MODELING AND SOLVING THE PERMUTATION FLOW SHOP SCHEDULING PROBLEM WITH LIMITED BUFFERS AND THE OBJECTIVES OF THE MINIMIZATION OF TOTAL ENERGY CONSUMPTION AND MAKESPAN

[a]AMIN RASI RAHIMI,[b] MOHSEN ZIAEE

[a]*Department of industrial engineering, University of Bojnord, Emam Street, Bojnord, Iran,*
[b]*Department of industrial engineering, University of Bojnord, Emam Street, Bojnord, Iran.*

email: [a]amin_ras_boj@yahoo.com, [b]m.ziaee.1789@hotmail.com

Abstract. A lot of energy, by machines that are unemployed, it is a waste. In this paper during the workshop on the issue of permutations with limited buffers and energy considerations will be discussed. The goal is minimizing total system cost that is included energy costs and overhead machinery. For a number of periods of unemployment is minimized to reduce energy waste with limited capacity buffers between two consecutive machine we have taken._Because high computational complexity point of review, of the Genetic Algorithm and simulated annealing is used to solve it. Three sets of problems with small, medium and large for the performance of the algorithm is established that the results obtained the proper functioning of both the algorithm and the dominance compared with simulated annealing genetic algorithm for all aspects of the show.

Keywords: Permutation flow shop scheduling, limited buffers, Total energy consumption, Makespan

1. Introduction

Our planet has recently encountered a serious issue on fuel reduction, carbon dioxide emission and global Warming, because of the vast usage of energy resources. Industrial sector accounts for 31% of energy consumption and 36% of carbon dioxide emission (Bruzzone et al. 2012). 34% of energy was consumed by manufacturing departments in USA in 2006 (Fang et al. 2011). Saving energy is an important issue and many researchers have focused on this subject to find strategies to reduce its consumption. Some international standards (EN 16001, ISO 50001) have been designed to manage the energy consumption of industrial plants and other sectors (Bruzzone et al. 2012). In this article we present a method that aims to select the optimum sequence work and unemployment as well as the interval of the machine must be idle or turned off and turned on again to continue operation (launch), So that the total cost of the overhead costs of equipment and systems that minimize their energy costs.

2. Literature review

A great amount of energy is being used in industry and it's very important to utilize proper policies that can reduce the energy consumption in this sector. Because of the importance of the energy consumption, many researchers have been attracted to this area and some investigations have been done recently. There are various strategies to prevent the waste of energy in industry that have been studied so far. One of the strategies is to turn the machines on and off while they are not processing any job. Mouzon and Yildirim (2008), introduced a multi-objective mathematical model for the mentioned strategy on a single machine, they tried to optimize the total tardiness and total energy consumption of the system by deciding the start time and the length of turn off/turn on periods while the machine is idle. They considered release date of jobs in the model and proved their problem is in the set of NP-hard problems so they utilized a randomized multi-objective adaptive search meta-heuristic to solve the problem. Liu et al. (2013), studied a similar problem for job shop environment. They assumed that each machine has three levels of power consumption: idle power, switched into run-time power and the process power. They proposed a bi-objective mathematical model. The objective included total electricity consumption and total weighted tardiness. A non-dominant Sorting Genetic Algorithm was employed as the solving method to obtain a proper pareto-front for the problem. Dai et al. (2013), investigated the flexible flowshop problem by considering the energy consumption. A multi-objective mathematical model and an improved genetic-simulated annealing algorithm were presented in their paper to obtain a trade-off between makespan and total energy consumption. In many industrialized countries the time of day has a significant influence on the price of the power consumption. Time of the day is usually divided in to three parts: peak load, mid-peak load, off-peak load. The peak time will lead to the most power cost so the goal is to schedule the jobs in off-peak time as the first priority and mid- peak time as the second and scape scheduling in peak time. Moon et al. (2013), studied the unrelated parallel machine scheduling problem in a 24 hour of a day divided to peak, mid-peak and off-peak to minimize the weighted sum of makespan and time dependent electricity cost. They suggested a hybrid genetic algorithm to solve the problem. Hybrid flowshop scheduling problem considering time dependent electricity cost is studied by Luo et al. (2013), they presented a new ant colony optimization meta-heuristic to solve the problem. Problem of scheduling a single machine considering variable energy prices during a day was investigated by Shrouf et al. (2014). They proposed a mathematical model and solved the problem by genetic algorithm. In industry one of the most effective factors that influence the energy cost of workshops, is their peak power consumption. Peak power is the maximum amount of the power is being used during a specific time period, if a workshop can reduce its peak power it can save a great amount of energy cost, the best solution to do so is to omit the overlap of the processing jobs at any time of the planning horizon but in other hand in will led to the maximization of time-based objectives such as makespan, so the goal is to find a trade-off between these objectives. Bruzzone et al. (2012), studied the flexible flowshop problem by limiting the peak power of the system and proposed a Mixed Integer Programming (MIP) model to minimize the sum of makespan and total tardiness. They provided a randomized neighborhood Search to solve the problem. Fang et al. (2013), studied the flowshop scheduling problem with peak power constraint and investigated both mathematical programming and combinatorial approaches to this problem. There is another way to decline power consumption of a workshop, in this strategy the manufacturer have to decide which job have to be done with which speed on each machine to reduce energy factors such as peak power or total power consumption. The flowshop scheduling problem with the objective function of makespan, peak total power and carbon footprint has been studied by Fang et al. (2011), in this problem they assumed each machine has multiple speeds, and the suitable speed has to be chosen for each one of jobs in the sequence. A multi-objective mathematical programming model has been presented for the problem. Strusevich, (1995) studied two-machine flowshop problem with no-wait while the machine speeds are controllable. He described an algorithm to solve the problem. Fang and Lin (2013), investigated the parallel machine scheduling problem to minimize total weighted tardiness and power cost. They presented a mathematical model for the problem. The purpose of the problem was to assign the products to parallel machines meanwhile the most suitable processing speed has to be selected for each job on each machine it is assigned to. They proposed two heuristics to solve the problem and used particle swarm optimization (PSO) to compare the solutions. Sharma *et al.* (2015), examined the so-called econological speed-scaling scheduling problem for multi-part multi-machine setup operating under TOU tariffs. The aim is to minimize the electricity cost and environmental impact for a target production quota by using a multi-criterion meta-heuristic optimization. Mikhaylidi *et al.* (2015), studied a manufacturing operations scheduling problem under TOU tariffs with a rechargeable battery. They proposed a dynamic programming algorithm to decide the time to process each operation such that the total electricity consumption and operations postponement penalty costs are minimized.

3. Problem description

The permutation flow shop scheduling with limited buffers considered in this paper can be described as follows. There are n jobs $J = \{1, 2, \dots, n\}$ and m machines $M = \{1, 2, \dots, m\}$. Each of the n jobs is to be sequentially processed on through machine $1 \sim m$. At any time, no job can be processed on more than one machine, while no machine can process more than one job simultaneously. The processing time of job j on machine i is given as t_{ij}. Between each successive pairs of machines i and $i-1$ there exist a buffer with the size b (i.e., at most b jobs can be held simultaneously), and jobs obey the FIFO (first in first out) rule in each buffer. Besides, the permutation of all jobs to be processed on each machine is the same. Drake et al. (2006) realised that in manufacturing environment, large quantities of energy are consumed by non-bottleneck machines as they lie idle, and that whenever a machine or is turn on or turn off, there are significant amounts of start-up or shut-down energy consumption. As a result, we have to decide at the time that the machine remains idle, turn it off and restart it to continue operations or put the machine in stand by mode. For this purpose, Mouzon and Yildirim (2008) proposed formula (1) in single machine environment.

$$tb = max\left\{\frac{Energy_{setup}}{Power_{idle}}, T_{setup}\right\} \quad (1)$$

In the above formula tb is defined as the least amount of duration required for a turn off/turn on operation (i.e. time required for a setup) and the amount of time for which a turn off/turn on operation is logical instead of running the machine at $stand\ by$. $Energy_{setup}$, $Power_{idle}$ and T_{setup} are the energy needed to setting up the machine, the power of idle working and the time needed to set up the machine respectively. In this paper, because we work on flow shop, we calculate tb for each machine, so the formula (1) is changed to (2):

$$tb_i = max\left\{\frac{es_i}{pi_i}, ts_i\right\} \quad (2)$$

In the above formula es_i, pi_i and ts_i are set up energy, idle power and set up time length for the machine i respectively. In the above formula tb_i is defined as follows:

If idle time of machine i is larger than tb_i, then $turn\ off/\ turn\ on$ option is selected and if it is less than or equal to tb_i, then $stand\ by$ option is selected.

