Local Search Methods for Solving Single Machine with Family Setup Time to Minimize the Multi-Objective Function Problem

Ghufran Khalil Joad [1*], Hanan Ali Cheachan [2]

[1] Department of Mathematics, College of Sciences, Al-Mustansireyah University, Baghdad, Iraq

[2] Department of Mathematics, College of Sciences, Al-Mustansireyah University, Baghdad, Iraq

*ghufrankalil@gmail.com

ABSTRACT

In this paper, three of the local search algorithms are used Bee Colony Optimization (BCO),Invasive Weed Optimization (IWO) and genetic Algorithm (GA), in scheduling number of products n jobs on a single machine with setup time to minimize a multi-objective functions$\sum_{j=1}^{n}(1-e^{-rc}jf)$ and T_{max}discount total completion time and maximum tardiness respectively, which is denoted as $1/s_f/\sum_{f=1}^{F}\sum_{j=1}^{n_f}((1-e^{-rc}jf)+T_{max})$. In this paper we used branch and bound method and local search methods (BCO, IWO, GA) respectively, to comparing the results for n=5,6,7,,.,,17, which the n of jobs more than 18 jobs we used only local search method for find near optimal solution. The results show that the three algorithms have found the near optimal solutions in an appropriate time and the genetic algorithm is better comparison to other algorithms and faster time.

Keywords

The Local search algorithms (LS), Setup times, The Genetic Algorithm (GA), Bee Colony Optimization (BCO) algorithm, Invasive Weed Optimization (IWO), The Multi-Objective problems.

Introduction

Scheduling is a very important decision-making process that occurs in manufacturing systems. Scheduling problems deal with the allocation of resources to jobs over given time periods and its goal is to optimize one or more performance measures. This type of problems has been thoroughly studied since the mid-1950s [1,11]. Nowadays, scheduling problems are one of the most studied problems because they have great practical and theoretical importance. These problems have many applications in several industries (like chemical, metallurgic, and textile) and most of these problems belong to the class of NP-hard problems [8].

Many practical scheduling problems involve processing several families of related jobs on common facilities, where a set-up time is incurred whenever there is a switch from processing a job in one family to a job in another family. For example, consider a mechanical parts manufacturing environment in which jobs have to be sequenced for processing on a multi-tool machine. Whenever production changes to a new family of jobs, time is spent in retooling the machine. Based on the principles of group technology, it is conventional to schedule contiguously all jobs from the same family(Van.Wassenhove1997) [14,10].

Setup time is the time required to prepare the necessary resource (machines) to perform a task (operation or job) [2]. Setup operations include obtaining tools, positioning work in process material, return tooling, cleanupand fixtures, adjusting tools, and inspecting material [1]. In the problem under study we do not consider the group technology (GT) assumption; that is, a family of jobs is not necessarily processed as a single batch (J.B. Wang 2014) [13].

In this paper, we consider a single machine scheduling problem in which every job belongs to a specified family. Also, each job is available for processing at time zero, and has a given processing time and due date. A sequence independent set-up time, which is associated with the family of thejob to be processed next, is necessary before the first job is processed and whenever there is a switch in processing jobs from one family to jobs of another family. The objective is to find a schedule which minimizes the sum of total discount completion time and maximum tardiness.

For an arbitrary number of families, the problem is NP-hard (Bruno and Downey [5]). Various attempts by the authors to solve problem instances of a reasonable size using a branch and bound approach indicate its challenging nature. Therefore, it is worthwhile to investigate some heuristic techniques. Over the past ten years, several new local search methods have been proposed and developed: simulated annealing, tabu search and genetic algorithms. An introductory overview of these methods and some applications are given by Pirlot [15].

A problem is said to be hard (or NP-hard) if there exist no such polynomial-time algorithm to solve it. So, when we deal with an NP-hard problem, there are two ways for treating it. The first one is to target an optimal solution, while the second is by using a heuristic algorithm. The solutions which are found by the heuristic algorithm do not need to be optimal, but they are found with an acceptable time (i.e. the heuristic algorithms trade off the optimality versus the computing time) [3,9].

The heuristic algorithms can be classified into constructive algorithms and local search algorithms. local search (LS) is a simple and generally applicable stochastic local search method that iteratively applies local search to perturbations of the current search point, leading to a randomized walk in the space of local optima. To apply a LS algorithm, four components have to be defined. The first component" Generate Initial Solution" generates an initial solution. The second component" Local Search" that provides an improved solution. The third component" Perturbation" that modifies the current solution leading to some intermediate solution. The last component" Acceptance Criterion" that decides to which solution (Jarboui and Rebaï 2014) [6].

Local Search Procedures (LSPs) are optimization methods that maintain a solution, known as current solution, and explore the search space by steps within its neighborhood. They usually go from the current solution to a better close solution, which is used, in the next iteration, as current solution. This process is repeated till a stop condition is fulfilled, e.g. there is no better solution within the neighborhood of the current solution (Martinez and Lozano 2007) [7].

Since the introduction of the local search techniques in the combinatorial optimization problems, the specialists have used these techniques for solving the NP-hard problems. Using these techniques in scheduling makes an allowance for us to test problems with a large number of jobs. his computational results showed that the three (GA) algorithms were getting the stability state when n becomes larger.

Local search methods have been used in this paper, which is an algorithm the Genetic Algorithm (GA), Bee Colony Optimization (BCO) algorithm and Invasive Weed Optimization (IWO), and the details of all of them are in section (4.1).

The remainder of the paper is organized as follows. InSection 2, we present the mathematic formulation for theproblem under study. In Section 3, we explain the local search used. In Section 4, review the local methods and the system used for each. In Section 5, we show thedesign of instances andreport the computational results. Finally, in Section 6, weconclude this paper and give future directions.

Literature Review

Problem Formulation and Analysis: -

Considering a set N of n jobs ($1 \leq n \leq N$) that are divided into F different families, in our paper we consider F=2,4,6 and 8. Each family f ($1 \leq f \leq F$) consists of n_f jobs (where $\sum_{j=1}^{n} n_f = n$) and it is appropriate to label the jobs as (1,f), (2,f),….(n_f,F) which are available to be processed on the machine at time zero. The jobs from the same family may have different processing times p_{jf} which is the processing time of job j from the family f, and they can be processed one after one without requiring any setup times between them. If the machine switches from one family to the other; we would say from the family f to the family g then the setup time is required. Every job will have a due date d_{jf} which is the due date of job j from the family f. The setup time is sequence – independent (i.e. depends only on the family that is about to start), so that it is denoted by s_f. If the first job to be processed in the sequence belongs to family f, then the setup at time zero is s_{0f}. The problem is to the by the notation of Graham et al. [12] as $1/s_f/\sum_{j=1}^{n_f}\sum_{f=1}^{F}((1-e^{-rc_{jf}})+T_{max})$.

Suppose the processing order $\sigma = (\sigma(1),\ldots,\sigma(n))$, a vector $(S_{\sigma(1)},\ldots,S_{\sigma(n)})$ of corresponding setup time is easily constructed. The setup time required immediately before the processing of job $\sigma(i) = (i = 1, \dots, n)$ is given by: -

$S_{\sigma(1)}$: is the setup time of the first job (positive integer constant).

$$S_{\sigma(i)} = \begin{cases} \alpha_{fg} & if\ i > 1, \sigma(i-1) \in f and\ \sigma(i) \in g, f \neq g; f, g \in F \\ 0 & o.w \end{cases}$$

Where α_{fg} is a positive integer constant.

$$\left.\begin{array}{c} \text{Min } Z = \min \sum_{f=1}^{F} \sum_{j=1}^{n_f} (1 - e^{-rc_{jf}}) + T_{max} \\ subject\ to: \\ c_{jf} = p_{jf} + s_f \quad j = 1; f = 1, \dots, F \\ c_{jf} = c_{j-1,f} + p_{jf} \quad j = 2, \dots, n_f; f = 1, \dots, F \\ 0 < r < 1 \\ T_{max} = \max\{0, c_{jf} - d_{jf}\} \quad j = 1, \dots, n_f\ ; f = 1, \dots, F \end{array}\right\}$$

………. (P)

3- Heuristic Methods:

It is well known that the computation can be reduced by using a heuristic to act as an upper bound on the optimal solution prior to the application of branch and bound method. Since our problem $1/s_f/ \sum_{f=1}^{F} \sum_{j=1}^{n_f} ((1 - e^{-rc_{jf}}) + T_{max})$ is NP-hard and hence the existence of a polynomial time algorithm for finding an optimal solution is unlikely. Therefore, developing fast heuristic algorithms yielding near optimal solution is of great interest. For our problem we proposed three heuristic methods, the minimum value is used to provide an upper bound (UB).

