0.5 : 0.5 : 0.5
Pop 8	0 0 1 0 0 1 0	EDD + CT	0.5 : 0.5 : 0.5
Pop 9	0 1 0 1 0 0 0	SPT + S/OPN	0.5 : 0.5 : 0.5
Pop 10	0 0 0 0 1 0 1	S/RPT + DDC	0.5 : 0.5 : 0.5

The crossover process involves the production of a pair of children chromosome from a pair of parent chromosomes. A single-point crossover operator is applied to each pair of parent chromosomes subject to a probability (crossover probability=0.8). Fig. 4 shows two examples of single-point crossover process and reproduction of offspring.

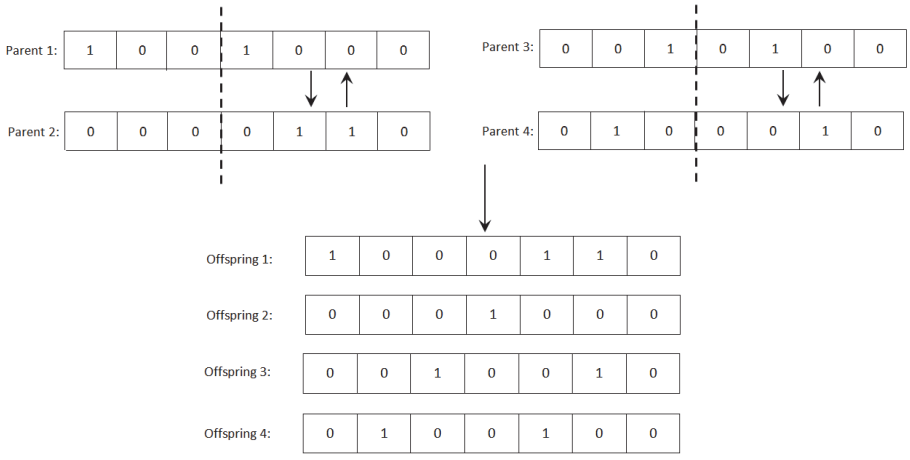

Fig. 4. Crossover

Table 4 shows the job sequencing and the results of the fitness function according to the stages and the machine number. Every stage consists of 3 machines, e.g. by using FIFO method, machine 1 is used to process job 1, 4, 7 and 10, machine 2 to process job 2, 5 and 8, and machine 3 is used to process job 3, 6 and 9 in stage 1. The sequencing for all jobs in every stage by different methods is shown in the table. We then compare the job sequencing followed by the 10 set of initial population. Next, we select the two best populations to maintain and the rest undergo crossover and mutation process.

Table 4
Initial evaluations

Method	Sequencing			Fitness
	Stage 1	Stage 2	Stage 3	
FIFO	1 4 7 10; 2 5 8; 3 6 9	1 4 7 10; 2 5 8; 3 6 9	1 4 7 10; 2 5 8; 3 6 9	1.2473
SPT	8 7 5 1; 9 2 6; 10 3 4	10 7 9 8; 2 5 4; 6 3 1	7 8 6 9; 10 5 1; 2 4 3	2.5370
EDD	10 5 1 4; 7 6 8; 2 3 9	10 5 1 4; 7 6 8; 2 3 9	10 5 1 4; 7 6 8; 2 3 9	1.3725
S/OPN	9 10 7 8; 5 6 2; 1 3 4	9 10 6 4; 5 1 2; 3 7 8	9 10 7 8; 3 6 2; 5 1 4	2.8346
S/RPT	10 2 3 9; 7 6 4; 5 1 8	10 7 3 8; 5 2 1; 9 6 4	10 7 3 8; 5 2 1; 9 6 4	2.3572
CT	8 7 5 1; 9 2 6; 10 3 4	10 7 9 8; 6 5 4; 2 3 1	7 8 6 9; 10 5 1; 2 4 3	2.5370
DDC	1 5 10 8; 9 2 4; 3 6 7	1 5 10 8; 9 2 4; 3 6 7	1 5 10 8; 9 2 4; 3 6 7	1.8360
Rules allocation	8 7 5 1; 9 2 6; 10 3 4	10 7 9 8; 2 5 4; 6 3 1	10 5 1 4; 7 6 9; 2 3 8	0.5092
		GA-GSA		
Pop 1	9 10 7 8; 5 6 2; 1 3 4	9 10 2 4; 5 7 6; 3 1 8	9 5 1 8; 3 10 7; 6 2 4	2.8346
Pop 2	**7 5 1 4; 10 6 8; 2 3 9**	**7 5 1 4; 10 6 8; 2 3 9**	**7 5 1 4; 10 6 8; 2 3 9**	**0.3742**
Pop 3	8 7 5 1; 9 2 6; 10 3 4	6 7 3 8; 2 5 4; 10 1 9	8 2 1 9; 7 5 6; 10 4 3	2.5370
Pop 4	9 7 5 1; 8 2 6; 10 3 4	6 7 3 8; 2 5 4; 10 1 9	8 2 1 9; 7 5 6; 10 4 3	2.4288
Pop 5	10 5 1 4; 7 6 8; 2 3 9	10 2 3 9; 7 6 4; 5 1 8	10 2 3 9; 7 6 8; 5 1 4	0.5389
Pop 6	2 3 5 4; 8 9 6; 7 10 1	2 5 3 8; 6 1 4; 7 10 9	7 4 10 9; 8 5 6; 2 1 3	2.0283
Pop 7	5 2 3 8; 10 6 9; 7 1 4	5 9 1 4; 10 2 6; 7 3 8	9 10 2 8; 3 6 1; 5 7 4	3.3058
Pop 8	**10 5 8 4; 7 6 9; 2 3 1**	**10 5 1 4; 7 6 9; 2 3 8**	**7 5 1 9; 10 6 4; 2 8 3**	**0.4770**
Pop 9	9 7 6 8; 10 3 1; 5 2 4	9 7 6 8; 10 3 1; 5 2 4	9 7 6 8; 10 3 1; 5 2 4	1.2674
Pop 10	10 2 3 8; 7 6 4; 5 1 9	10 2 3 4; 5 6 8; 7 1 9	9 3 6 8; 10 7 1; 5 2 4	2.3634