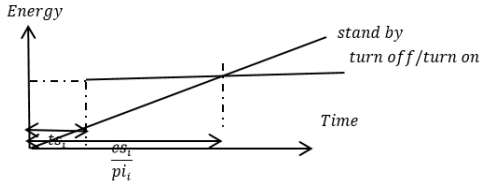

Fig.1 $\boldsymbol{tb_i = max\left\{\frac{es_i}{pi_i}, ts_i\right\} = \frac{es_i}{pi_i}}$

In the above graph the $stand\ by$ line is the time that the job stays on machine and the machine is $stand\ by$ mode, i.e. the state that the machine does not turns off so that there is no need to energy while starting. So in this state it uses fixed energy so that it has a linear diagram with time. The $turn\ off/turn\ on$ line is about the time that the job stays on machine and we turn it off and we start it again for the operation. So fixed energy is needed for setting up. It is apparent in figure 1 that when the time length of the idle is less than tb_i, we should machine i $stand\ by$ and when the time length of the idle is more than tb_i the machine i should be turned off and start it again for the operation.

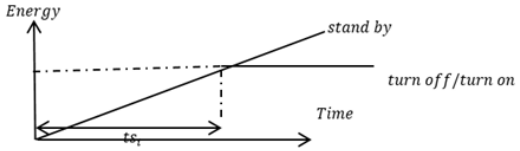

Fig. 2 $\boldsymbol{tb_i = max\left\{\frac{es_i}{pi_i}, ts_i\right\} = ts_i}$

The objective function of the problem that we are considering is composed of two parts, the first part is about overhead costs of machines and the second part is about consumed electric costs. Obtaining departure time of the last processed job on the last machine and multiplying by the overhead cost of all machines (c), the overhead cost of the all machinery is calculated. The second part of objective function is about electric consumption of machinery. It is assumed that electric consumption of machines is consist of job processing energy (the energy that is consumed for operation on the job) and stand by energy (the energy the machine uses while stand by work) and set up energy (the energy that is used for setting the machine up). For obtaining the set up energy of machine i for a special operation we use formula (3). In this formula es_i and Tof_{io} are setup energy of a machine i and a binary variable which if the turn off/turn on operation on machine i between positions of o and o+1 was done that variable would be 1 and otherwise it would be zero respectively.

$$setup\ energy\ \text{of}\ machine\ i\ = Tof_{io} es_i \quad (3)$$

For obtaining stand by energy of machine i for a special operation we use formula (4) which in it is refereed in Halliday et al. (2010). In this formula pi_i and SCD_{io} show the idle power of machine i and the the time length of idle between positions of o and o+1 for machine i which in this time length machine i is ready to start (stand by).

$$\begin{aligned} stand\ by\ energy\ \text{of}\ machine\ i \\ &= power\ \times\ Time \\ &= pi_i SCD_{io} \end{aligned} \quad (4)$$

By adding the values of equation (3) and (4) and multiplying that value with total electricity costs (ec), we can get the energy cost of the system and then we will get the total system cost by adding the energy costs with overhead costs. Note that the total processing energy ($\sum_{i=1}^{m}\sum_{j=1}^{n} P_i t_{ij}$) is not included in the optimization problem, since it is a fixed constant regardless of the sequence. In this formula P_i and t_{ij} are processing power of machine i and processing time of job j on machine i respectively.

Assumptions contained in the question under consideration include:

- Preemption in jobs is not allowed.
- An operation cannot be performed by more than one machine at the same time.
- Each machine can process at most one job at a time.
- Jobs are independent of each other.
- All parameters are certain.
- Buffer space is limited.
- Two types of energy that are intended for the machine, including total setup energy and total stand by energy.
- All jobs have equal priorities.
- Machines are so flexible, so that we can turn them off and on any time we need.

4. Solution method

The studied problem is strongly NP-hard. Therefore, two meta-heuristic algorithms: a genetic algorithm (GA) and a simulated annealing (SA) algorithm are proposed to solve the problem.

4.1 Genetic algorithm

The genetic algorithm was introduced in 1970 by Holland. It's an efficient stochastic search method that has been used by many researchers in various fields such as scheduling problems. This algorithm is a kind of evolutionary algorithms and is inspired by natural behavior of human body. In the first step it starts by an initial population (a set of solutions), which are coded as a sequence of numbers, called chromosome. New chromosomes (offsprings) are created by applying crossover and mutation operations to the parents. Crossover operator creates new offsprings by exchanging some parts of the selected parents, and the mutation operator tries to make slight changes in the parents by swapping its elements or other known methods. The algorithm uses a fitness function to evaluate the quality of existing solutions to select the elitist ones and create the next generation by probabilistic methods. The outline of the proposed genetic algorithm is given in Fig. 3.

Genetic algorithm procedure

Initialization

Parameter setting (p_c , p_m, Stopping Criteria)

Generate initial population by some heuristic methods or using random solutiRepeat

Select elite parents for crossover and generate new offsprings

Select randomly parents for mutation and generate new offsprings

Evaluate the fitness of parents and generated offsprings

Select the solution with the best fitness as the best solution of that generation

***If** the fitness of the best solution in current generation is the same as the fitness of the previous generations*

***Then** Give a shock to the system by keeping the best solution and regenerating new populations*

End if

***If** the stop criterion is satisfied*

***Then** stop*

***Else** Select the bests of parents and offsprings to enter to the next generation*

Fig. 3 Encoding, Initial solution and fitness function of GA

4.1.1 Encoding

Showing the chromosome response, the order of jobs is obtained. So the refereed chromosome is defined so that the first number of chromosome from the left shows the number of first job of order and the second number of chromosome from the left shows the number of the second job of the order and so on. So the length of the chromosome is n. For clarification an example is given in Table 1. In that example it is assumed that four jobs are there for processing. The numbers 2,3,4 and 1 shows the first, the second, the third and the last job of the order.

Table 1: Chromosome representation

2	3	4	1

4.1.2 Initial population

Two initial populations are used in the algorithm, and the other populations are generated randomly. Two methods are used to generate different sequences of jobs. In the first method, we use the well-known NEH[1] method to generate the sequence of jobs, the second method is a SPT[2]-based heuristic.

4.1.3 Crossover

Crossover is a significant operator to create the next generation. Two parents have to be selected among the existing solutions to create crossover offsprings. The algorithm tries to select the best solutions as parents to obtain decent offsprings with suitable fitness values. There are two common ways to select parents: Roulette wheel, Tournament. In the proposed algorithm, Roulette wheel is used as the selection operator. Parents are selected according to their fitness. Better solutions have more chances to be selected. After the parents are selected, one point crossover is used to generate new offsprings. After the crossover is implemented, some offsprings may need to be reformed to become a feasible solution. It occurs because of the identicality of some job numbers in the chromosome. Reformers remove the job number replications and replace the missing job numbers in the blank places. For more illustration of the crossover operation see Figure 4. Suppose we have nine jobs and the break-point is 6. As it is shown in offspring 1, job number 6 is repeated two times. The same situation is happened in offspring 2 for job 3, therefore, a reformation is done in the next step to correct the chromosomes.