3-1 The Branch and Bound method:

As an exact method, the (BAB) method is used for searching at the optimality or the area that near of it by setting a number of upper bounds as initial solutions to start from then we put the one with the minimum value, and a lower bound to reduce the searching space. the heuristic method is obtained by applying the following: -

3-1-1-The First Upper Bound (U_1): -

This upper bound is obtained by applying the shortest processing time (SPT) rule (i.e. sorting the jobs within each family in order of ($p_1 \leq p_2 \leq \cdots \leq p_n$)).

3-1-2-The Second Upper Bound (U_2): _

This upper bound is obtained by applying the earliest due date (EDD) rule (i.e. sorting the jobs within each family in order of ($d_1 \leq d_2 \leq \cdots \leq d_n$)).

3-1-3 The Third Upper bound (U_3): _

Obtain by Genetic Algorithm (GA):

Genetic algorithms (GA) are a type of search algorithms and an optimization, for returns optimal solutions to computationally difficult problems. The basic ideas for this method were developed by (Holland (1975), Goldberg and Holland (1988)) during their investigations on how to build computing machines that are capable of learning [16].

The genetic algorithm (GA) is founded on principle "survival of the fittest" for Darwin's (1859), in another meaning it's founded on the principles of genetics and natural selection. So, it is normal the concepts of Genetic Algorithm are directly derived from biological science.

3-1-3-1 Genetic algorithm: _

Step 1: Create an initial population of (50) chromosomes, we take any solutions arranged by SPT rule and the second is according EDD rule, the three is according bees' algorithm, the four-weed algorithm while the remain of them are randomly the rest of the solutions are random order.

Step 2: Evaluate the objective (fitness) function for each chromosome and select the best five chromosomes (i.e. the five chromosomes with the minimum values) to generate the new population

Step 3: Generate the new population by mating (i.e. applying crossover and mutation) each chromosome from step 2 with the whole four initial chromosomes, and every parent chromosome will produce 8 children chromosomes and add 5 chosen solutions, selected from the population so the resulting new population will consist of 50 new chromosomes.

Step 4: If the termination criterions are met, then go to step 5, else go to step 2.

Step 5: End.

In this subsection, we will introduce the main upper bound (UB) of the problem P which it is obtained by: UB=min $\{U_1, U_2, U_3\}$.

3.2 The Lower Bound (LB): The lower bound for the problem (p) is based on decomposing (p) of two sub-problems to get the a lower bound LB for problem (p), where

$$\left.\begin{array}{ll} Z_1 = Min_{\sigma(j)f}\{\sum_{f=1}^{F}\sum_{j=1}^{n_f}(1-e^{-irc_{jf}})\} & \\ \quad subject\ to: - & \\ c_{\sigma(j)f} = p_{jf} + s_f & j = 1, f = 1, \dots F \\ c_{\sigma(j)f} = c_{\sigma(j-1)f} + p_{\sigma(j)f} & j = 2, \dots, n_f ; f = 1, \dots F \\ 0 < r < 1 & \\ p_{\sigma(j)f} > 0 , d_{\sigma(j)f} > 0 & j = 1, \dots, n_f ; f = 1, \dots, F \end{array}\right\}$$

$\dots\dots\dots(p_1)$

And

$$Z_2 = Min_{\sigma(j)f}\{T_{max}\}$$

$subject\ to: -$

$c_{\sigma(j)f} \geq p_{\sigma(j)} \quad j = 1, \dots, n_f; f = 1, \dots, F$

$T_{max} = c_{jf} - d_{jf} \quad j = 1, \dots, n_f; f = 1, \dots, F$

$d_{\sigma(j)f} > 0 \quad , p_{\sigma(j)f} > 0 \quad j = 1, \dots, n_f; f = 1, \dots, F$

$\dots\dots(p_2)$

Algorithm (LB): _

Step 1: Initialize order the un-scheduling jobs by using SPT rule and adding setup time for the first job from jobs un-sequence if the not least family job of sequence.

Step 2: Calculate the value of cost function Z_1 for the problem P_1.

Step 3: Re-order the jobs by using EDD rule and adding setup time for the first job from jobs un-sequence if the not least family job of sequence.

Step 4: calculate the value of cost function Z_2 for the problem P_2.

Step 5: Sum cost the functions (i.e. total cost the problem $P_1$and the problem P_2).

Step 6: we repeat the solutions 5 times to get the solutions and compare with the old solutions.

Step 7: Go back to step 2.

4-Local Search Methods: -

In this paper, three of these methods are applied the genetic algorithm (GA), Bee Colony Optimization **(BCO)** and Invasive Weed Optimization (IWO). local search methods:

4-1 Bee Colony Optimization(BCO):many related works appeared to promote the performance of the standard **(BCO)** in the literature to meet up with challenges of recent research problems being encountered. The major advantages of **(BCO)** are simplicity, flexibility and robustness, use of fewer control parameters compared to many other search technique, ease of hybridization with other optimization algorithms, ability to handle the objective cost with stochastic nature and ease of implementation with basic mathematical and logical operations. Bees Algorithm is a new population-based search algorithm, first developed in (2005) and later by other researchers independently. The algorithm mimics the food foraging behavior Then, the algorithm conducts searches in the neighborhood of the selected sites, assigning more bees to search near to the best sites. Searches in the neighborhood of the best sites are made more detailed by recruiting more bees to follow them than the other selected bees. Together with scouting, this differential recruitment is a key operation of the bee's algorithm. The remaining bees in the population are assigned randomly around the search space scouting for new potential solutions. These steps are repeated until a stopping criterion is met. At the end of each iteration; the colony will have two parts, those that were the fittest representatives from a patch and those that have been sent out randomly. The algorithm performs a kind of neighborhood search combined with random search and can be used for both combinatorial and functional optimization [17,18].

4-1-1 The Bees Colony Optimization Algorithm (BCOA): -

Step 1: Initialize the population with 100 bees in which have random positions and velocities.

Step 2: we take any two solutions arranged by SPT rule and the second is according EDD rule, the rest of the solutions are random order.

Step 3: Evaluate the objective (the fitness) function for each bee in the swarm to get the solution.

Step 4: we choose the 20 best solutions; each solution is divided into two parts:

S1: Ordering the first half in SPT rule.

S2: Ordering the second half in EDD rule. (the division number is random between [2, n-1])

Note: (the number of working bees equals the number of solutions)

Step 4: we get 20 new solutions and we calculate the objective function for them. If it is better than the old solutions, replace them.

Step 5: we add 80 new solutions to a random number and we repeat the process.

Step 6: Go back to step 2.

4-2- The Invasive Weed Optimization (IWA): -

It is a numerical random optimization algorithm that was first proposed before (Mehrabian and Lucas) in 2006 years. Which simulates the natural behavior of weeds in colonization and finding a suitable place for growth and reproduction, an algorithm based on the intelligence of society.It includes an algorithm (IWO). Basic steps. They are interconnected and this algorithm cannot be applied to an issue unless all of these steps are applied [4]: -

1-Intialize a population: it is generated from the solutions and published on the dimension of the problem area with random locations and calculating the value of the fitness function for this community.
2-Reproduction:-plants are allowed to produce seeds (reproduction) depending on their fitness function as well as the upper and lower limit of the fitness function in the colony.
3-spatial dispersal: _ provides the characteristics of randomness and adaptation, the seeds generated are randomly distributed over a number of dimensions in the search space by random number distributed naturally at a rate of (M=0) and variable variance.
4-competitive exclusion: - if the plant does not leave any offspring, it will become extinct from existence, so there is a need for some kind of competition between plants to limit the maximum number of plants in the colony.
4-2-1 The Invasive Weed Optimization Algorithm (IWOA): -

Step 1: Initialize the population with n=100 of weed in which have random positions.
Step 2: we take any two solutions arranged by SPT rule and the second is according EDD rule, the rest of the solutions are random order.
Step 3: Evaluate the objective (the fitness) function for each weed to get the best solution. Keep the new solution.
Step 4: we choose the 20 best solutions, each solution of them alternate between two neighboring works (toggle with work next job).
Step 5: we find new solutions and calculate the objective function for them.
Step 6: we repeat the solutions 5 times to get the solutions and compare with the old solutions.
Step 7: Go back to step 2.
4-3- Genetic Algorithm (GA):
See the section (3-1-3).