The mutation operation is important to provide diversity in GA search directions and to prevent the search from being converged to local optima. In this study, the binary code for each offspring chromosome is randomly swapped to ensure the constraint is fulfilled (i.e., the sum of binary code must be equal to 2). Fig. 5 shows an example of the mutation swapping method. A new population is then formed by replacing some of the chromosomes in current population with newly generated offspring.

4.4 Weightage Optimization of Gravitational Search Algorithm

Due to its simplicity, suitability for multi-dimension problem and ability to find near global optimum solution, GSA is selected in this study. In this paper, a number of N agents are randomly initialized in d

dimensions (number of stages). X_i is the location (solution/weightage) of the agent number i. Each agent evaluates the objectives and ranks the agent based on a minimization function. Fig. 6 shows the sample illustration of GSA in three dimensions with the corresponding parameters.

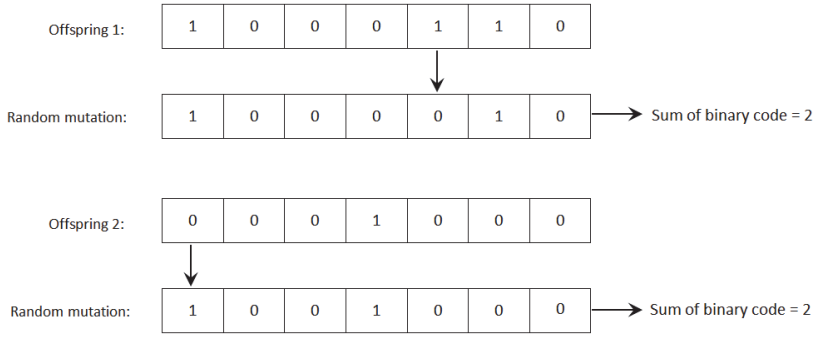

Fig. 5. Mutation

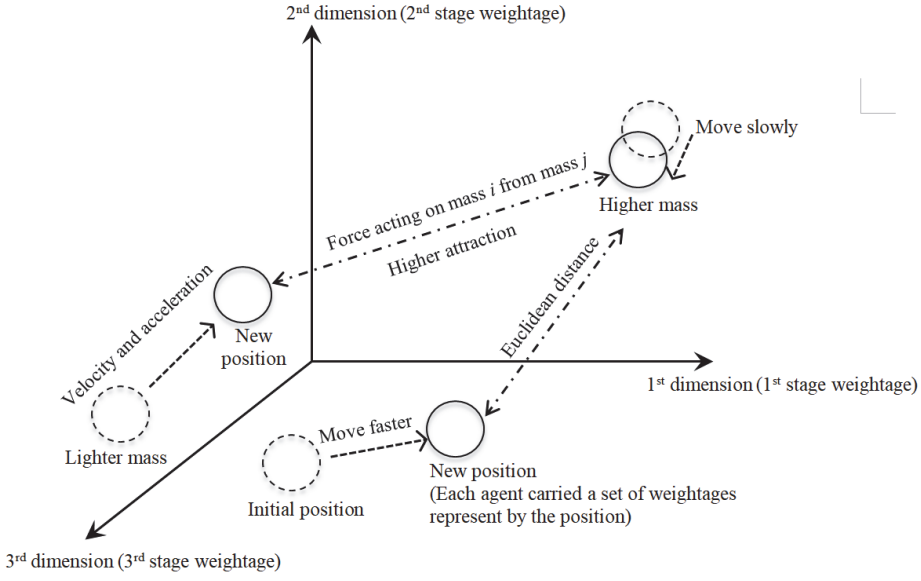

Fig. 6. Illustration of GSA search in 3 dimensional space.