[1] Nawaz, Enscore, Ham
[2] Shortest Processing Time

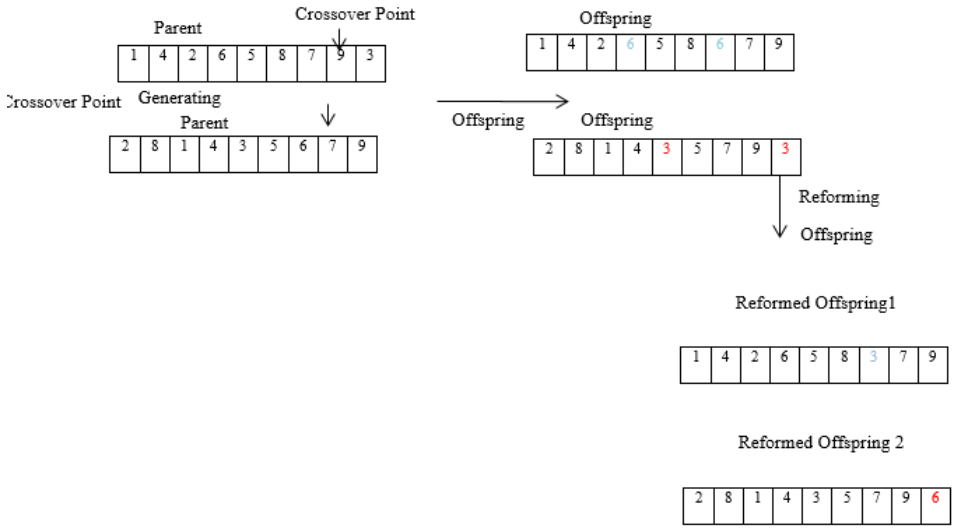

Fig. 4 An example of crossover operation for nine jobs

4.1.4 Mutation

After crossover, the strings are subjected to mutation. Mutation is a genetic operator used to maintain genetic diversity from one generation of a population of genetic algorithm chromosomes to the next. It is analogous to biological mutation. Mutation alters one or more gene values in a chromosome from its initial state. In mutation, the solution may change entirely from the previous solution. Hence GA can come to better solution by using mutation. Mutation occurs during evolution according to a user-definable mutation probability. This probability should be set low. If it is set too high, the search will turn into a primitive random search. The purpose of mutation in GAs is preserving and introducing diversity. Mutation should allow the algorithm to avoid local minima by preventing the population of chromosomes from becoming too similar to each other, thus slowing or even stopping evolution. This reasoning also explains the fact that most GA systems avoid only taking the fittest of the population in generating the next but rather a random (or semi-random) selection with a weighting toward those that are fitter. Referring to Abdoun et al. (2012), five modes of mutation are used in the proposed algorithm to keep diversity of chromosomes in all iterations.

4.1.4.1 Twors Mutation

Twors mutation allows the exchange of position of two genes randomly chosen.

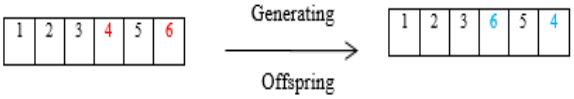

Fig. 5 Mutation operator TWORS

4.1.4.2 Centre inverse mutation (CIM)

The chromosome is divided into two sections. All genes in each section are copied and then inversely placed in the same section of a child.

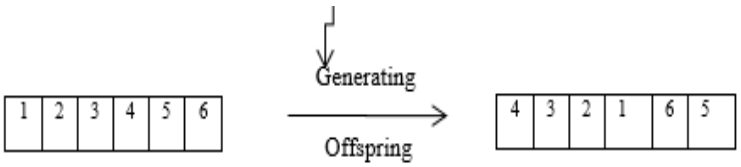

Fig. 6 *Mutation operator CIM*

4.1.4.3 Reverse Sequence Mutation (RSM)

In the reverse sequence mutation operator, we take a sequence S limited by two positions i and j randomly chosen, such that i<j. The gene order in this sequence will be reversed by the same way as what has been covered in the previous operation. The algorithm (Figure 7) shows the implementation of this mutation operator.

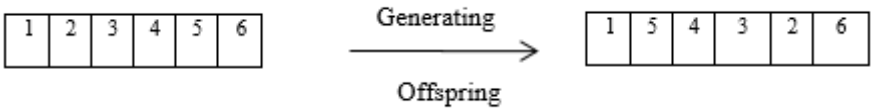

Fig. 7 *Mutation operator RSM*

4.1.4.4 Throas Mutation

We construct a sequence of three genes: the first is selected randomly and the two others are those two successors. Then, the last becomes the first of the sequence, the second becomes last and the first becomes the second in the sequence.

4.1.4.5 Thrors Mutation

Three genes are chosen randomly which shall take the different positions not necessarily successive i < j < l. the gene of the position i becomes in the position j and the one who was at this position will take the position l and the gene that has held this position takes the position i .

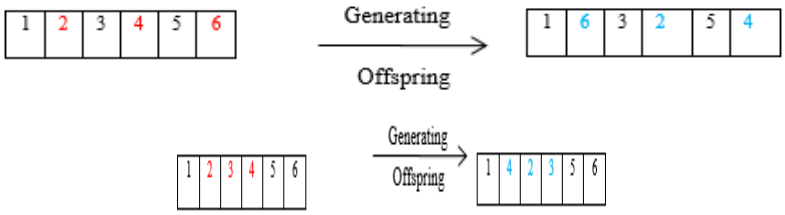

Fig. 8 *GA Operators*

4.1.5 Fitness Function

After performance of mutation and crossover operators, the problem has three population which is consist of offsprings, main population and mutants. So the chromosome of these three population according to the quality of their answers and depending on the objective function of the problem are joined together and compose the composed population. In other words the corresponding amount of each chromosome is in the objective function of the problem and the chromosomes with the best answer is chosen. in this paper the amount of the objective function is consist of energy costs and the costs related to makespan which is put according to the amount of buffers obtained among machines and put as fitness of each chromosome. As the size of the buffer gets bigger, the amount of objective function decreases. The reason is that bigger size of the buffer can decrease the number of blocks (blocking product line). So the number of the times that the machine goes stand by and also the number of the times that the machine shuts down and starts again for the operation (setup) decreases and so less energy is used. And also growing the size of buffer makespan decreases, so the total amount of objective function decreases and the fitness increases.

4.2 Simulated annealing

Kirkpatrick et al. (1983), were the first researchers who introduced simulated annealing. Simulated annealing (SA) is a stochastic iterative search algorithm which was successfully implemented by many researchers to solve different optimization problems. The first step of this algorithm deals with the creation of initial solution, then a specific number of neighbors is created for initial solution. In the next step, the neighbors of each solution are compared to each other and the best neighbor is selected. The difference between the objective (fitness) values of a solution (S) and its neighbor (S') is calculated as follows: $(S') - C(S)$, where $C(S)$ denotes the fitness value of solution S. In the problems with minimization objective functions, if < 0 , the base solution (S) is omitted and the best neighbor is replaced with it. Because we have obtained a solution with better objective value, but if $E \geq 0$, then the acceptance of the new (neighbor) solution depends on an exponential probability function called Boltzmann. It helps SA not to be trapped in local optimums, and its formula is as follows: $pr = \exp(^{-E}/_{T})$, where T is the Temperature of the algorithm which is reduced by the means of a cooling function in each iteration. This process is repeated for all solutions in all iterations until the stopping criterion is met. The outline of the algorithm is shown in Fig. 9.

Simulated annealing procedure
Initialization
Parameter setting
(Number of Neighbors, T_0, α, Stopping Criteria)
Generate initial solution by some heuristic methods
Repeat
Generate neighborhood solutions (S') for each solution
Select the best neighbor of each solution by the means of its objective value
if $E \geq 0$
Then *replace S' with S (downhill move)*
Else *replace S' with S using a probability function, $pr =$ $\exp(^{-E}/_{T})$, (uphill move)*
End if
If *the stop criterion is satisfied*
Then *stop*
Else $T = \alpha T$
End if

Fig. 9 Encoding, Initial solution and fitness function of SA

Encoding form, initial solutions and fitness function of SA are the same as GA and the same methods used in mutation of GA are used to create new neighbors of each solution in SA.

4.3. Parameters of GA and SA

Meta-heuristic algorithms are too sensitive to the value of their parameters and it is important to select a suitable parameter set for each algorithm. In this paper we selected the most effective parameters which seem to have efficient influences on the solutions. GA and SA algorithms are implemented to solve the problem with different solve the problem with different parameters and finally the most proper parameter is selected among the others Parameters of GA have set as follows.

- p_c, is the crossover percentage. Suppose the population size (popsize) is set to be 100, in each iteration we have to create 100 other offsprings, merge them with the old ones and select the best 100 chromosomes. There are two operators to create new offsprings, namely crossover and mutation. p_c is the proportion of the new offsprings made by crossover operator compared to the number of offsprings made by mutation. For example if $p_c = 0.7$, we will have 70 of the new offsprings made by crossover operator.
- p_m, is the mutation percentage. In the implemented GA, $p_m = 1 - p_c$. Hence following the above example, $p_m = 1 - 0.7$ and the number of new offsprings made by mutation operator will be equal to 30.
- Stop criterion is defined in section 5.

Parameters of SA have set as follows.

- Number of neighbors, is the number of new neighbors created for each solution.
- T_0, is the initial temperature. We have used the formula $T_0 = {}^{-E}/_{\ln(P_0)}$, to obtain the value of T_0, where E is the difference between the objective values of a solution and its best neighbor, P_0 is the acceptance probability of Boltzmann function in the 1^{st} iteration. The obtained value of T_0 in small size problems is 5.12 and for large scale problems we have used the value 6.64.
- α $(0 < \alpha < 1)$, is the cooling ratio. It is applied to reduce the temperature as the algorithm goes on.
- Stop criterion, is defined in section 5 (same as GA).