Results

5-1 The Problems Instances:

The performance of the (BAB) procedure is compared on 5 problem instances, he sizes of these examples are n = [5, 17]. j where j $\in \{1, \dots, n\}$.with family f={2,4,6,8} he processing times and setup times are randomly generated integer from uniform distribution in [1,10]. While the due date $d_j$was uniformly generated in the interval [(1-T-RDD/2) TP, (1-T-RDD/2) TP] as it has been showed in the literature. Where T=$\sum_{f=1}^{F}\sum_{j=1}^{n_f} p_{jf}$ and the two parameters (TP and RDD) are said to be the tardiness factor and related range of due dates respectively, and have thefollowing values: RDD= 0.2, 0.4, 0.6, 0.8,1 and TP= 0.2, 0.4

5-2 Computational Results

In this subsection, the computational results are given in tables, each table of them gives the results. In Table 1. we put the comparison among BAB,Bees, Weed and Geneticfor n = [5, 17]. The Table 2. is contained the values and times for each Bee, Wee and GA for n = [50, 30000]. For each n there is 5 problems examples are tested. The symbols which used in the tables are:

n: The number of jobs,
F: number of family,
EX: number of examples,
BAB: The branch and bound method,
BA.: The bees Algorithm,
WA.: The weed algorithm,
GA: The genetic algorithm,
T: The time.

Note: The symbol (*) refers to the minimum value, and the symbol (#) refers to the minimum time of each (n).

5-3 The Tables of Results

In Table 1. the results of applying (BAB, Bee, Wee and GA) are showed for n= [5, 17] Jobs with the family f={2,4,6,8}. For each n and there are 5 different examples are tested. These results showed for n={5,6,7,8,9} of using (BAB and GA) are equal, while the value of using (Wee) are bigger with small differences. The execution time results showed the priority of (GA) among them (i.e. the (GA) is faster than the others).

In the table below we compare the local search methods and the time of each, as the results showed that the Genetic Algorithm (GA) is better compared to the of Bee Colony Optimization (BCO) algorithm, Invasive Weed Optimization (IWO), while the time for weeds faster.

Notes:1-The Genetic algorithm Create an initial population of (50) chromosomes and the jobs according for four algorithms (SPT-EDD-Weed-Bee); that using in their upper (3.1.3) and compare with the local search methods (4.3).

2-The Branch and Bound suggestion three upper bound and we mix among local search method to get new upper bound and we find a one lower bound for the problem with family setup time.
3-compare the local search methods with branch and bound method, for finding which one method gives near optimal solution

Table 1: comparison among BAB, Berea, Wee and GA.

N	F	Ex	BAB	T	Berea	T	Wee	T	GA	T
5	2	1	18.5606*	0.0985	18.5606*	0.4645	18.5606*	0.1519	18.5606*	0.0658#
		2	24.3825*	0.0228#	24.3825*	0.4299	24.3825*	0.1459	24.3825*	0.0540
		3	2.9510*	0.0128#	12.9510*	0.4290	12.9510*	0.1439	12.9510*	0.0519
		4	24.5334*	0.0047#	24.5334*	0.3990	24.5334*	0.1431	24.5334*	0.0516
		5	21.9118*	0.0113#	21.9118*	0.4156	21.9118*	0.1450	21.9118*	0.0521
	4	1	42.1817*	0.0507#	42.1817*	0.3826	42.1817*	0.1365	42.1817*	0.0535
		2	27.0042*	0.0087#	27.0042*	0.4608	27.0042*	0.1590	27.0042*	0.0558
		3	26.7009*	0.0096#	26.7009*	0.4044	26.7009*	0.1403	26.7009*	0.0500
		4	13.2115*	0.0032#	13.2115*	0.4510	13.2115*	0.1495	13.2115*	0.0536
		5	28.9000*	0.0072#	28.9000*	0.4142	28.9000*	0.1602	28.9000*	0.0523
	6	1	35.1482*	0.0093#	35.1482*	0.4087	35.1482*	0.1435	35.1482*	0.0521
		2	36.2341*	0.0117#	36.2341*	0.4005	36.2341*	0.1312	36.2341*	0.0530
		3	18.8498*	0.0033#	18.8498*	0.4124	18.8498*	0.1349	18.8498*	0.0518
		4	42.6685*	0.0094#	42.6685*	0.3859	42.6685*	0.1487	42.6685*	0.0613
		5	25.8115*	0.0054#	25.8115*	0.4265	25.8115*	0.1501	25.8115*	0.0541
	8	1	26.9330*	0.0084#	26.9330*	0.4362	26.9330*	0.1494	26.9330*	0.0494
		2	30.6073*	0.0069#	30.6073*	0.4638	30.6073*	0.1467	30.6073*	0.0526
		3	31.8215*	0.0066#	31.8215*	0.4436	31.8215*	0.1382	31.8215*	0.0543
		4	42.0722*	0.0082#	42.0722*	0.4072	42.0722*	0.1358	42.0722*	0.0505
		5	39.1439*	0.0096#	39.1439*	0.3876	39.1439*	0.1373	39.1439*	0.0535
6	2	1	21.1585*	0.0077#	21.1585*	0.4162	21.1990	0.1467	21.1585*	0.0605
		2	9.1939*	0.0031#	9.1939*	0.4144	9.1939*	0.1410	9.1939*	0.0530
		3	17.2970*	0.0038#	17.2970*	0.4007	17.2970*	0.1405	17.2970*	0.0530
		4	27.4871*	0.0042#	27.4871*	0.3956	27.5324	0.1355	27.4871*	0.0531
		5	25.2247*	0.0045#	25.2247*	0.3972	25.2674	0.14096	25.2247*	0.0529
	4	1	32.5588*	0.0153#	32.5588*	0.3892	32.7121	0.1372	32.5588*	0.05272
		2	38.5210*	0.0165#	38.5210*	0.3870	38.5210*	0.1397	38.5210*	0.0523
		3	44.9892*	0.0197#	44.9892*	0.3883	44.9892*	0.1387	44.9892*	0.0523
		4	28.9647*	0.0081#	28.9647*	0.4057	28.9647*	0.1374	28.9647*	0.0524
		5	30.7474*	0.0182#	30.7474*	0.3883	30.7774*	0.1359	30.7474*	0.0525
		1	27.6416*	0.0119#	27.6416*	0.41188	27.7153	0.1357	27.6416*	0.0525
		2	29.3259*	0.0101#	29.3259*	0.3969	29.3259*	0.1342	29.3259*	0.0523