The agents are located in a three-dimensional space, where each dimension represents a single stage of weightage. Hence, each agent carries a set of 3 weightages. The value of the first dimension of X location represents the contribution weightage in the objective function of the first stage; the value of the second dimension of X location represents the weightage of the second stage and so on. The agent in a higher mass represents a better solution and it moves slowly, while the agent in a lighter mass represents a worse solution, and it moves faster towards the agent in a heavier mass. The mass of the agent, gravitational constant and acceleration are calculated to determine the agent movement. The force acting on mass i from mass j at a specific time t is given by:

$$F_{ij}^{d}(t) = G(t)\frac{M_{pi}(t)\times M_{aj}(t)}{R_{ij}(t)+\varepsilon}(X_j^d(t) - X_i^d(t)), \tag{15}$$

where M_{aj} is the active gravitational mass related to agent j, M_{pj} is the passive gravitational mass related to agent j, $G(t)$ is the gravitational constant at time t, ε is a small constant and $R_{ij}(t)$ is the Euclidean distance between agents i and j.

The gravitational constant, G, is a function of the initial value (G_0) and time (t).

$$G(t) = G(G_0, t). \tag{16}$$

Parameter G_0 is set to a default value as G_0 =100. The acceleration of agent i at time t in the d^{th} direction is given by:

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)}, \tag{17}$$

where M_{ii} is the inertial mass of the i^{th} agent. The next velocity of an agent is considered as the sum of the fraction of the agent's current velocity and its acceleration. Therefore, the position and velocity of the agent can be calculated as follows:

$$V_i^d(t+1) = rand_i \times V_i^d(t) + a_i^d(t), \tag{18}$$

$$X_i^d(t+1) = X_i^d(t) + V_i^d(t+1), \tag{19}$$

where $rand_i$ is a uniform random variable within the interval [0,1] which gives a randomized characteristic to the search. The gravitational and inertial mass is changed using the following equations:

$$M_{ai} = M_{pi} = M_{ii} = M_i, \quad i = 1,2,\dots,N, \tag{20}$$

$$mm_i(t) = \frac{fit_i(t) - worst(t)}{best(t) - worst(t)}, \tag{21}$$

$$M_i(t) = \frac{mm_i(t)}{\sum_{j=1}^{N} mm_j(t)}, \tag{22}$$

where $fit_i(t)$ represents the fitness value of the agent i at time t. For a minimization problem, the following parameters *worst(t)* and *best(t)* are defined as follows:

$$best(t) = \min_{j \in (1,\dots,N)} fit_j(t), \tag{23}$$

$$worst(t) = \min_{j \in (1,\dots,N)} fit_j(t). \tag{24}$$

Table 5 shows the new populations and the optimal weightages by using GSA optimization method for each stage.

Table 5
New populations' generation

	Binary Chromosomes	Rules selected	Optimized Weightages
Pop 1	1 0 1 0 0 0 0	FIFO + EDD	0.7390 : 0.6959 : 0.7485
Pop 2	0 0 1 0 0 1 0	EDD + CT	0.6035 : 0.5315 : 0.6799
new Pop 1	0 0 0 1 0 1 0	S/OPN + CT	0.4209 : 0.3843 : 0.3438
new Pop 2	0 1 0 0 0 0 1	SPT + DDC	0.0379 : 0.7575 : 0.1060
new Pop 3	0 0 0 0 1 1 0	S/RPT + CT	0.7337 : 0.8378 : 0.4350
new Pop 4	0 1 0 0 0 0 1	SPT + DDC	0.2315 : 0.1423 : 0.1336
new Pop 5	1 0 0 0 1 0 0	FIFO + S/RPT	0.5991 : 0.9883 : 0.9830
new Pop 6	0 0 0 1 0 1 0	S/OPN + CT	0.5671 : 0.2640 : 0.3529
new Pop 7	0 1 0 0 1 0 0	SPT + S/RPT	0.2789 : 0.2821 : 0.5350
new Pop 8	0 1 1 0 0 0 0	SPT + EDD	0.3701 : 0.5699 : 0.3424

4.5 Re-evaluation and stopping criteria

The new population will be re-evaluated based on the objectives function. The process from the step 4 genetic mutation until re-evaluation is repeated until a stopping criterion (number of iteration) is met. Table 6 shows the fitness from the new populations with different rules combination, and weightages is optimized for each population.

Table 6
Re-evaluation for new populations

GA-GSA				Fitness
Pop 1	2 5 3 4; 7 6 8; 10 1 9	2 5 3 4; 7 6 8; 10 1 9	2 5 3 4; 7 6 8; 10 1 9	0.3657
Pop 2	10 5 8 4; 7 6 9; 2 3 1	10 5 1 4; 7 6 9; 2 3 8	10 5 1 9; 7 6 3; 2 8 4	0.3868
new Pop 1	9 7 6 4; 10 3 1; 5 2 8	10 7 3 8; 9 6 1; 5 2 4	10 2 3 4; 5 6 8; 7 1 9	0.5088
new Pop 2	9 10 6 4; 3 5 7; 2 1 8	6 7 3 8; 2 5 4; 10 1 9	7 5 4 9; 2 1 6; 10 8 3	1.0902
new Pop 3	10 2 3 4; 7 6 8; 5 1 9	10 2 3 9; 7 6 8; 5 1 4	10 2 8 9; 7 6 3; 5 1 4	0.3763
new Pop 4	9 8 5 4; 10 7 6; 2 3 1	6 5 3 8; 2 7 9; 10 1 4	7 8 4 9; 2 5 6; 10 1 3	0.8135
new Pop 5	2 10 3 9; 7 6 4; 5 1 8	1 4 7 10; 2 5 8; 3 6 9	1 4 7 10; 2 5 8; 3 6 9	1.4611
new Pop 6	9 3 2 8; 5 7 1; 10 6 4	10 5 3 8; 6 2 1; 7 9 4	10 2 3 4; 5 6 9; 7 1 8	0.5429
new Pop 7	10 2 3 4; 7 6 8; 5 1 9	10 2 3 9; 7 6 8; 5 1 4	10 2 8 9; 7 6 3; 5 1 4	0.3763
new Pop 8	10 5 8 4; 7 6 9; 2 3 1	10 5 1 4; 7 6 9; 2 3 8	10 5 1 9; 7 6 3; 2 8 4	0.3568