5. Computational experiments

Because of the lack of benchmarks for the studied problem, we have generated some sample instances to investigate the efficiency effectiveness of the proposed algorithms. Three types of instances have been generated, namely small size (ARFS), medium size (ARFM) and large size (ARFL) problems. 40 small size, 60 medium size and 48 large scale problems have been generated. Each problem instance is defined by 3 parameters which describe the size of the problem, these parameters are the number jobs (n), the number of machines (m), the number of buffers (b). For small size problems, number of jobs, machines and buffers are as follows respectively: {4, 6}, {2, 3, 5, 7}, $\{0,1,2,4,\infty\}$, for medium size instances, the numbers are {8, 10, 15}, {2, 3, 5, 7}, $\{0,1,2,4,\infty\}$ and for large scale problems, the numbers are: {20, 30, 40, 50}, {10, 12, 15}, $\{0,2,4,\infty\}$.

The minimum processing time of jobs, i.e. the time with full power of machines are generated randomly by the uniform distribution [1,100]. Set up energy of machine of each machine is generated by the uniform distribution of $\left[\frac{11}{10}, \frac{50}{10}\right]$. Idle power of each machine is generated by the uniform distribution of $\left[\frac{2}{10}, \frac{50}{10}\right]$. Set up time of each machine is generated by the uniform distribution of [1,20]. Total overhead costs is generated by the uniform distribution of $\left[\frac{5}{60}, \frac{50}{60}\right] \times m$, that m is number of machines. Total electricity costs is generated by the uniform distribution of $\left[\frac{4}{60}, \frac{30}{60}\right]$.

We have used five different values of 0.1, 0.3, 0.5, 0.7 and 0.9 for crossover percentage (p_c) in GA, and therefore the values of p_m will be equal to 0.9, 0.7, 0.5, 0.3 and 0.1 respectively.

As mentioned in section 4, we have used the formula: $T_0 = {}^{-E}/_{\ln(P_0)}$ to calculate the values of T_0, and two different values have been obtained for small and large size problems. The values 20, 30 and 0.97, 0.985, 0.99 are selected for parameters nn, α, respectively. So we have six combinations of the values of these two parameters for SA.

For small size problems, we can obtain the optimal solutions by using Lingo software. But because of the complexity of the problem, Lingo is unable to obtain the optimal solutions in time limit of 300 minutes for some medium and all large size problems.

The results of the numerical experiments are given in Tables 2 to 8. The bolded values in column four of Tables 1 to 4 are the optimal solutions obtained by Lingo.

We have used equation (5) to calculate Dev value for the obtained solutions.

$$Dev = \frac{Obj(GA,SA) - Best\ Obj}{Best\ Obj} \times 100 \quad (5)$$

In which $Obj(GA,SA)$ shows the the answer of GA or SA algorithms and $Best\ Obj$ shows the best answer obtained by the lingo software and SA and GA algorithms.

According to Tables 2 & 3, genetic algorithm with crossover percentage of 0.1 has the best performance and the lowest Dev value for small size problems in comparison with the other parameter settings. Tables 4 and 5 show the same results for medium size problems.

Figure 10 shows the objective values obtained by Lingo and GA with crossover percentage of 0.1 for small and medium size problems. It shows that the performance of GA is better than Lingo to obtain optimal or near optimum solutions.

In the large size problems for obtaining the amount of Dev we use (6) formula. In this formula, $Obj(GA,SA)$ shows the answer of GA or SA algorithms and $Best\ Obj$ shows the best answer obtained by SA and GA algorithms.

$$Dev = \frac{Obj(GA,SA) - Best\ Obj}{Best\ Obj} \times 100 \quad (6)$$

Tables 6, 7, 8 show the objective and Dev values for large size problems.

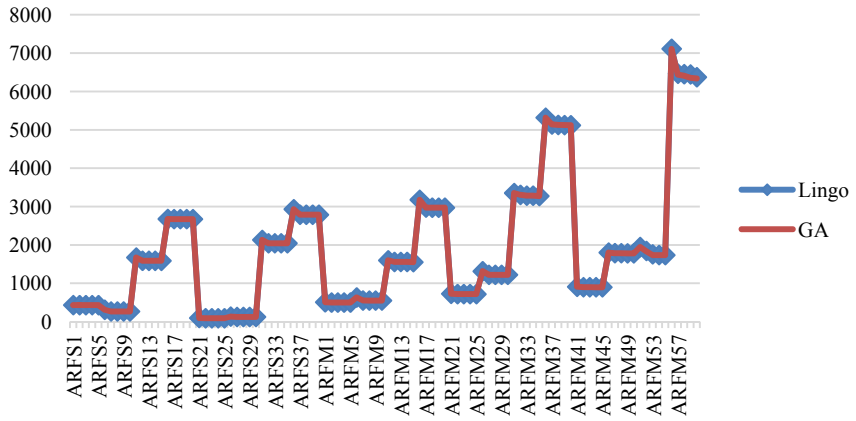

Fig. 10 Comparison between the objectives of Lingo and GA

To have a fair comparison between two algorithms, a time-based stop criterion of $0.2 \times n \times m \times (b+1)$ seconds is used for both algorithms. Table 6 shows the best, worst and mean objective values obtained by GA for each values of after five runs for each instance, and Table 7 shows the results obtained by SA. Bolded values in these tables are the best objective values those have been reached by the use of all 11 different parameter setting of both GA and SA algorithms. Table 7 shows the *Dev* value of all different cases of GA and SA and the mean deviations show that GA with the p_c of 0.1 has the best performance for large scale problems in comparison with the others.