	6	3	37.7544*	0.0147#	37.7544*	0.3994	37.7544*	0.1340	37.7544*	0.0552
		4	43.5263*	0.0282#	43.5263*	0.4109	43.5263*	0.1346	43.5263*	0.0527
		5	36.9614*	0.0268#	36.9614*	0.3916	36.9614*	0.1391	36.9614*	0.0524
	8	1	44.6377*	0.0316#	44.6377*	0.4087	44.6377*	0.1358	44.6377*	0.0526
		2	47.8967*	0.0416#	47.8967*	0.3889	47.8967*	0.1351	47.8967*	0.0578
		3	40.8785*	0.0206#	40.8785*	0.4098	40.8785*	0.1385	40.8785*	0.0576
		4	43.6828*	0.0431#	43.6828*	0.3899	43.6828*	0.1595	43.6828*	0.0567
		5	33.4795*	0.0388#	33.4795*	0.4094	33.4795*	0.1633	33.4795*	0.0565
7	2	1	25.9323*	0.0139#	25.9323*	0.4291	25.9545	0.1493	25.9323*	0.0547
		2	24.9601*	0.0153#	24.9601*	0.4057	25.0073	0.1424	24.9601*	0.0541
		3	41.2338*	0.0120#	41.2338*	0.4055	41.9309	0.1418	41.2338*	0.0551
		4	44.1869*	0.0204#	44.1869*	0.4085	44.1869*	0.1410	44.1869*	0.0545
		5	16.2182*	0.0159#	16.2182*	0.4177	16.2468	0.1432	16.2182*	0.0600
	4	1	39.9456*	0.0256#	39.9456*	0.4060	39.9701	0.1548	39.9456*	0.0546
		2	47.6593*	0.0511#	47.6593*	0.4175	47.6918	0.1528	47.6593*	0.0544
		3	44.1861*	0.0429#	44.1861*	0.4165	45.1709	0.1514	44.1861*	0.0542
		4	34.5836*	0.0156#	34.5836*	0.4346	35.2004	0.1501	34.5836*	0.0544
		5	34.8067*	0.0634#	34.8067*	0.4238	35.1206	0.1509	34.8067*	0.0595
	6	1	51.7444*	0.1412	51.7444*	0.4170	51.9163	0.1487	51.7444*	0.0554#
		2	46.7451*	0.2027	46.7451*	0.4022	46.8926	0.1432	46.7451*	0.0546#
		3	47.1403*	0.0798	47.1403*	0.4047	47.1403*	0.1410	47.1403*	0.0552#
		4	43.3529*	0.1657	43.3529*	0.4066	43.3847	0.1428	43.3529*	0.0551#
		5	44.4276*	0.1094	44.4276*	0.4258	44.4276*	0.1425	44.4276*	0.0546#
	8	1	54.0482*	0.1786	54.0482*	0.4095	54.0482*	0.1415	54.0482*	0.0545#
		2	74.0625*	0.3086	74.0625*	0.4035	74.0698	0.1404	74.0625*	0.0547#
		3	56.9049*	0.2873	56.9049*	0.4030	56.9271	0.1421	56.9049*	0.0541#
		4	45.6390*	0.1526	45.6390*	0.4126	45.6873	0.1462	45.6390*	0.0544#
		5	67.9602*	0.3101	67.9602*	0.4024	67.9803	0.1419	67.9602*	0.0543#
	2	1	39.3047*	0.1026	39.3128	0.4179	39.4823	0.1496	39.3047*	0.0563#
		2	32.9801*	0.0080#	32.9801*	0.4153	35.1115	0.1471	32.9801*	0.0562
		3	30.1949*	0.0173#	30.1949*	0.4155	32.1947	0.1478	30.1949*	0.0564
		4	32.4527*	0.0999	32.4914	0.4161	32.5564	0.4161	32.4527*	0.0569#
		5	23.5777*	0.0260#	23.5777*	0.4185	23.9652	0.1494	23.5777*	0.0570
		1	61.9362*	0.1261	61.9414	0.4144	62.1407	0.1522	61.9362*	0.0563#
		2	43.1333*	0.1246	43.2867	0.4132	43.1982	0.1463	43.1333*	0.0566#

8	4	3	52.5488*	0.0495#	52.6209	0.4141	53.5642	0.1471	52.5488*	0.0569
		4	38.1544*	0.2606	38.1544*	0.4087	38.2289	0.1467	38.2725	0.0568#
		5	44.4976*	0.0477#	44.4976*	0.4142	44.4976*	0.1503	44.4976*	0.0568#
	6	1	58.3657*	0.5651	58.3903	0.4251	58.3821	0.1472	58.3657*	0.0588#
		2	60.3769*	0.6345	60.3769	0.4227	60.6756	0.1477	60.3769*	0.0563#
		3	45.6313	0.5575	45.6368	0.5796	45.6801	0.1909	45.6412	0.0727#
		4	48.8810*	0.5503	48.9080	0.4586	49.3592	0.1542	48.8810*	0.0573#
		5	61.0708*	1.2641	61.0708*	0.4132	61.1317	0.1478	61.0708*	0.0572#
	8	1	76.0324*	0.9324	76.0324*	0.4194	76.0324*	0.1518	76.0324*	0.0561#
		2	79.0159*	1.6005	79.0177	0.4225	79.0236	0.1441	79.0159*	0.0565#
		3	63.2340*	1.1738	63.2340	0.4155	63.2920	0.1570	63.2340*	0.0562#
		4	53.3537*	0.8522	53.3563	0.4133	53.4215	0.1464	53.3537*	0.0559#
		5	63.7790*	0.8429	63.7803	0.4159	63.8197	0.1455	63.7790*	0.0561#
9	2	1	48.2202*	0.0446#	48.2202*	0.4286	48.9988	0.1574	49.6757	0.0577
		2	43.9143*	0.0257#	43.9143*	0.4200	44.5653	0.1517	43.9143*	0.0590
		3	43.5595*	0.0261#	43.5752	0.4292	44.6261	0.1523	43.5595*	0.0576
		4	39.0451*	0.0163#	39.0451*	0.4244	39.0651*	0.1518	39.0451*	0.0606
		5	42.4435*	0.2566	42.4509	0.4284	42.8073	0.1516	42.4435*	0.0585#
	4	1	57.5230*	0.4580	58.0085	0.4424	60.0511	0.1564	57.5230*	0.0599#
		2	39.1369*	0.4868	39.8540	0.4242	39.3937	0.1543	39.1369*	0.0610#
		3	33.7555*	0.1202	33.7555*	0.4321	36.1272	0.1520	33.7555*	0.0583#
		4	60.6951*	0.6293	61.0948	0.4185	63.7713	0.1564	60.6951*	0.0583#
		5	58.2350*	1.2132	58.2985	0.4217	59.4496	0.1536	58.2350*	0.0581#
	6	1	64.7264*	3.6172	64.9804	0.4267	66.0411	0.1599	64.7264*	0.0579#
		2	63.2888*	5.2996	63.3447	0.4413	63.7373	0.1615	63.2888*	0.0602#
		3	71.0934*	6.5950	71.1325	0.4442	71.3152	0.1650	71.0934*	0.0578#
		4	51.4014*	1.4572	51.8328	0.4488	51.9634	0.1604	51.4014*	0.0588#
		5	58.4470*	7.8996	58.4470*	0.4314	58.6750	0.1569	58.4470*	0.0580#
	8	1	73.1080*	17.9807	73.1113	0.4200	73.1266	0.1561	73.1080*	0.0576#
		2	82.3050*	8.0750	82.3053	0.4454	82.5769	0.1610	82.3050*	0.0578#
		3	51.3196*	4.0594	51.3689	0.4458	52.8253	0.1605	51.3196*	0.0578#
		4	62.9748*	8.2472	63.0120	0.4492	63.2946	0.1637	62.9748*	0.0579#
		5	61.7406*	4.8405	61.7517	0.4397	62.0277	0.1599	61.7406*	0.0582#
		1	36.4785*	0.0648	36.5218	0.4359	37.1572	0.1591	36.4785*	0.0596#
		2	49.1367*	0.0290#	49.1367*	0.4372	49.7424	0.1609	49.1367*	0.0606