4.6 Final schedule

Table 7 shows the final results. The proposed algorithm shows the best performance in terms of the total fitness (mean ratio of excess time and the number of tardy jobs). Rules allocation algorithm can only provide a closer result to the proposed algorithm in terms of mean ratio of excess time but the total fitness is still better than other dispatching rules. This particular example only illustrates the methodology of the proposed framework. Hence, a number of randomly generated problems based on the same condition are created to validate the proposed framework in the following section.

Table 7
Final results

Method	Sequencing			ratio of excess time	No. tardy jobs	Fitness
	Stage 1	Stage 2	Stage 3			
FIFO	1 4 7 10; 2 5 8; 3 6 9	1 4 7 10; 2 5 8; 3 6 9	1 4 7 10; 2 5 8; 3 6 9	0.9473	3	1.2473
SPT	8 7 5 1; 9 2 6; 10 3 4	10 7 9 8; 2 5 4; 6 3 1	7 8 6 9; 10 5 1; 2 4 3	2.0370	5	2.5370
EDD	10 5 1 4; 7 6 8; 2 3 9	10 5 1 4; 7 6 8; 2 3 9	10 5 1 4; 7 6 8; 2 3 9	1.2725	1	1.3725
S/OPN	9 10 7 8; 5 6 2; 1 3 4	9 10 6 4; 5 1 2; 3 7 8	9 10 7 8; 3 6 2; 5 1 4	2.6346	2	2.8346
S/RPT	10 2 3 9; 7 6 4; 5 1 8	10 7 3 8; 5 2 1; 9 6 4	10 7 3 8; 5 2 1; 9 6 4	2.0572	3	2.3572
CT	8 7 5 1; 9 2 6; 10 3 4	10 7 9 8; 6 5 4; 2 3 1	7 8 6 9; 10 5 1; 2 4 3	2.3370	2	2.5370
DDC	1 5 10 8; 9 2 4; 3 6 7	1 5 10 8; 9 2 4; 3 6 7	1 5 10 8; 9 2 4; 3 6 7	1.3360	5	1.8360
Rules allocation	8 7 5 1; 9 2 6; 10 3 4	10 7 9 8; 2 5 4; 6 3 1	10 5 1 4; 7 6 9; 2 3 8	0.4092	1	0.5092
GA-GSA	10 5 8 4; 7 6 9; 2 3 1	10 5 1 4; 7 6 9; 2 3 8	10 5 1 9; 7 6 3; 2 8 4	0.2568	1	0.3568

5. Results

In this section, the proposed GA-GSA-DR is validated by using randomly generated test problem (design of experiment) to compare with the conventional dispatching rules and other intelligent algorithms.

5.1 Random generated problems

We have conducted a set of experiments to evaluate the performance of the proposed algorithm based on the problem constraints listed in Table 8 in order to validate different conditions, where the flow shop comprises 3 stages and at each stage, 3 machines operate in parallel. Assumptions were made which

included, no backup machine was available and no transportation time exists between stages. There were 10 ready jobs to be processed as well as 30 random generated cases with random processing time, and due date was carried out for each problem. The design of experiments is given in Table 9. In total, we have tested 15 problems by combining different range of parameters**.** The results obtained by the proposed algorithm are compared with 7 general dispatching rules and GA rules allocation method.

Table 8
Problem constraint with machines and job setting

Problem constraint	**parameters**
Number of stages	3
Number of parallel machines	3
Number of jobs	10
Number of random generated cases	30

Table 9
Factor levels for the experiments

No	Factors	Levels
1	Processing time, P	(5,35) small range problem
		(5,65) medium range problem
		(5,95) large range problem
2	Due date, D	(+5,+80) tight range due date
		(+40, +120) medium range due date
		(+80, +160) large range due date
3	Objectives index, λ	>1 (ratio: time-related > job related)
		1:1 (ratio: time-related = job related)
		<1 (ratio: time-related < job related)

The average fitness results comparison are presented in Table 10.