Table 3 Objective and Dev values of Lingo and GA for small size problems

Instance Name	n	m	b	Lingo	Makespan	$p_c = 0.1$		$p_c = 0.3$		$p_c = 0.9$	
						Mean	Dev	Mean	Dev	Mean	Dev
ARFS1	4	2	0	**432.1326**	317	432.1326	0.00000	432.1326	0.00000	432.1326	0.00000
ARFS2	4	2	1	**430.5105**	315	430.5105	0.00000	430.5105	0.00000	430.5105	0.00000
ARFS3	4	2	2	**430.5105**	315	430.5105	0.00000	430.5105	0.00000	430.5105	0.00000
ARFS4	4	2	4	**430.5105**	315	430.5105	0.00000	430.5105	0.00000	430.5105	0.00000
ARFS5	4	2	∞	**430.5105**	315	430.5105	0.00000	430.5105	0.00000	430.5105	0.00000
ARFS6	4	3	0	**310.9288**	286	310.9288	0.00000	310.9288	0.00000	310.9288	0.00000
ARFS7	4	3	1	**267.3215**	279	267.3215	0.00000	267.3215	0.00000	267.3215	0.00000
ARFS8	4	3	2	**267.3215**	279	267.3215	0.00000	267.3215	0.00000	267.3215	0.00000
ARFS9	4	3	4	**267.3215**	279	267.3215	0.00000	267.3215	0.00000	267.3215	0.00000
ARFS10	4	3	∞	**267.3215**	279	267.3215	0.00000	267.3215	0.00000	267.3215	0.00000
ARFS11	4	5	0	**1671.6182**	468	1671.6182	0.00000	1671.6182	0.00000	1671.6182	0.00000
ARFS12	4	5	1	**1587.6182**	425	1587.6182	0.00000	1587.6182	0.00000	1587.6182	0.00000
ARFS13	4	5	2	**1586.7947**	423	1586.7947	0.00000	1586.7947	0.00000	1586.7947	0.00000
ARFS14	4	5	4	**1586.7947**	423	1586.7947	0.00000	1586.7947	0.00000	1586.7947	0.00000
ARFS15	4	5	∞	**1586.7947**	423	1586.7947	0.00000	1586.7947	0.00000	1586.7947	0.00000
ARFS16	4	7	0	**2676.8394**	556	2676.8394	0.00000	2676.8394	0.00000	2676.8394	0.00000
ARFS17	4	7	1	**2673.4894**	552	2673.4894	0.00000	2673.4894	0.00000	2673.4894	0.00000
ARFS18	4	7	2	**2673.3094**	552	2673.3094	0.00000	2673.3094	0.00000	2673.3094	0.00000
ARFS19	4	7	4	**2673.3094**	552	2673.3094	0.00000	2673.3094	0.00000	2673.3094	0.00000
ARFS20	4	7	∞	**2673.3094**	552	2673.3094	0.00000	2673.3094	0.00000	2673.3094	0.00000
ARFS21	6	2	0	**95.0842**	351	95.0842	0.00000	95.0842	0.00000	95.1397	0.05800
ARFS22	6	2	1	**89.5112**	334	89.5112	0.00000	89.5112	0.00000	89.5112	0.00000
ARFS23	6	2	2	**89.0778**	334	89.0778	0.00000	89.0778	0.00000	89.0778	0.00000
ARFS24	6	2	4	**89.0778**	334	89.0778	0.00000	89.0778	0.00000	89.0778	0.00000
ARFS25	6	2	∞	**89.0778**	334	89.0778	0.00000	89.0778	0.00000	89.0778	0.00000
ARFS26	6	3	0	**132.8124**	528	132.8124	0.00000	132.8124	0.00000	133.7791	0.72700
ARFS27	6	3	1	**123.4331**	453	123.4331	0.00000	123.4331	0.00000	123.833	0.32300
ARFS28	6	3	2	**123.2332**	453	123.2332	0.00000	123.2332	0.00000	123.3664	0.10800
ARFS29	6	3	4	**123.2332**	453	123.2332	0.00000	123.2332	0.00000	123.2332	0.00000
ARFS30	6	3	∞	**123.2332**	453	123.2332	0.00000	123.2332	0.00000	123.2332	0.00000
ARFS31	6	5	0	**2131.2345**	625	2131.2345	0.00000	2131.2345	0.00000	2131.2345	0.00000
ARFS32	6	5	1	**2043.534**	604	2043.534	0.00000	2043.534	0.00000	2043.534	0.00000
ARFS33	6	5	2	**2043.534**	604	2043.534	0.00000	2043.534	0.00000	2043.534	0.00000
ARFS34	6	5	4	**2043.534**	604	2043.534	0.00000	2043.534	0.00000	2043.534	0.00000
ARFS35	6	5	∞	**2043.534**	604	2043.534	0.00000	2043.534	0.00000	2043.534	0.00000
ARFS36	6	7	0	**2930.88**	862	2930.88	0.00000	2930.88	0.00000	2930.88	0.00000
ARFS37	6	7	1	**2787.29**	828	2787.29	0.00000	2787.29	0.00000	2787.29	0.00000
ARFS38	6	7	2	**2787.29**	828	2787.29	0.00000	2787.29	0.00000	2787.29	0.00000
ARFS39	6	7	4	**2787.29**	828	2787.29	0.00000	2787.29	0.00000	2787.29	0.00000
ARFS40	6	7	∞	**2787.29**	828	2787.29	0.00000	2787.29	0.00000	2787.29	0.00000
Mean							0.00000		0.00000		0.03040

Table 4 Objective and *Dev* values of Lingo and SA for small size problems

Instance Name	n	m	b	Lingo	Makespan	$\alpha = 0.97$ $nn = 20$		$\alpha = 0.97$ $nn = 30$		$\alpha = 0.985$ $nn = 20$		$\alpha = 0.99$ $nn = 30$	
						Mean	Dev	Mean	Dev	Mean	Dev	Mean	Dev
ARFS1	4	2	0	432.1326	317	432.1326	0.00000	432.1326	0.00000	432.1326	0.00000	432.1326	0.00000
ARFS2	4	2	1	430.5105	315	430.5105	0.00000	430.5105	0.00000	430.5105	0.00000	430.5105	0.00000
ARFS3	4	2	2	430.5105	315	430.5105	0.00000	430.5105	0.00000	430.5105	0.00000	430.5105	0.00000
ARFS4	4	2	4	430.5105	315	430.5105	0.00000	430.5105	0.00000	430.5105	0.00000	430.5105	0.00000
ARFS5	4	2	∞	430.5105	315	430.5105	0.00000	430.5105	0.00000	430.5105	0.00000	430.5105	0.00000
ARFS6	4	3	0	310.9288	286	310.9288	0.00000	310.9288	0.00000	310.9288	0.00000	310.9288	0.00000
ARFS7	4	3	1	267.3215	279	267.3215	0.00000	267.3215	0.00000	267.3215	0.00000	267.3215	0.00000
ARFS8	4	3	2	267.3215	279	267.3613	0.01500	267.3412	0.00007	267.3412	0.00700	267.3613	0.01500
ARFS9	4	3	4	267.3215	279	267.3215	0.00000	267.3215	0.00000	267.3517	0.01100	267.3215	0.00000
ARFS10	4	3	∞	267.3215	279	267.3215	0.00000	267.3314	0.00400	267.3215	0.00000	267.3215	0.00000
ARFS11	4	5	0	1671.6182	468	1671.6182	0.00000	1681.6182	0.60000	1671.6182	0.00000	1681.5390	0.60000
ARFS12	4	5	1	1587.6182	425	1587.6182	0.00000	1587.6182	0.00000	1588.6512	0.06500	1587.6182	0.00000
ARFS13	4	5	2	1586.7947	423	1586.7947	0.00000	1586.7947	0.00000	1586.7947	0.00000	1586.7947	0.00000
ARFS14	4	5	4	1586.7947	423	1586.7947	0.00000	1586.7947	0.00000	1586.7947	0.00000	1586.7947	0.00000
ARFS15	4	5	∞	1586.7947	423	1586.7947	0.00000	1586.7947	0.00000	1586.7947	0.00000	1586.7947	0.00000

ARFS16	4	7	0	2676.8394	556	2676.8394	0.00000	2676.8394	0.00000	2676.8394	0.00000	2676.8394	0.00000
ARFS17	4	7	1	2673.4894	552	2673.4894	0.00000	2673.4894	0.00000	2673.4894	0.00000	2673.4894	0.00000
ARFS18	4	7	2	2673.3094	552	2673.3094	0.00000	2673.3094	0.00000	2673.3094	0.00000	2673.3094	0.00000
ARFS19	4	7	4	2673.3094	552	2673.3094	0.00000	2673.3094	0.00000	2673.3094	0.00000	2673.3094	0.00000
ARFS20	4	7	∞	2673.3094	552	2673.3094	0.00000	2673.3094	0.00000	2673.3094	0.00000	2673.3094	0.00000
ARFS21	6	2	0	95.0842	351	95.0842	0.00000	95.0842	0.00000	95.0842	0.00000	95.0842	0.00000
ARFS22	6	2	1	89.5112	334	89.5112	0.00000	89.5112	0.00000	89.5112	0.00000	89.5112	0.00000
ARFS23	6	2	2	89.0778	334	89.0778	0.00000	89.0778	0.00000	89.0778	0.00000	89.0778	0.00000
ARFS24	6	2	4	89.0778	334	89.0778	0.00000	89.0778	0.00000	89.0778	0.00000	89.0778	0.00000
ARFS25	6	2	∞	89.0778	334	89.0778	0.00000	89.0778	0.00000	89.0778	0.00000	89.0778	0.00000
ARFS26	6	3	0	132.8124	528	132.8124	0.00000	132.8124	0.00000	132.8124	0.00000	132.8124	0.00000
ARFS27	6	3	1	123.4331	453	123.4331	0.00000	123.4331	0.00000	123.4331	0.00000	123.4331	0.00000
ARFS28	6	3	2	123.2332	453	123.2332	0.00000	123.2332	0.00000	123.2332	0.00000	123.2332	0.00000
ARFS29	6	3	4	123.2332	453	123.2332	0.00000	123.2332	0.00000	123.2332	0.00000	123.2332	0.00000
ARFS30	6	3	∞	123.2332	453	123.2332	0.00000	123.2332	0.00000	123.2332	0.00000	123.2332	0.00000
ARFS31	6	5	0	2131.2345	625	2131.2345	0.00000	2131.2345	0.00000	2131.2345	0.00000	2131.2345	0.00000
ARFS32	6	5	1	2043.534	604	2043.534	0.00000	2043.534	0.00000	2043.534	0.00000	2043.534	0.00000
ARFS33	6	5	2	2043.534	604	2043.534	0.00000	2043.534	0.00000	2043.534	0.00000	2043.534	0.00000
ARFS34	6	5	4	2043.534	604	2043.534	0.00000	2043.534	0.00000	2043.534	0.00000	2043.534	0.00000
ARFS35	6	5	∞	2043.534	604	2043.534	0.00000	2043.534	0.00000	2043.534	0.00000	2043.534	0.00000
ARFS36	6	7	0	2930.88	862	2930.88	0.00000	2930.88	0.00000	2930.88	0.00000	2930.88	0.00000
ARFS37	6	7	1	2787.29	828	2787.29	0.00000	2787.29	0.00000	2787.29	0.00000	2787.29	0.00000
ARFS38	6	7	2	2787.29	828	2787.29	0.00000	2787.29	0.00000	2787.29	0.00000	2787.29	0.00000
ARFS39	6	7	4	2787.29	828	2798.22	0.39200	2797.15	0.35400	2788.39	0.00039	2796.36	0.32500
ARFS40	6	7	∞	2787.29	828	2797.18	0.35500	2787.29	0.00000	2797.14	0.00353	2787.29	0.00000
Mean							0.01900		0.02400		0.01200		0.02300