10	2	3	41.2873*	0.1990	41.3021	0.4327	41.3667	0.1568	41.2873*	0.0608#
		4	60.1788*	0.2678	60.1788*	0.4376	60.3732	0.1580	60.4483	0.0621#
		5	60.4654*	0.1077	60.4674*	0.4343	61.5736	0.1589	60.4654*	0.0622#
	4	1	65.3664*	3.7450	65.4118	0.4325	66.5497	0.1587	65.3664*	0.0594#
		2	43.2773	1.6108	43.8183	0.4394	43.8997	0.1613	43.7637	0.0596#
		3	57.3485	0.8363	57.8054	0.4353	59.7358	0.1582	59.2147	0.0596#
		4	51.4936*	0.5623	51.6626	0.4316	54.6468	0.1628	51.4936*	0.0600#
		5	54.0759*	2.1895	54.3668	0.4315	56.8210	0.1586	54.0759*	0.0601#
	6	1	70.3132*	2.4590	70.3522	0.4418	71.5449	0.1589	70.3132*	0.0596#
		2	69.8474*	1.5037	70.9208	0.4414	72.4866	0.1574	69.8474*	0.0592#
		3	78.3835*	42.8273	78.4125	0.4329	78.6769	0.1585	78.3835*	0.0595#
		4	45.3040*	0.8183	45.3040*	0.4642	49.9062	0.1757	45.3040*	0.0666#
		5	54.5406*	6.4310	54.5602	0.4718	56.3261	0.1778	54.5406*	0.0663#
	8	1	87.1547*	116.2090	87.4453	0.4580	87.3657	0.1743	87.1547*	0.0655#
		2	79.0112*	52.9967	79.0275	0.4684	80.2198	0.1782	79.0112*	0.0650#
		3	102.1067*	114.3312	102.115	0.4648	102.1646	0.1732	102.1067*	0.0656#
		4	70.2734*	34.6518	70.2767	0.4619	70.6254	0.1711	70.2734*	0.0657#
		5	83.6773	64.7658	83.7237	0.4664	83.9163	0.1754	83.8038	0.0643#
11	2	1	72.7247*	2.0814	72.7247*	0.4449	72.7247*	0.1646	72.7247*	0.0668#
		2	39.3959	3.9015	39.6799	0.4510	43.2460	0.1673	39.6541	0.0644#
		3	38.3651	0.2382	38.3868	0.4556	39.6968	0.1656	39.3033	0.0640#
		4	49.4398*	0.1749	49.4689	0.4423	52.8083	0.1637	49.4398*	0.0631#
		5	48.0935*	1.3023	48.1028	0.4509	49.4937	0.1637	48.0935*	0.0648#
	4	1	50.6531	6.9562	51.5934	0.4479	55.0821	0.1636	50.7143	0.0638#
		2	67.0284	2.6572	67.1244	0.4417	74.0042	0.1652	67.9767	0.0630#
		3	64.2894	2.7612	65.4815	0.4417	65.6325	0.1655	64.4063	0.0632#
		4	58.2023	3.7702	60.1895	0.4470	64.1659	0.1649	58.3479	0.0638#
		5	37.2267*	3.2627	39.3023	0.4473	39.6844	0.1647	37.2267*	0.0645#
	6	1	75.5646	61.5642	75.8901	0.4474	76.6730	0.1646	75.6703	0.0635#
		2	75.6740	15.4453	79.4626	0.4508	78.7191	0.1649	75.8112	0.0637#
		3	72.2324	60.3902	72.2589	0.4415	73.2640	0.1647	74.3380	0.0635#
		4	51.0511	7.4209	51.4128	0.4484	54.3781	0.1694	51.0688	0.0649#
		5	59.3391	10.2895	60.0727	0.4431	65.2751	0.1682	59.4919	0.0636#
		1	97.4213	790.5983	97.8363	0.4449	99.6661	0.1674	97.4859	0.0643#
		2	89.9661*	221.6080	90.0132	0.4446	92.5163	0.1682	89.9661*	0.0645#

	8	3	70.7659	8.7755	72.9767	0.4532	75.6705	0.1655	72.13662	0.0646#
		4	59.2787	54.4932	59.4976	0.4530	60.9492	0.1660	60.1251	0.0647#
		5	88.1852	629.5126	88.3545	0.4533	88.5335	0.1660	88.1855	0.0638#
12	2	1	52.9215	0.6428	53.9726	0.4905	56.6219	0.1918	53.3028	0.0714#
		2	53.4536*	27.4181	53.4925	0.4996	56.4311	0.1938	53.4536*	0.0663#
		3	73.0304*	0.5517	73.0350	0.4529	74.5681	0.1672	73.0304*	0.0647#
		4	67.3891*	1.9310	67.3891	0.4558	71.0137	0.1692	67.3891*	0.0653#
		5	64.2999*	16.9135	66.0649	0.4582	67.5596	0.1672	64.2999*	0.0650#
	4	1	83.7581	8.4155	86.9135	0.4550	90.1659	0.1732	83.8157	0.0684#
		2	68.5505	21.4935	68.6973	0.4588	74.6238	0.1701	68.8611	0.0655#
		3	53.4514	2.6660	59.1933	0.4617	58.3252	0.1716	54.1842	0.0650#
		4	77.7415*	12.1894	80.9297	0.4546	82.1134	0.1717	77.7415*	0.0687#
		5	57.7770	49.0465	61.1666	0.4552	58.5655	0.1688	58.5310	0.0654#
	6	1	75.8596	66.1950	79.2566	0.4560	84.3469	0.1708	78.05211	0.0664#
		2	96.0855	96.9198	97.5211	0.4516	105.6269	0.1703	96.5508	0.0651#
		3	81.6870	143.1084	82.2626	0.4562	82.1328	0.1705	81.7411	0.0649#
		4	80.5243	107.3939	83.5676	0.4569	88.8981	0.1704	81.2309	0.0647#
		5	83.6811	513.1630	84.8448	0.4549	89.2920	0.1700	83.7092	0.0677#
	8	1	/	1800.0002	105.5285	0.4918	105.8505	0.1746	105.1296	0.0698#
		2	/	1800.0002	103.6437	0.4614	104.5907	0.1705	104.0560	0.0657#
		3	81.8674*	928.4289	82.1228	0.4655	83.3169	0.1701	81.8674*	0.0655#
		4	74.8196	679.1056	76.3064	0.4569	82.5659	0.1721	75.2834	0.0686#
		5	/	1800.0001	111.2247	0.4480	113.4008	0.1720	110.1802	0.0651#
13	2	1	61.3626	0.7790	61.3721	0.4702	64.4809	0.1768	64.3115	0.0677#
		2	80.9849*	1.2446	80.9853	0.4786	83.9871	0.1849	80.9849*	0.0686#
		3	64.5674	4.3214	64.5726	0.4663	68.1694	0.1801	64.9331	0.0661#
		4	72.0750	132.4878	73.6092	0.4683	76.5185	0.1760	73.0994	0.0675#
		5	78.0532	1.03884	80.9022	0.4653	81.5961	0.1749	80.6649	0.0669#
	4	1	69.5527*	27.5206	73.8889	0.4616	76.6243	0.1758	69.5527*	0.0671#
		2	66.0666	33.5796	69.0807	0.4634	72.5096	0.1787	66.1312	0.0684#
		3	80.2413	29.8601	81.4636	0.4629	83.8502	0.1766	80.3991	0.0676#
		4	73.9158	271.8963	76.1911	0.4699	79.7818	0.1831	73.9556	0.0681#
		5	71.9607	100.8326	76.4127	0.4717	78.2636	0.1771	72.9570	0.0679#
		1	75.3635	742.1095	81.8796	0.4699	85.0463	0.1760	75.5487	0.0672#
		2	77.6531	905.5780	80.3428	0.4622	88.2237	0.1791	78.1220	0.0668#

	6	3	106.6035	1100.5083	108.1604	0.4637	114.4999	0.1747	110.2544	0.0670#
		4	82.5116	367.0831	86.7071	0.4684	88.8364	0.1774	83.2591	0.0672#
		5	43.0417	28.8051	48.4491	0.4647	49.8256	0.1776	43.7288	0.0675#
14	2	1	43.4003*	11.0523	45.8731	0.4763	50.2732	0.1827	43.4003*	0.0704#
		2	56.7608	146.1543	59.7070	0.4855	62.9436	0.1828	59.6941	0.0698#
		3	61.8356*	577.6754	61.9113	0.4761	66.7987	0.1813	61.8356*	0.0698#
		4	43.4232*	0.3046	46.5233	0.4806	50.7521	0.1852	43.4232*	0.0702#
		5	72.4318	176.8153	72.8423	0.4743	78.9924	0.1918	72.7792	0.0696#
	4	1	79.1435*	147.6929	85.1706	0.4750	88.5661	0.1902	79.1435*	0.0733#
		2	92.7653	92.7637	98.4035	0.4721	99.0052	0.1896	93.7483	0.0691#
		3	48.6522	233.6976	55.7616	0.4781	60.8046	0.1905	50.2635	0.0702#
		4	57.1401*	261.1641	58.7544	0.4783	70.9572	0.1836	57.1401*	0.0704#
		5	67.3696	137.2345	72.6525	0.4833	78.8918	0.1823	69.5732	0.0691#
	6	1	/	1800.0001	80.8135	0.4785	84.9408	0.1821	75.8199	0.0698#
		2	68.5429	569.4346	76.4052	0.4786	84.4298	0.1818	69.4252	0.0699#
		3	/	1800.0004	104.6150	0.4733	107.4962	0.1816	99.6035	0.0691#
		4	68.7711	891.9382	76.9306	0.4764	73.9319	0.1834	69.1199	0.0702#
		5	/	1800.0001	82.1216	0.4921	92.2784	0.1851	80.9134	0.0699#
15	2	1	54.2869	684.8232	59.4226	0.5327	60.1225	0.2113	57.0941	0.0779#
		2	60.5660	3.9229	63.0954	0.5303	66.2781	0.2079	63.3588	0.0757#
		3	45.1227	22.8234	45.9052	0.4904	53.0254	0.1944	45.9358	0.0714#
		4	63.0105	1196.1326	63.0150	0.5214	64.8667	0.1876	63.1903	0.0714#
		5	56.5249	60.8069	62.0387	0.4905	65.1173	0.1864	59.2991	0.0736#
	4	1	87.4431	760.6983	93.1169	0.4907	96.9351	0.1901	89.8828	0.0710#
		2	66.1123	473.5506	92.3165	0.4858	94.5200	0.1882	81.1576	0.0737#
		3	63.6930	184.9003	68.4550	0.4826	74.0407	0.1885	63.8109	0.0717#
		4	58.4665	90.8379	69.0210	0.4905	74.6061	0.1963	60.7093	0.0721#
		5	67.58889	1603.3751	74.2826	0.4873	79.4201	0.1874	69.3061	0.0740#
16	2	1	53.0390	43.7779	58.7303	0.5008	63.7399	0.1934	55.7126	0.0774#
		2	69.7273	39.4724	72.6160	0.5066	75.2888	0.1966	72.6159	0.0772#
		3	118.3106	14.9773	119.4529	0.4916	120.1649	0.1939	121.1975	0.0755#
		4	70.4269	39.5166	76.1073	0.5044	82.2543	0.1928	71.4085	0.0759#
		5	42.5575	14.7345	44.5656	0.5030	49.1660	0.1976	42.6720	0.0761#
		1	67.0088	1356.1333	78.1382	0.4977	84.4185	0.1954	71.1079	0.0764#