Table 10
Average of fitness results from different algorithm

Problem	Processing time, P	Due date, D	λ	Average of fitness							
				FIFO	CT	DDC	SPT	S/OPN	S/RPT	EDD	Proposed algorithm
1	(5, 35)	(+5,+80)	1:1	0.8839	0.3812	0.9350	0.4261	1.1087	0.8431	0.5552	**0.2082**
2	(5, 35)	(+40,+120)	1:1	0.6608	0.3209	0.8135	0.2706	0.7211	0.6484	0.4729	**0.1666**
3	(5, 35)	(+80,+160)	1:1	0.6184	0.2370	0.8160	0.2223	0.7759	0.6084	0.4313	**0.1841**
4	(5,65)	(+5,+80)	1:1	1.1561	0.5054	1.4063	0.5373	1.7501	1.2846	0.6880	**0.3282**
5	(5,65)	(+40,+120)	1:1	0.9234	0.4160	1.2047	0.3835	1.3946	0.9101	0.4936	**0.1933**
6	(5,65)	(+80,+160)	1:1	0.7722	0.3229	1.1516	0.2671	1.1257	0.7251	0.3897	**0.1647**
7	(5,90)	(+5,+80)	1:1	1.3458	0.5788	1.3489	0.5757	1.9819	1.3230	0.7963	**0.3973**
8	(5,90)	(+40,+120)	1:1	1.0687	0.4564	1.3364	0.4706	1.7621	1.0908	0.6367	**0.2222**
9	(5,90)	(+80,+160)	1:1	0.9684	0.3452	1.2875	0.3755	1.5507	0.8671	0.4425	**0.1976**
10	(5, 35)	(+5,+80)	>1	0.9506	0.4189	1.0828	0.4556	1.2689	0.8996	0.6505	**0.2871**
11	(5, 35)	(+40,+120)	>1	0.8677	0.3534	1.0245	0.3766	1.0504	0.8429	0.5758	**0.2617**
12	(5, 35)	(+80,+160)	>1	0.8253	0.2953	1.1352	0.3046	1.0000	0.8756	0.6009	**0.2510**
13	(5, 35)	(+5,+80)	<1	0.8945	0.3744	0.8201	0.4002	0.9602	0.7207	0.5100	**0.1652**
14	(5, 35)	(+40,+120)	<1	0.5245	0.2188	0.6088	0.2228	0.5384	0.4241	0.3098	**0.1424**
15	(5, 35)	(+80,+160)	<1	0.4328	0.1675	0.5870	0.1600	0.5399	0.4110	0.3126	**0.1393**

We can observe that the proposed algorithm provides the best solution. The proposed algorithm can provide the minimum fitness value (minimization objectives function) with a variant of problem constraint such as tighten the due date, varies the objectives index and increase the range of the processing time for each job. It showed the flexibility of the algorithm in the bi-objective (time and job-related objectives) definition. The average mean ratio of excess time comparison is presented in Table 11. The average of tardiness is presented in Table 12. In Table 11 and Table 12, the mean ratio of excess time and job tardiness are evaluated separately. The results obtained from the proposed algorithm are better than all eight methods in terms of the time and job-related objectives. The CT and SPT rules can provide

good results in the mean ratio of excess time (which is close to proposed algorithm) but they are inferior in job tardiness. On the other hand, EDD can give good result in job tardiness but its mean ratio of excess time is longer. It is evident that optimum solutions can be attained by the proposed algorithm for both mean excess time and tardiness with a variety of objectives index.

5.2 Comparison with others intelligent algorithms

The same design of experiment is generated by only changing the number of jobs, and the number of stages. In this set of experiment, different intelligent algorithms are used to compare with the proposed GA-GSA-DR. Four types of rules allocation methods are used to compare with the proposed framework which are included GA-SM-DR (Korytkowski et al., 2013a), DRGA-II (Vázquez-Rodríguez & Petrovic, 2009), Fuzzy-DR (Lu & Liu, 2010) and ACO-DR (Korytkowski et al., 2013b). Besides, Genetic Algorithm (GA), Ant Colony Optimization (ACO), Harmony Search (HS), Particle swarm optimization (PSO), and Cuckoo optimization are used to compare with the proposed GSA optimization method in weighted dispatching rules method.