Table 5 Objective and *Dev* values of Lingo and GA for medium size problems

Instance Name	n	m	b	Lingo	Makespan	$p_c = 0.1$		$p_c = 0.3$		$p_c = 0.9$	
						Mean	Dev	Mean	Dev	Mean	Dev
ARFM1	8	2	0	507.3121	546	507.3121	0.00000	507.3121	0.00000	507.3121	0.00000
ARFM2	8	2	1	498.9009	434	498.9009	0.00050	499.1509	0.05011	499.7409	0.16837
ARFM3	8	2	2	498.9009	434	498.9009	0.00000	498.9009	0.00000	498.9009	0.00000
ARFM4	8	2	4	498.9009	434	498.9009	0.00000	498.9009	0.00000	498.9009	0.00000
ARFM5	8	2	∞	498.9009	434	498.9009	0.00000	498.9009	0.00000	498.9009	0.00000
ARFM6	8	3	0	630.81	642	630.81	0.00000	630.81	0.00000	630.81	0.00000
ARFM7	8	3	1	552.15	663	552.15	0.00000	552.15	0.00000	552.53	0.06882
ARFM8	8	3	2	549.69	629	549.69	0.00000	549.75	0.01092	549.84	0.02729
ARFM9	8	3	4	549.69	629	549.69	0.00050	549.69	0.00000	549.69	0.00000
ARFM10	8	3	∞	549.69	629	549.69	0.00000	549.69	0.00000	549.69	0.00000
ARFM11	8	5	0	1598.5583	910	1598.5583	0.00000	1598.5583	0.00000	1601.349	0.17458
ARFM12	8	5	1	1557.0355	733	1557.0355	0.00000	1562.0662	0.32309	1557.0355	0.00000
ARFM13	8	5	2	1557.0355	733	1557.0355	0.00000	1557.0355	0.00000	1557.3954	0.02311
ARFM14	8	5	4	1557.0355	733	1557.0355	0.00000	1557.097	0.00000	1557.3731	0.02168
ARFM15	8	5	∞	1552.0662	700	1552.0662	0.00000	1552.0755	0.00000	1552.0762	0.00064
ARFM16	8	7	0	3177.487	940	3177.487	0.01197	3177.3083	0.00000	3177.43	0.01018
ARFM17	8	7	1	2974.37	985	2974.37	0.00000	2975	0.02118	2975	0.02118
ARFM18	8	7	2	2971.15	923	2971.15	0.00050	2971.4417	0.01397	2971.4417	0.01397
ARFM19	8	7	4	2971.15	923	2971.15	0.00000	2971.4417	0.00000	2971.505	0.01195
ARFM20	8	7	∞	2971.15	923	2971.15	0.00000	2971.4417	0.01128	2971.6283	0.01756
ARFM21	10	2	0	727.7255	729	727.6255	0.08437	727.0121	0.00000	727.5266	0.07077
ARFM22	10	2	1	718.5122	681	718.5122	0.00000	718.5122	0.00000	718.5122	0.00392
ARFM23	10	2	2	718.5122	681	718.5122	0.00050	718.5122	0.00000	718.5122	0.00000
ARFM24	10	2	4	718.5122	681	718.5122	0.00000	718.5122	0.00000	718.5122	0.00000
ARFM25	10	2	∞	718.5122	681	718.5122	0.00000	718.5112	0.00000	718.5122	0.00014
ARFM26	10	3	0	1315.0115	829	1315.0115	0.00000	1319.3836	0.33248	1315.8957	0.06724
ARFM27	10	3	1	1222.5296	789	1222.5296	0.00000	1222.5296	0.00000	1223.494	0.07889
ARFM28	10	3	2	1222.5296	789	1222.5296	0.00000	1222.5296	0.00000	1222.9884	0.03753
ARFM29	10	3	4	1222.5296	789	1222.5296	0.00000	1222.5296	0.00000	1222.8329	0.02481
ARFM30	10	3	∞	1222.5296	789	1222.5296	0.00000	1222.5296	0.00000	1222.8329	0.02481

Table 6 Objective and *Dev* values of Lingo and GA for medium size problems

Instance Name	n	m	b	Lingo	Makespan	$p_c = 0.1$		$p_c = 0.3$		$p_c = 0.9$	
						Mean	Dev	Mean	Dev	Mean	Dev
ARFM31	10	5	0	3352.0593	1063	3352.0593	0.05096	3351.9173	0.04672	3352.6818	0.06954
ARFM32	10	5	1	3305.1966	1036	3305.1966	0.04833	3303.6001	0.00000	3308.4358	0.14638
ARFM33	10	5	2	3283.7372	1010	3283.7372	0.03673	3283.7372	0.03673	3283.7372	0.03673
ARFM34	10	5	4	3283.7372	1010	3283.7372	0.00000	3283.7372	0.00000	3283.7372	0.00000
ARFM35	10	5	∞	3274.9061	990	3274.9061	0.00000	3274.9061	0.00000	3274.9061	0.00000
ARFM36	10	7	0	5318.0434	1124	5318.0434	0.00000	5318.0434	0.00000	5318.0434	0.00668
ARFM37	10	7	1	5139.9375	1028	5139.9375	0.04449	5139.688	0.03963	5138.0363	0.00748
ARFM38	10	7	2	5128.0152	1012	5128.0152	0.05224	5126.5662	0.02397	5127.6585	0.04528
ARFM39	10	7	4	5123.2875	1020	5123.2875	0.03382	5123.66	0.04110	5122.1375	0.01137
ARFM40	10	7	∞	5119.0775	1038	5119.0775	0.01240	5119.7675	0.02588	5119.2463	0.01570
ARFM41	15	2	0	910.402	853	910.402	0.04236	910.0165	0.00000	910.0815	0.00714
ARFM42	15	2	1	898.723	761	898.5886	0.27978	897.0869	0.11220	896.0815	0.00000
ARFM43	15	2	2	898.723	761	898.5886	0.26708	896.912	0.08000	896.7802	0.06530
ARFM44	15	2	4	898.723	761	896.561	0.10287	895.71	0.00000	896.5114	0.09733
ARFM45	15	2	∞	898.723	761	896.561	0.16009	895.71	0.06502	896.5114	0.15455
ARFM46	15	3	0	1801.24	1125	1801.24	0.09150	1799.5933	0.00000	1804.5466	0.27525
ARFM47	15	3	1	1786.9466	1022	1782.9466	0.03553	1782.3133	0.00000	1782.3866	0.00411
ARFM48	15	3	2	1784.4	973	1782.2	0.16890	1781.3866	0.12318	1781.7266	0.14229
ARFM49	15	3	4	1781.27	1119	1781.2	0.11118	1781.3266	0.11829	1781.83	0.14659
ARFM50	15	3	∞	1781.27	1119	1781.2	0.17017	1781.3266	0.17729	1781.5033	0.18723
ARFM51	15	5	0	1945.5209	1287	1945.5209	0.08128	1944.6771	0.03787	1945.1943	0.06448
ARFM52	15	5	1	1842.97	1062	1835.8887	0.00000	1837.7205	0.09978	1840.6977	0.26194
ARFM53	15	5	2	1756.62	1052	1733.1944	0.00000	1733.7795	0.03376	1735.7731	0.14878
ARFM54	15	5	4	1737.433	1063	1733.1028	0.03390	1732.5154	0.00000	1736.2767	0.21710
ARFM55	15	5	∞	1734.45	960	1732.4865	0.09958	1732.0066	0.07185	1733.5913	0.16341
ARFM56	15	7	0	7113.25	1567	7106.1205	0.00000	7109.8561	0.06221	7107.017	0.02225
ARFM57	15	7	1	6463.77	1380	6438.2879	0.07591	6433.4041	0.00000	6439.655	0.09716
ARFM58	15	7	2	6451.37	1508	6409.8226	0.09076	6408.5056	0.07020	6408.8179	0.07507
ARFM59	15	7	4	6443.723	1480	6356.9092	0.06981	6355.0682	0.04083	6356.6365	0.06552
ARFM60	15	7	∞	6372.13	1288	6340.0028	0.00000	6343.7085	0.05845	6341.8284	0.02879
Mean							0.03563		0.03746		0.05718