	4	2	75.8311	1545.4118	84.3829	0.5124	92.6576	0.2013	76.3043	0.0758#
		3	61.5876	240.5099	69.2331	0.5199	78.8534	0.2033	64.8543	0.0790#
		4	104.3130	926.0965	112.2197	0.5062	117.1087	0.1992	104.3130	0.0753#
		5	74.4973	1431.6475	83.4189	0.4932	87.7998	0.1937	74.8304	0.0760#
17	2	1	68.7261	7.9549	73.3908	0.5383	78.1459	0.2184	68.7268	0.0814#
		2	/	1800.0002	70.3767	0.5370	70.1798	0.2183	64.6948	0.0786#
		3	/	1800.0003	61.3333	0.5343	67.2152	0.2148	59.1816	0.0777#
		4	56.9582	1013.1670	60.7818	0.5383	64.5790	0.2111	57.8007	0.0785#
		5	107.5272	936.4970	109.5848	0.5409	115.1226	0.2110	108.1564	0.0786#

Note: The Branch and Bound (BAB) used the time until hour half (i.e. 1800 second) to get on the optimal solution. The number n=12 in f=8 there is an example 1,2 and 5 exceeding required time, as well as in n=14 in f=6 there is an example 1,3,5 and n=17 in f=2 there is an example 2 and 3.

In Table 2. For each (n) there are (5) problems examples for testing. **The Table 2**. Begins with n = 50(50)100, 100(200)1000, 1000(5000)10000,10000(10000)300000. The results showed that valueof (GA) are better than the (Bee and Wee),The GA problem is solved for (n=1000) jobs.

Table 2: comparison among (Bee), (Wee) and (GA).

N	F	Ex	Bee	T	Wee	T	GA	T
50	2	1	285.6248	0.8565	298.1992	0.4112	272.2037*	0.2208#
		2	240.3685	0.8274	259.6488	0.3932	223.8489*	0.1982#
		3	312.1920	0.8177	321.3399	0.4043	297.4896*	0.2038#
		4	248.1713	0.8500	268.6176	0.3876	237.0461*	0.2004#
		5	243.3204	0.8162	254.3407	0.38257	222.6653*	0.1974#
	4	1	301.3078	0.8250	331.6300	0.40800	264.4872*	0.1981#
		2	364.3056	0.8356	372.4126	0.39612	326.2671*	0.2043#
		3	289.1727	0.8460	296.4667	0.39363	239.5360*	0.1980#
		4	394.8440	0.8315	397.2111	0.40125	350.6242*	0.1984#
		5	349.5986	0.8301	365.2232	0.39963	317.0047*	0.2086#
	6	1	238.9409	0.8301	266.4017	0.3977	191.2833*	0.2008#
		2	397.7028	0.8365	409.0373	0.40834	340.7898*	0.2001#
		3	315.5189	0.8231	333.2989	0.39548	257.1961*	0.1976#
		4	326.1823	0.8370	350.7216	0.39065	276.2980*	0.1983#
		5	295.7448	0.8224	345.8219	0.39414	243.0984*	0.1982#
	8	1	372.5185	0.8210	406.2621	0.38855	295.6395*	0.2060#
		2	317.3019	0.8557	341.4149	0.3908	248.2926*	0.2000#
		3	346.4201	0.8305	385.3443	0.3972	265.6467*	0.1990#
		4	326.8899	0.9075	345.3496	0.4673	231.4370*	0.2301#

		5	442.5610	0.9220	477.4072	0.4584	361.9610*	0.2443#
100	2	1	386.3432	1.2719	394.8079	0.6855	357.1158*	0.5284#
		2	426.1387	1.2739	394.8079	0.6753	396.0528*	0.5256#
		3	458.2648	1.2873	483.1500	0.6822	424.5937*	0.5383#
		4	583.4622	1.2956	614.8734	0.6799	567.0114*	0.5269#
		5	414.2429	1.2700	430.1341	0.6684	389.0664*	0.5375#
	4	1	596.0188	1.2809	639.7345	0.6725	526.1996*	0.5262#
		2	583.9119	1.2729	623.2947	0.6811	504.4669*	0.5244#
		3	634.5438	1.2758	668.8486	0.6754	580.7406*	0.5274#
		4	693.6834	1.2776	743.1192	0.6750	639.0269*	0.5255#
		5	391.0733	1.2798	402.9428	0.6768	333.8653*	0.5263#
	6	1	669.7862	1.2845	693.9833	0.6802	556.6213*	0.5285#
		2	529.5529	1.2868	539.8521	0.6754	462.0204*	0.5751#
		3	514.9868	1.3332	549.0606	0.7015	412.3060*	0.5528#
		4	845.0091	1.3447	862.2468	0.7006	720.1364*	0.6206#
		5	680.7628	1.3302	726.3723	0.6930	577.9203*	0.5522#
	8	1	914.5381	1.3381	927.3203	0.6913	771.1989*	0.5322#
		2	845.0189	1.28244	860.3668	0.6797	675.0984*	0.5330#
		3	601.5429	1.4651	636.5682	0.7592	491.3633*	0.5330#
		4	917.2911	1.3083	941.2415	0.6881	761.2316*	0.5465#
		5	785.2267	1.2915	836.5709	0.68762	619.8794*	0.5406#
200	2	1	1327.5916	2.19284	1368.6852	1.2289#	1251.6473*	1.7786
		2	687.6957	2.5487	723.8622	1.3206#	641.5186*	1.7300
		3	654.4890	2.1812	673.2885	1.2539#	618.8961*	1.8871
		4	1340.4773	2.5169	1375.8953	1.4682#	1273.1364*	1.8023
		5	1416.8471	2.2794	1422.9917	1.2997#	1354.2200*	1.7414
	4	1	1591.5161	2.2250	1650.5485	1.2514#	1454.8950*	1.8214
		2	1610.0767	2.2566	1631.6363	1.3742#	1459.1003*	1.8296
		3	1239.0205	2.2426	1282.6812	1.2508#	1107.4963*	2.1096
		4	1276.1287	2.2388	1343.0005	1.2400#	1156.9052*	1.7327
		5	729.0351	2.2032	780.9464	1.2447#	657.0412*	1.9284
	6	1	1368.0649	2.2659	1432.7518	1.4543#	1192.7120*	2.1037
		2	1741.6851	2.5498	1761.8686	1.4822#	1532.0623*	1.7436
		3	1265.6253	2.2530	1288.3112	1.4439#	1112.2003*	2.0520
		4	1600.3860	2.5353	1652.7411	1.4413#	1406.1970*	1.7145