Table 11
Average of mean ratio of excess time results from different algorithm

Problem	Processing time, P	Due date, D	λ	Average of mean ratio of excess time							
				FIFO	CT	DDC	SPT	S/OPN	S/RPT	EDD	Proposed algorithm
1	(5, 35)	(+5,+80)	1:1	0.5772	0.2146	0.7317	0.2261	0.7454	0.6198	0.4419	**0.1982**
2	(5, 35)	(+40,+120)	1:1	0.5708	0.2509	0.7635	0.2406	0.6911	0.6484	0.4729	**0.1666**
3	(5, 35)	(+80,+160)	1:1	0.6018	0.2370	0.8060	0.2189	0.7759	0.6084	0.4313	**0.1841**
4	(5,65)	(+5,+80)	1:1	0.6795	0.2021	0.8796	0.2273	1.0167	0.7146	0.3480	**0.1849**
5	(5,65)	(+40,+120)	1:1	0.6667	0.2160	0.8813	0.2069	0.9879	0.6834	0.3903	**0.1700**
6	(5,65)	(+80,+160)	1:1	0.6655	0.2329	0.9616	0.2138	0.9923	0.6584	0.3764	**0.1614**
7	(5,90)	(+5,+80)	1:1	0.7791	0.2254	0.7822	0.2157	1.1452	0.7196	0.3463	**0.1906**
8	(5,90)	(+40,+120)	1:1	0.6354	0.2064	0.8564	0.2140	1.0754	0.6808	0.3567	**0.1522**
9	(5,90)	(+80,+160)	1:1	0.6784	0.1985	0.9342	0.2122	1.0807	0.6571	0.3425	**0.1709**
10	(5, 35)	(+5,+80)	>1	0.5697	0.2289	0.7144	0.2463	0.7840	0.5663	0.4483	**0.1993**
11	(5, 35)	(+40,+120)	>1	0.6208	0.2485	0.7665	0.2502	0.7954	0.6412	0.4429	**0.1995**
12	(5, 35)	(+80,+160)	>1	0.6277	0.2254	0.8643	0.2343	0.7692	0.6735	0.4623	**0.1931**
13	(5, 35)	(+5,+80)	<1	0.6526	0.2378	0.6764	0.2312	0.7775	0.6148	0.4500	**0.2236**
14	(5, 35)	(+40,+120)	<1	0.6316	0.2383	0.7830	0.2317	0.7321	0.5996	0.4363	**0.2034**
15	(5, 35)	(+80,+160)	<1	0.6183	0.2393	0.8386	0.2286	0.7712	0.5871	0.4466	**0.1990**

Table 12
Average of tardiness results from different algorithm

Problem	Processing time, P	Due date, D	λ	Average of tardiness							
				FIFO	CT	DDC	SPT	S/OPN	S/RPT	EDD	Proposed algorithm
1	(5, 35)	(+5,+80)	1:1	0.3067	0.1667	0.2033	0.2000	0.3633	0.2233	0.1133	**0.0100**
2	(5, 35)	(+40,+120)	1:1	0.0900	0.0700	0.0500	0.0300	0.0300	0.0000	0.0000	**0.0000**
3	(5, 35)	(+80,+160)	1:1	0.0167	0.0000	0.0100	0.0033	0.0000	0.0000	0.0000	**0.0000**
4	(5,65)	(+5,+80)	1:1	0.4767	0.3033	0.5267	0.3100	0.7333	0.5700	0.3400	**0.1433**
5	(5,65)	(+40,+120)	1:1	0.2567	0.2000	0.3233	0.1767	0.4067	0.2267	0.1033	**0.0233**
6	(5,65)	(+80,+160)	1:1	0.1067	0.0900	0.1900	0.0533	0.1333	0.0667	0.0133	**0.0033**
7	(5,90)	(+5,+80)	1:1	0.5667	0.3533	0.5667	0.3600	0.8367	0.6033	0.4500	**0.2067**
8	(5,90)	(+40,+120)	1:1	0.4333	0.2500	0.4800	0.2567	0.6867	0.4100	0.2800	**0.0700**
9	(5,90)	(+80,+160)	1:1	0.2900	0.1467	0.3533	0.1633	0.4700	0.2100	0.1000	**0.0267**
10	(5, 35)	(+5,+80)	>1	0.3000	0.1733	0.2200	0.1933	0.3567	0.2333	0.0967	**0.0400**
11	(5, 35)	(+40,+120)	>1	0.0867	0.0433	0.0400	0.0733	0.0233	0.0133	0.0000	**0.0033**
12	(5, 35)	(+80,+160)	>1	0.0133	0.0033	0.0167	0.0000	0.0000	0.0000	0.0000	**0.0000**
13	(5, 35)	(+5,+80)	<1	0.3367	0.1600	0.2667	0.1833	0.3200	0.2233	0.1500	**0.0067**
14	(5, 35)	(+40,+120)	<1	0.0633	0.0400	0.0467	0.0467	0.0200	0.0033	0.0033	**0.0000**
15	(5, 35)	(+80,+160)	<1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	**0.0000**

The effectiveness of the algorithms was testified by solving 15 different problems. Table 13 shows the comparative results of the intelligent rules allocation methods, with respect to the average fitness, for different size of jobs and stages. As can be seen from Tables 13, it should be said that the proposed method has performed better than other approaches especially when the number of jobs increases. The proposed framework outperformed due to the flexibility of rules selection and the combining the optimized weightages. It might also due to the rules selected in this paper is limited. On the others hands, in terms of the optimization algorithm used in the framework, GSA provides better suitability by comparing with other optimization approaches. The non-randomize iteration helped GSA improve the reasonable convergence ability in search space. In addition, the effect of the weight contribution of both objectives is evaluated by the changes the skewness of the objectives index, λ. From Table 13, we have found that when the number of jobs is getting bigger, the influence of the mean ratio time is getting larger compared with tardiness. When the number of jobs is getting bigger, the ascendency of the proposed method is getting obvious especially when GSA optimization is used.