Table 7 Objective and *Dev* values of Lingo and SA for medium size problems

Instance Name	n	m	b	Lingo	Makespan	$\alpha = 0.97$ $nn = 20$		$\alpha = 0.97$ $nn = 30$		$\alpha = 0.985$ $nn = 20$		$\alpha = 0.99$ $nn = 30$		
						Mean	Dev	Mean	Dev	Mean	Dev	Dev	Mean	Dev
ARFM1	8	2	0	507.3121	546	507.3121	0.00000	507.3121	0.00000	507.3121	0.00000	0.00000	507.3121	0.00000
ARFM2	8	2	1	498.9009	434	498.9009	0.00000	498.9009	0.00000	498.9009	0.00000	0.00000	498.9009	0.00000
ARFM3	8	2	2	498.9009	434	498.9009	0.00000	498.9009	0.00000	498.9009	0.00000	0.00000	498.9009	0.00000
ARFM4	8	2	4	498.9009	434	498.9009	0.00000	498.9009	0.00000	498.9009	0.00000	0.00000	498.9009	0.00000
ARFM5	8	2	∞	498.9009	434	498.9100	0.00000	498.9009	0.00000	498.9009	0.00000	0.04398	498.9009	0.00000
ARFM6	8	3	0	630.81	642	630.81	0.00000	630.81	0.00000	630.81	0.00000	0.00000	630.81	0.00000
ARFM7	8	3	1	552.15	663	552.15	0.00000	552.15	0.00000	552.15	0.00000	0.00000	552.15	0.00000
ARFM8	8	3	2	549.69	629	549.69	0.00000	549.73	0.00000	549.81	0.02183	0.00000	549.69	0.00000

ARFM9	8	3	4	549.69	629	549.71	0.00000	549.69	0.00000	549.69	0.00000	0.00000	549.80	0.02001
ARFM10	8	3	∞	549.69	629	549.69	0.00000	549.69	0.00000	549.69	0.00000	0.00000	549.69	0.00000
ARFM11	8	5	0	1598.5583	910	1598.5583	0.00000	1598.5671	0.00000	1598.5583	0.00000	0.00000	1598.5583	0.00000
ARFM12	8	5	1	1557.0355	733	1557.1241	0.00000	1557.1352	0.00000	1557.0355	0.00000	0.07255	1559.5674	0.16261
ARFM13	8	5	2	1557.0355	733	1557.0355	0.00000	1557.0355	0.00000	1557.0355	0.00000	0.00000	1557.0355	0.00000
ARFM14	8	5	4	1557.0355	733	1557.0355	0.00000	1557.0355	0.00000	1557.0355	0.00000	000000	1557.0355	0.00000
ARFM15	8	5	∞	1552.0662	700	1552.0662	0.00000	1552.0662	0.00000	1552.0662	0.00000	0.00000	1552.0662	0.00000
ARFM16	8	7	0	3177.487	940	3177.487	0.01197	3177.487	0.01197	3177.487	0.01197	0.01197	3177.487	0.01197
ARFM17	8	7	1	2974.37	985	2974.37	0.00000	2974.37	0.00000	2974.37	0.00000	0.00000	2974.37	0.00000
ARFM18	8	7	2	2971.15	923	2972.1451	0.03764	2971.15	0.00000	2971.15	0.00000	0.00000	2971.1443	0.00000
ARFM19	8	7	4	2971.15	923	2971.15	0.00000	2971.15	0.00000	2971.15	0.00000	0.00000	2971.15	0.00000
ARFM20	8	7	∞	2971.15	923	2971.1661	0.00000	2971.1743	0.00000	2971.15	0.00000	0.00000	2971.15	0.00000
ARFM21	10	2	0	727.7255	729	727.6343	0.08558	727.4466	0.05977	727.6341	0.08556	0.09813	727.7255	0.09813
ARFM22	10	2	1	718.5122	681	718.5034	0.00000	718.4849	0.00000	718.4840	0.00000	0.00000	718.5122	0.00000
ARFM23	10	2	2	718.5122	681	718.5122	0.00000	718.5122	0.00000	718.5122	0.00000	0.00000	718.5122	0.00000
ARFM24	10	2	4	718.5122	681	718.5122	0.00000	718.5122	0.00000	718.5122	0.00000	0.00000	718.5122	0.00000
ARFM25	10	2	∞	718.5122	681	718.5122	0.00000	718.5122	0.00000	718.5122	0.00000	0.00000	718.5122	0.00000
ARFM26	10	3	0	1315.0115	829	1315.0115	0.00000	1315.0115	0.00000	1315.0115	0.00000	0.00000	1315.0115	0.00000
ARFM27	10	3	1	1222.5296	789	1222.5296	0.00000	1222.5296	0.00000	1222.5296	0.00000	0.00000	1222.5296	0.00000
ARFM28	10	3	2	1222.5296	789	1222.5376	0.00000	1222.5296	0.00000	1222.5481	0.00000	0.00000	1222.5296	0.00000
ARFM29	10	3	4	1222.5296	789	1222.5296	0.00000	1222.5296	0.00000	1222.5296	0.00000	0.00000	1222.5296	0.00000
ARFM30	10	3	∞	1222.5296	789	1222.5296	0.00000	1222.5296	0.00000	1222.5296	0.00000	0.00000	1222.5296	0.00000

Table 8 Objective and *Dev* values of Lingo and SA for medium size problems

Instance Name	n	m	b	Lingo	Makespan	$\alpha = 0.97$ $nn = 20$		$\alpha = 0.97$ $nn = 30$		$\alpha = 0.985$ $nn = 20$		$\alpha = 0.99$ $nn = 30$	
						Mean	Dev	Mean	Dev	Mean	Dev	Mean	Dev
ARFM31	10	5	0	3352.0593	1063	3351.6362	0.00000	3352.0044	0.04932	3352.0418	0.05043	3352.0593	0.05096
ARFM32	10	5	1	3305.1966	1036	3305.1746	0.03833	3305.1966	0.04833	3305.1966	0.04833	3305.1966	0.04833
ARFM33	10	5	2	3283.7372	1010	3283.7372	0.04766	3283.7372	0.03673	3283.7372	0.03673	3283.7372	0.03673
ARFM34	10	5	4	3283.7372	1010	3283.7372	0.03673	3283.7372	0.00000	3283.7372	0.00000	3283.7372	0.00000
ARFM35	10	5	∞	3274.9061	990	3274.9061	0.00000	3274.9061	0.00000	3274.9061	0.00000	3274.9061	0.00000
ARFM36	10	7	0	5318.0434	1124	5318.0434	0.00000	5318.0434	0.00000	5318.0434	0.00000	5318.0434	0.00000
ARFM37	10	7	1	5139.9375	1028	5139.9375	0.00000	5139.9375	0.04449	5139.9375	0.04449	5139.9375	0.04449
ARFM38	10	7	2	5128.0152	1012	5129.1943	0.04449	5129.1774	0.07492	5139.9375	0.28486	5139.9375	0.28486
ARFM39	10	7	4	5123.2875	1020	5124.956	0.07525	5124.01991	0.04812	5123.2875	0.03382	5124.2341	0.05231