		5	1633.2431	2.2055	1674.6119	1.2385#	1415.2450*	1.7070
	8	1	1355.5856	2.28166	1402.5382	1.3729#	1136.4840*	2.1607
		2	1309.4227	2.2927	1388.4386	1.2509#	1128.3606*	1.7230
		3	1765.9527	2.2116	1813.4903	1.2659#	1466.7952*	2.0079
		4	1676.7937	2.22466	1729.5611	1.2568#	1440.5447*	1.7512
		5	1608.7772	2.2926	1659.4327	1.3083#	1359.5298*	1.7536
400	2	1	2195.6685	4.0139	2259.2252	2.3711#	2120.8361*	6.1749
		2	1739.7771	4.0880	1755.3538	2.7577#	1642.6810*	6.1574
		3	2369.1630	3.9531	2395.6406	2.3861#	2249.7581*	6.2068
		4	1580.8979	3.9498	1641.9552	2.3449#	1513.2394*	7.3684
		5	2385.6631	3.9381	2435.9320	2.3059#	2310.7403*	6.1707
	4	1	2996.6243	4.0971	3006.2077	2.3524#	2781.5826*	6.3048
		2	1564.0626	4.0221	1598.0797	2.3975#	1424.6881*	7.5178
		3	2165.3269	4.1647	2229.8195	2.3734#	2007.4876*	6.6517
		4	2385.1171	4.1603	2461.0160	2.3881#	2201.9558*	6.4602
		5	2392.7262	4.0483	2489.5034	2.3525#	2227.1954*	7.3561
	6	1	2135.0390	4.0648	2240.4635	2.4061#	1988.3556*	6.1786
		2	2134.3008	4.0009	2255.1325	2.3890#	1995.4497*	6.2222
		3	2843.6033	4.0234	2906.1424	2.3523#	2577.1282*	6.2539
		4	2125.6397	3.9957	2185.2517	2.4043#	1886.6348*	6.1329
		5	2989.2613	4.0131	3061.7826	2.3732#	2740.5660*	6.2049
	8	1	3057.3168	4.0378	3217.5733	2.3787#	2773.3328*	7.0626
		2	3377.2694	4.0482	3420.3701	2.3701#	3046.5444*	6.2913
		3	3398.0216	4.2599	3473.2082	2.4409#	3078.9608*	7.0451
		4	2544.7661	4.6764	2623.2017	2.4087#	2282.1270*	6.2523
		5	3217.8187	4.1004	3285.1507	2.3786#	2882.1306*	6.2790
	2	1	4252.3479	6.0784	4281.2117	3.6691#	4118.3783*	13.7173
		2	2301.2043	5.8531	2334.6126	3.5140#	2211.6701*	13.5917
		3	2635.2967	5.8084	2731.1793	3.4734#	2539.1709*	13.4707
		4	3066.2751	5.8236	3132.9207	3.4627#	2965.4423*	13.6140
		5	2879.3403	5.8106	2906.3613	3.4550#	2754.6277*	13.5191
	4	1	2962.4374	5.8938	3028.0807	3.5294#	2802.8286*	13.5973
		2	3927.8350	5.8687	4012.9062	3.5060#	3740.7258*	13.5885
		3	3270.1021	5.8908	3357.7121	3.5322#	3078.0595*	14.2181
		4	2991.7035	5.8906	3059.6864	3.5117#	2821.0263*	13.5146

600		5	3280.6026	5.8509	3335.9996	3.4983#	3084.5506*	13.4597
	6	1	4287.47281	5.9164	4421.1963	3.5117#	4004.2798*	14.3896
		2	4544.3984	6.6459	4642.1190	3.4977#	4319.8018*	13.5155
		3	4566.0251	5.8589	4686.7436	3.4965#	4320.8003*	13.4984
		4	3230.3381	5.8457	3310.7250	3.5223#	3006.6786*	13.4634
		5	4219.6661	5.8741	4317.8279	3.5226#	3969.4124*	13.5106
	8	1	4765.9096	6.5090	4906.1872	4.1568#	4355.3492*	13.5164
		2	5203.0122	5.8811	5314.1876	3.5122#	4788.9746*	13.5795
		3	5117.7782	5.8868	5239.3237	3.5119#	4822.3203*	13.5992
		4	3863.6424	5.8906	3985.1742	3.5282#	3563.2912*	13.4548
		5	5404.9077	5.8962	5471.8401	4.0726#	5043.2599*	14.6838
800	2	1	3100.7373	7.6117	3210.0628	4.5977#	3014.2857*	23.5818
		2	5345.4960	7.9392	5423.5352	5.3473#	5193.8695*	26.3841
		3	3927.9169	8.9849	3970.7933	5.0129#	3803.8714*	23.6275
		4	3564.8100	7.6074	3657.5789	4.5899#	3467.6256*	23.7735
		5	3534.6190	7.6330	3578.6935	4.5877#	3422.5652*	23.6983
	4	1	4909.0328	7.70563	4973.0731	4.6205#	4668.4293*	24.5679
		2	3530.7122	7.6926	3586.5359	4.7179#	3352.8100*	23.6335
		3	6213.3680	7.7575	6343.0815	4.6017#	5988.2055*	23.7965
		4	5720.3123	7.7623	5811.5061	4.6178#	5455.7762*	23.9152
		5	4429.0194	7.6943	4490.0879	5.2070#	4230.2790*	24.3050
	6	1	5724.4599	7.8281	5852.9806	4.6478#	5389.4577*	25.8881
		2	6605.2904	8.6377	6739.6512	4.6181#	6285.2643*	23.6771
		3	6113.4891	7.7597	6188.5072	5.2654#	5799.2641*	23.8709
		4	6659.3616	7.7649	6778.5977	4.6425#	6295.6038*	23.6153
		5	4366.3806	7.7192	4469.3285	4.6598#	4138.9438*	23.6831
	8	1	7707.6245	7.9231	7912.2871	4.6898#	7255.3120*	23.9906
		2	6004.4922	7.7471	6212.5334	4.6723#	5791.2960*	23.8292
		3	6496.5874	8.2202	6597.8437	4.6862#	6150.7923*	23.7539
		4	7781.3999	7.8884	7835.7040	4.6721#	7370.1525*	24.9208
		5	6552.4229	8.1009	6649.2362	4.6599#	6117.2147*	23.6914
	2	1	2821.5301	9.3670	2911.9788	5.7506#	2797.4159*	41.5944
		2	4954.6026	10.5148	5031.5498	6.6757#	4832.7847*	36.3648
		3	4534.5280	9.3790	4583.5442	5.6663#	4410.7260*	36.5366
		4	5522.2252	10.7781	5561.3809	5.5990#	5380.3775*	36.8804

1000		5	6727.8836	9.3992	6806.7916	5.6353#	6577.8880*	37.2380
	4	1	6741.0673	9.6060	6771.8019	5.7863#	6407.8452*	36.2330
		2	6773.8376	9.5211	6880.5206	5.7195#	6543.8233*	36.3917
		3	7228.4064	9.5599	7317.6855	5.7090#	6964.2380*	36.2751
		4	6612.3572	9.5713	6725.8547	5.7365#	6381.3082*	36.2634
		5	6183.5253	9.6043	6208.0589	5.7644#	5890.8219*	36.3950
	6	1	8871.3002	9.5971	8981.9105	5.7309#	8486.4900*	36.6881
		2	5517.8734	9.5594	5673.8278	5.7835#	5317.5735*	36.4546
		3	6550.9363	9.5596	6623.1505	5.7709#	6306.7807*	38.8866
		4	7153.0184	9.5917	7250.0480	5.7215#	6895.4981*	37.5142
		5	7179.2647	9.5343	7346.0020	5.7139#	6876.9513*	36.6185
	8	1	8155.044	10.3357	8355.6422	6.3400#	7679.5613*	38.4897
		2	9727.4854	9.6016	9847.0274	5.7327#	9313.0376*	36.6472
		3	7522.2758	9.4891	7639.7937	5.7397#	7199.7937*	38.6669
		4	10298.9011	9.9122	10348.0913	6.0446#	9857.1700*	36.7722
		5	8732.8996	9.5317	8861.0365	5.7069#	8316.3536*	36.6553
5000	2	1	30463.7711	48.2108	30582.0014	28.3208#	0	0
		2	27971.9330	47.4285	28018.4339	27.9960#	0	0
		3	25174.9591	47.3207	25342.8573	28.9641#	0	0
		4	16939.9252	47.7670	17069.3787	28.7961#	0	0
		5	25111.5494	47.0773	25290.0624	28.1802#	0	0
	4	1	41703.5390	49.4792	41735.8917	29.7541#	0	0
		2	33610.1650	47.2077	33956.1019	28.2429#	0	0
		3	27960.8019	48.3227	28108.3184	28.6032#	0	0
		4	30623.6897	47.3816	30801.4593	28.6520#	0	0
		5	33516.3790	47.4319	33763.9694	27.6199#	0	0
	6	1	46608.5128	49.5887	46676.0498	29.5379#	0	0
		2	33257.9164	46.9079	33595.2723	27.7100#	0	0
		3	30336.3947	46.8341	30555.9277	28.1373#	0	0
		4	41582.8206	47.6096	41862.4689	28.0254#	0	0
		5	38582.0314	46.8796	38861.4423	28.3160#	0	0
	8	1	48860.9671	47.3805	49235.5746	27.3479#	0	0
		2	43798.7295	47.0213	44257.7937	28.0643#	0	0
		3	29798.5994*	46.8825	30053.2034	28.3602#	0	0
		4	32631.8759*	46.7695	32760.2706	27.6604#	0	0