Table 13
Comparison with others intelligent algorithms

No. jobs	Objectives index, λ	Intelligence Algorithms (Average Fitness)								
		Intelligent Rules Allocation methods				Proposed Weighted Dispatching Rules method				
		GA-SM-DR	DRGA-II	Fuzzy-DR	ACO-DR	ACO	HS	PSO	Cuckoo	GSA
10	1:1	2.25748	2.95798	2.96645	0.95887	1.03245	1.15008	1.00304	1.08081	**0.92989**
10	>1	2.88373	3.16691	2.82885	0.88309	0.87618	0.92592	0.86204	1.02185	**0.86021**
10	<1	2.27462	2.48585	2.13731	1.01373	1.06275	1.10451	1.05565	1.20683	**0.96471**
15	1:1	4.17078	3.75328	3.79817	1.32760	**1.28859**	1.39595	1.39304	1.42543	1.37469
15	>1	4.63921	4.33664	4.54607	1.28381	1.30578	1.38945	1.24205	1.37641	**1.21886**
15	<1	3.46808	3.75328	3.24709	1.40507	1.41942	1.51924	1.38788	1.46603	**1.37252**
20	1:1	5.33261	5.73953	5.66514	1.71039	1.69884	1.77304	**1.62687**	1.96380	1.64511
20	>1	6.02927	6.94752	6.86860	1.68712	1.63200	1.80920	1.63535	1.84163	**1.61456**
20	<1	4.70898	4.44254	4.63385	1.67943	1.66583	1.69217	**1.64129**	1.88312	1.66065
25	1:1	7.76314	6.80807	7.06828	1.94937	1.96671	1.94229	1.96626	2.42644	**1.92802**
25	>1	9.62020	8.29962	7.56973	2.08695	2.06559	2.12042	2.02144	2.33328	**2.00555**
25	<1	5.61133	5.47384	5.61590	1.86899	1.86135	1.99463	**1.85846**	2.11725	1.87852
30	1:1	7.87419	7.98858	9.50783	2.25323	2.24064	2.58023	2.21160	2.66897	**2.20215**
30	>1	11.60066	10.47108	10.32352	2.38538	2.30336	2.63825	2.33758	2.98728	**2.30092**
30	<1	6.73558	7.54030	6.68989	2.06587	2.10714	2.10008	2.06579	2.76103	**2.06123**

6. Conclusion

This study has attempted to determine the optimal use of dispatching rules in a flow shop system by a proposed enhancement method where the fitness function includes the time and job-related objectives. The multiple objective method considered in this paper has tried to find a schedule by minimizing the mean ratio of excess time and tardiness simultaneously by assigning the jobs (sequencing) to the available machines. Traditional dispatching rules might not be applicable or perform well in multi-objective problems due to the single criteria for each dispatching rules. Hence, in this paper, we have proposed a hybrid GA-GSA in weighted dispatching rules to respond to the multi-objective scenarios and enhance conventional dispatching rules method. GA has been used to select two appropriate dispatching rules to combine as a weighted multi-attribute function, while the GSA has been used to optimize the contribution weightage of each rule in each stage of the flow shop. The test results which include the case study and randomly generated cases has concluded that among the single rules and rules allocation method, the proposed hybrid algorithm has provided the best solution for both time and job-related objectives. It performed better than optimized rules allocation methods.

Acknowledgment

The authors would like to thank the reviewers for their constructive comments and the Ministry of Higher Education of Malaysia for the support of this work through the Foundation Research Grant Scheme (FRGS/1/2015/TK03/MMU/02/3).

References

Abraham, A., Grosan, C., & Pedrycz, W. (2008). Engineering Evolutionary Intelligent Systems. *Springer Berlin Heidelberg*.

Chen, T., Rajendran, C,. & Wu, C.W. (2013). Advanced dispatching rules for large-scale manufacturing systems. *The International Journal of Advanced Manufacturing Technology, 67*(1-4), 1-3.

Choi, S. H., & Wang, K. (2012). Flexible flow shop scheduling with stochastic processing times: *A decomposition-based approach. Computers & Industrial Engineering, 63*(2), 362-373. doi:10.1016/j.cie.2012.04.001

El Bouri, A. & Amin, G.R. (2015). A combined OWA–DEA method for dispatching rule selection. *Computers & Industrial Engineering, 88*, 470-478.

Eldos, T. & Qasim, R.A. (2013). On the performance of gravitational search algorithm. *International Journal of Advanced Computer Science and Applications, 4*(8), 74-78.

Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Publishing Company.

Jayamohan, M. S., & Rajendran, C. (2000). A comparative analysis of two different approaches to scheduling in flexible flow shops. *Production Planning & Control, 11*(6), 572-580. doi:10.1080/095372800414133

Joo, B. J., Choi, Y. C., & Xirouchakis, P. (2013). Dispatching rule-based algorithms for a dynamic flexible flow shop scheduling problem with time-dependent process defect rate and quality Feedback. *Procedia CIRP, 7,* 163-168.

Jungwattanakit, J., Reodecha, M., Chaovalitwongse, P., & Werner, F. (2008). Algorithm for flexible flow shop problems with unrelated parallel machines, setup times and dual criteria. *International Journal of Advance Manufacturing Technology, 37*(3-4), 354-370. doi:10.1007/s00170-007-0977-0

Khalouli, S., Ghedjati, F., & Hamzaoui, A. (2010). A meta-heuristic approach to solve a JIT scheduling problem in hybrid flow shop. *Engineering Applications of Artificial Intelligence*, *23*, 765-771.