ARFM40	10	7	∞	5119.0775	1038	5119.0019	0.06640	5119.0118	0.01112	5119.0775	0.01240	5118.4428	0.00000
ARFM41	15	2	0	910.402	853	910.3212	0.01092	910.2115	0.02143	910.402	0.04236	910.402	0.04236
ARFM42	15	2	1	898.723	761	898.666	0.03348	898.594	0.28039	898.723	0.29478	897.2341	0.12863
ARFM43	15	2	2	898.723	761	896.195	0.28842	896.9124	0.08005	898.2150	0.22540	898.723	0.28208
ARFM44	15	2	4	898.723	761	896.012	0.00000	896.8150	0.13122	896.7112	0.11964	897.1943	0.17357
ARFM45	15	2	∞	898.723	761	895.1963	0.04157	898.723	0.40162	896.1542	0.11464	895.1280	0.00000
ARFM46	15	3	0	1801.24	1125	1801.24	0.00000	1801.24	0.09150	1800.8141	0.06784	1801.24	0.09150
ARFM47	15	3	1	1786.9466	1022	1783.9370	0.09150	1783.6050	0.07247	1785.3387	0.16975	1784.6243	0.12966
ARFM48	15	3	2	1784.4	973	1780.19	0.09110	1779.6112	0.02339	1779.1950	0.00000	1780.4370	0.06981
ARFM49	15	3	4	1781.27	1119	1779.2219	0.05592	1779.3114	0.00000	1779.2219	0.00000	1781.27	0.11511
ARFM50	15	3	∞	1781.27	1119	1779.1871	0.00000	1778.1741	0.00000	1781.27	0.17411	1778.237	0.00000
ARFM51	15	5	0	1945.5209	1287	1944.922	0.05697	1944.1774	0.01217	1944.2050	0.01359	1945.388	0.07444
ARFM52	15	5	1	1842.97	1062	1840.568	0.05047	1840.7544	0.26503	1839.371	0.18968	1839.5367	0.19870
ARFM53	15	5	2	1756.62	1052	1744.9271	0.25488	1740.9066	0.44497	1739.7095	0.37590	1739.637	0.37172
ARFM54	15	5	4	1737.433	1063	1735.906	0.67694	1735.7441	0.18636	1733.7536	0.07147	1735.196	0.15472
ARFM55	15	5	∞	1734.45	960	1732.186	0.19570	1731.295	0.03074	1732.687	0.11116	1731.225	0.02669
ARFM56	15	7	0	7113.25	1567	7108.713	0.08222	7108.435	0.04221	7106.5365	0.01549	7105.4357	0.00000
ARFM57	15	7	1	6463.77	1380	6442.763	0.04612	6441.75	0.12973	6435.2766	0.02911	6441.4266	0.12470
ARFM58	15	7	2	6451.37	1508	6421.401	0.14547	6419.7475	0.24574	6417.0431 3313131	0.20351	6419.23	0.23766
ARFM59	15	7	4	6443.723	1480	6368.6273	0.27156	6365.5124	0.20524	6368.476	0.25190	6367.538	0.23713
ARFM60	15	7	∞	6372.13	1288	6344.21	0.25428	6341.143	0.01798	6343.14	0.04948	6344.151	0.06543
Mean							0.05152		0.05111		0.05250		0.05557

The mathematical model is coded and solved by the modeling language Lingo 9.0. Meta-heuristic algorithms are coded in Matlab software, version 2013. A personal computer with the configuration of Core i5 2.5 GHz and 4 GB Ram is applied to solve the test problems.

In the next part the best answers obtained from lingo software and two GA and SA algorithms in the 8 to 10 tables are examined, and the percentage of using *stand by* and *turn off/turn on* in the answers according to the factor of the buffer numbers in theses tables are presented. As shown in the tables increasing the amount of buffer factor the number of setting up decreases. So the number of stand by increases and the number of *turn off/turn on* decreases.

Table 9: Percentage of using stand by and turn off/turn on in small size problems according to the factor of the buffer numbers

Buffer	Stand by	turn off/turn on
0	24%	76%
1	32%	68%
2	44%	56%
4	59%	41%
∞	86%	14%

Table 10: Percentage of using stand by and turn off/turn on in medium size problems according to the factor of the buffer numbers

Buffer	Stand by	turn off/turn on
0	27%	73%
1	35%	65%
2	42%	58%
4	63%	37%
∞	91%	9%

Table 11: Percentage of using stand by and turn off/turn on in large size problems according to the factor of the buffer numbers

Buffer	Stand by	turn off/turn on
0	21%	79%
2	52%	48%
4	67%	33%
∞	97%	3%

5. Conclusion

In this paper we investigated the permutation flow shop scheduling problem with limited buffers and the objectives of the minimization of total energy consumption and makespan. We formulated a mathematical model for the described problem. Since the proposed problem is NP-hard, so two well-known meta-heuristics namely; genetic algorithm and simulated annealing, have been used to produce approximate solutions in a reasonable time. We generated three different sizes of the problem, small, medium and large size problems. Lingo was able to give us the exact solution for all small size problems in time limit of 300 minutes, but for medium and large scale problems, Lingo is inefficient, so GA and SA have been used to reach near optimal solutions. The computational experiments show that with the used parameter settings of the algorithms for all problem sizes GA outperforms SA. At the end the best answers obtained from lingo software and two GA and SA algorithms in the 8 to 10 tables are examined, and the percentage of using *stand by* and *turn off/turn on* in the answers according to the factor of the buffer numbers in theses tables are presented. For future work it's suggested to use some other metaheuristic methods to solve the problem and compare the solutions with the existing ones, or maybe suggest a new heuristic for the problem. In our future research, the proposed algorithm might be extended to other machine environments such as job shop. Another extension is considering multi-objective optimization method such as the Non-dominated Sorting Genetic Algorithm-II (NSGA-II) for the problem.

References

1. Bruzzone, A.A. Anghinolfi, G., Paolucci, D., & Tonelli, F.: *Energy-aware scheduling for improving manufacturing process sustainability: A mathematical model for flexible flow shops*. CIRP Annals-Manufacturing Technology, 61(1), 2012. P.459-462.
2. Dai, M., Tang, D., Giret, A., Salido, M.A., & Li, W.D.: *Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm*. Robotics and Computer-Integrated Manufacturing, 29(5), 2013. P. 418-429.
3. Fang, K., Uhan, N.A., Zhao, F., & Sutherland, J.W.: *Flow shop scheduling with peak power consumption constraints*. Annals of Operations Research, 206(1), 2013. P. 115-145.
4. Fang, K.T., & Lin, B.M. *Parallel-machine scheduling to minimize tardiness penalty and power cost*. Computers & Industrial Engineering, 64(1), 2013. P.224-234.
5. Halliday, D., Resnick, R., & Walker, J.: *Fundamentals of physics extended* (Vol. 1). John Wiley & Sons. 2010.
6. Kirkpatrick, S., Gelatt, C., & Vecchi, M.: *Optimization by simulated annealing*. Science (220/4598), 1993. P. 671–680.
7. Liu, Y., Dong, H., Lohse, N., Petrovic, S., & Gindy, N.: *An investigation into minimising total energy consumption and total weighted tardiness in job shop*s. Journal of Cleaner Production, 65, 2013. P. 87-96.
8. Luo, H., Du, B., Huang, G. Q., Chen, H., & Li, X.: *Hybrid flow shop scheduling considering machine electricity consumption cost*. International Journal of Production Economics, 146(2), 2013. P. 423-439.
9. Moon, J. Y., Shin, K., & Park, J.: *Optimization of production scheduling with time-dependent and machine-dependent electricity cost for industrial energy efficiency*. The International Journal of Advanced Manufacturing Technology, 68(1-4), 2013. P. 523-535.
10. Mouzon, G., & Yildirim, M.B.: *A framework to minimise total energy consumption and total tardiness on a single machine.* International Journal of Sustainable Engineering, 1(2), 2008. P. 105-116.
11. Shrouf, F., Ordieres-Meré, J., García-Sánchez, A., & Ortega-Mier, M.: *Optimizing the production scheduling of a single machine to minimize total energy consumption costs.* Journal of Cleaner Production, 67, 2014. P.197-207.
12. Strusevich, V.A.: *Two machine flow shop scheduling problem with no wait in process: Controllable machine speeds*. Discrete applied mathematics, 59(1), 1995. P. 75-86.
13. Abdoun, O., Abouchabaka, J., & Tajani, C.: *Analyzing the performance of mutation operators to solve the travelling salesman problem.* arXiv preprint arXiv, 2012. P.1203.3099.
14. Sharma, A., Zhao, F., & Sutherland, J.W.: *Econological scheduling of a manufacturing enterprise operating under a time-of-use electricity* tariff.Journal of Cleaner Production, 108, 2015. P. 256-270.
15. Mikhaylidi, Y., Naseraldin, H., & Yedidsion, L.: *Operations scheduling under electricity time-varying prices*. International Journal of Production Research, 53(23), 2015. P. 7136-7157.
16. Drake, R., Yildirim, M. B., Twomey, J. M., Whitman, L. E., Ahmad, J. S., & Lodhia, P.: *Data collection framework on energy consumption in manufacturing.* 2006.

Primary Paper Section: A

Secondary Paper Section: DF