Kim, Y.-D., Joo, B.-J., & Shin, J.-H. (2007). Heuristics for a two-stage hybrid flowshop scheduling problem with ready times and a product-mix ratio constraint. *Journal of Heuristics, 15*(1), 19-42. doi:10.1007/s10732-007-9061-z

Kumar, Y. & Sahoo, G. (2014). A review on gravitational search algorithm and its applications to data clustering & classification. *International Journal of Intelligent Systems and Applications, 6*(6), 79-93.

Korytkowski, P., Wiśniewski, T., & Rymaszewski, S. (2013a). An evolutionary simulation-based optimization approach for dispatching scheduling. *Simulation Modelling Practice and Theory, 35*, 69-85. doi:10.1016/j.simpat.2013.03.006

Korytkowski, P., Rymaszewski, S., & Wiśniewski, T. (2013b). Ant colony optimization for job shop scheduling using multi-attribute dispatching rules. *The International Journal of Advanced Manufacturing Technology, 67*(1-4), 231-241.

Liptak, B.G. (2005). *Instrument Engineers' Handbook, Fourth Edition, Volume Two: Process Control and Optimization*. CRC Press.

Li, D., Meng, X., Liang, Q., & Zhao, J. (2014). A heuristic-search genetic algorithm for multi-stage hybrid flow shop scheduling with single processing machines and batch processing machines. *Journal of Intelligent Manufacturing, 26*(5), 873-890.

Li, D. (2014). A multi-objective TLBO algorithm for balancing two-sided assembly line with multiple constraints. *Journal of Intelligent Manufacturing, 27*(4), 725-739.

Lu, M.-S., & Liu, Y.-J. (2010). Dynamic dispatching for a flexible manufacturing system based on fuzzy logic. *International Journal of Advanced Manufacturing Technology, 54(9-12),* 1057-1065.

Morita, H., & Shio, N. (2005). Hybrid branch and bound method with genetic algorithm for flexible flowshop scheduling problem. *JSME International Journal Series C-Mechanical Systems Machine Elements and Manufacturing, 48*(1)*,* 46-52.

Nguyen, S., Zhang, M., Johnston, M., & Tan, K. C. (2013). Learning iterative dispatching rules for job shop scheduling with genetic programming. *International Journal of Advanced Manufacturing Technology, 67*(1-4), 85-100.

Pérez, M.A.F. and Raupp, F.M.P. (2014). A Newton-based heuristic algorithm for multi-objective flexible job-shop scheduling problem. *Journal of Intelligent Manufacturing, 27*(2), 409-416.

Pinedo, M. (2008). *Scheduling: Theory, algorithm, and systems* (3rd ed.). New York: Springer.

Rashedi, E., Nezamabadi-pour, H., & Saryazdi, S. (2009). GSA: A Gravitational Search Algorithm. *Information Sciences, 179*(13), 2232-2248.

Ribas, I., Leisten, R., & Framiñan, J. M. (2010). Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective. *Computers & Operations Research, 37*(8), 1439-1454.

Rodriguez, J. A. V., & Salhi, A. (2005, September). Performance of single stage representation genetic algorithms in scheduling flexible flow shops. In *2005 IEEE Congress on Evolutionary Computation* (Vol. 2, pp. 1364-1371). IEEE.

Ruiz, R., & Vázquenz-Rodríguez, J. A. (2010). The hybrid flow shop scheduling problem. *European Journal of Operational Research, 205*(1), 1-18.

Sabri, N.M., Puteh, M., & Mahmood, M.R. (2013). A review of gravitational search algorithm. *International Journal of Advance in Soft Computing, 5(3).*

Simon, D. (2013). *Evolutionary Optimization Algorithms*. Wiley.

Singh, A. & Deep, K. (2015). Real coded genetic algorithm operators embedded in gravitational search algorithm for continuous optimization. *International Journal of Intelligent Systems and Applications, 7*(12), 1-12.

Spears, W.M. (2000). *Evolutionary Algorithms: The Role of Mutation and Recombination.* Springer Berlin Heidelberg.

Tian, Y., Li, D., Zhou, P., Guo, R., & Liu, Z. (2018). An ACO-based hyperheuristic with dynamic decision blocks for intercell scheduling. *Journal of Intelligent Manufacturing*, 29(8), 1905–1921.

Vázquez-Rodríguez, J.A. & Petrovic, S. (2009). A new dispatching rule based genetic algorithm for the multi-objective job shop problem. *Journal of Heuristics, 16*(6), 771-793.

Wang, K., & Choi, S. H. (2012). A decomposition-based approach to flexible flow shop scheduling under machine breakdown. *International Journal of Production Research, 50*(1)*,* 215-234. doi:10.1080/00207543.2011.571456

Wang, K., & Choi, S. H. (2014). A holonic approach to flexible flow shop scheduling under stochastic processing times. *Computers & Operations Research, 43,* 157-168.

Xu, Y. (2013). An effective immune algorithm based on novel dispatching rules for the flexible flow-shop scheduling problem with multiprocessor tasks. *The International Journal of Advanced Manufacturing Technology, 67*(1-4), 121-135.