		5	51667.7162*	49.4426	51766.3392	29.2731#	0	0
10000	2	1	44995.62959*	91.3101	45095.4263	55.2857#	0	0
		2	50397.3919*	93.1353	50594.3232	55.0188#	0	0
		3	44678.9752*	91.5378	44773.8882	54.6213#	0	0
		4	72030.8102*	95.1735	72098.9131	57.1263#	0	0
		5	55626.1801*	92.7132	55850.1388	54.9278#	0	0
	4	1	56178.1336*	96.7222	56299.1400	56.0003#	0	0
		2	61648.6367*	94.2923	62123.0192	55.4919#	0	0
		3	50602.4253*	93.8726	51000.8010	55.6772#	0	0
		4	83427.7945*	98.3289	83498.3744	58.5123#	0	0
		5	61322.2689*	94.5945	61499.7119	55.7823#	0	0
	6	1	77367.8716*	95.4415	77994.1690	55.9685#	0	0
		2	71695.6170*	93.9842	71897.4887	56.5441#	0	0
		3	82413.1384*	95.5073	82998.8523	55.8619#	0	0
		4	71753.0207*	96.9672	72149.8843	55.1249#	0	0
		5	71721.3494*	93.2621	72200.2590	55.1061#	0	0
	8	1	65494.3089*	94.5845	65882.5474	55.4735#	0	0
		2	92452.1676*	93.3139	93061.5878	55.5215#	0	0
		3	70840.7464*	94.3601	71238.3939	56.3244#	0	0
		4	70976.6378*	95.0655	71374.9500	55.4191#	0	0
		5	103422.2464*	98.1516	103558.4557	60.3018#	0	0
20000	2	1	111765.8647*	182.7161	112085.1843	108.9479#	0	0
		2	111727.2388*	190.8428	112206.5213	108.5324#	0	0
		3	67796.9667*	180.1987	68028.3165	112.0981#	0	0
		4	78694.1771*	182.9115	78799.6933	112.0081#	0	0
		5	56599.4680*	178.8788	56855.8259	114.3366#	0	0
	4	1	144864.8255*	190.4318	145417.7687	112.7082#	0	0
		2	89847.9308*	189.0057	90195.7137	112.8773#	0	0
		3	155496.5881*	183.9597	156004.5582	110.4789#	0	0
		4	156063.6331*	188.9985	156664.8936	112.2890#	0	0
		5	123018.7277*	183.7560	123397.6262	114.6917#	0	0
	6	1	111373.5054*	193.1688	111755.4988	115.4670#	0	0
		2	132728.0185*	185.0246	133413.0394	111.9773#	0	0
		3	132615.1747*	213.9691	133152.2969	151.9103#	0	0
		4	143547.9770*	247.2365	143979.2754	152.7364#	0	0

		5	166028.1800*	252.0135	166444.1665	152.6140#	0	0
	8	1	163982.9366*	193.0285	164421.0688	116.5118#	0	0
		2	152680.0590*	194.0022	153258.6031	114.6965#	0	0
		3	197317.6382*	191.6965	198162.4881	116.4662#	0	0
		4	185634.6418*	223.6339	186858.3181	156.8681#	0	0
		5	164039.8064*	257.2186	164816.4179	159.8602#	0	0
30000	2	1	184235.5864*	297.7340	184771.4676	181.6212#	0	0
		2	151224.6161*	295.6989	151762.8928	181.0651#	0	0
		3	184206.6423*	291.8734	184513.0225	177.3899#	0	0
		4	134615.1962*	281.2754	135062.4338	184.0240#	0	0
		5	151634.9375*	297.1372	151891.1863	182.9432#	0	0
	4	1	201088.6911*	329.6129	201485.3616	193.0447#	0	0
		2	151895.4943*	309.7966	152408.4398	190.9758#	0	0
		3	135135.0474*	305.3099	135298.1429	191.9990#	0	0
		4	250851.5289*	331.7673	250871.7989	259.9966#	0	0
		5	168200.4950*	382.8928	168608.7894	235.2251#	0	0
	6	1	215992.1343*	296.3368	216582.1854	185.1597#	0	0
		2	215579.5857*	292.7890	216383.5029	180.5908#	0	0
		3	232249.7397*	290.2822	233038.6755	178.1758#	0	0
		4	183313.3774*	296.4093	183874.4985	177.2847#	0	0
		5	265385.8867*	300.4016	265968.0943	180.3260#	0	0
	8	1	263265.9515*	308.9004	264029.19420	178.1202#	0	0
		2	278947.0550*	296.6717	279843.1408	188.9767#	0	0
		3	246739.3221*	310.1531	247449.7233	177.0255#	0	0
		4	196669.8673*	291.7890	197358.9994	180.5270#	0	0
		5	197220.7125*	286.7491	198212.1388	175.6457#	0	0

References

[1] A. Allahverdi, C. T. Ng, T. C. E. Cheng, and M. Y. Kovalyov, "A survey of scheduling problems with setup times or costs," European Journal of Operational Research, vol. 187, no. 3, pp. 985–1032,(2008).

[2] A. Allahverdi, "The third comprehensive survey on scheduling problems with setup times/costs," European Journal of Operational Research, vol. 246, no. 2, pp. 345–378, (2015).

[3] A. S. Hameed and H. A. Cheachan ‘’ Genetic Algorithm and Particle Swarm Optimization Techniques for Solving Multi-Objectives on Single Machine Scheduling Problem’’, Ibn Al-Haitham Jour. for Pure & Appl. Sci. 33 (1), (2020)

[4] B. Mitras, H. Talaat Yaseen, A Novel Invasive Weed Optimization Algorithm (IWO) by Whale Optimization Algorithm(WOA) to solve Large Scale Optimization Problems, Vol.25 No.110. (2019).

[5] Bruno, J., and Downey, P., "Complexity of task sequencing with deadlines, set-up times and changeover costs", SIAM Journal on Computing 7 ,393-404, (1978).

[6]B. Jarboui and Abdelwaheb Rebaï ‘’ A nested iterated local search algorithm for scheduling a flowline manufacturing cell with sequence dependent family setup times’’ (2014).

[7]C. G. Martinez and M. Lozano ‘’local search based on genetic algorithms’’, (2007).

[8] Chachan, H.A., Hameed, A.S. ‘’Exact methods for solving multi-objective problem on single machine scheduling’’ Iraqi Journal of Science, 60(8), pp, 1802-1813 (2019).

[9] Chachan, H.A, Ali, F.H., Ibrahim, M.H. ‘’Branch and Bound and Heuristic Methods for solving Multi-objective Function for Machine Scheduling Problem’’ Proceedings of the 6th International Engineering Conference ‘’Sustainable Technology and Development’’, IEC 2020, p.p. 109-114,9122793 (2020).

[10] Cheachan, H.A., Ibrhim, M.H., Mohammed, H.A.A ‘’ Scheduling of single machine with release date to minimize multiple objective functions’’ IOP Conference series: Materials Science and Engineering, 571(1), 012003, (2019).

[11] Cheachan, H.A. Mohammed, H.A., Cheachan, F.A. ‘’Fuzzy Distances And Their Applications On Fuzzy Scheduling’’ Eastern-European Journal Of Enterprise Technologies 3(4-63), pp.23-30 (2013).

[12] Graham R.L., Lawler E.L., Lenstra J.K., and Rinnooy Kan A.H.G. "Optimization and Approximation in Deterministic Sequence and Schedule: A Survey Annals of Discrete Mathematics, volum5, (1979).

[13]J.-B. Wang and J.J. Wang, "Single machine group scheduling with time dependent processing times and ready times," Information Sciences, vol. 275, pp. 226–231,(2014).

[14] L. N. Van Wassenhove, C. N. Potts and H. A. J. Crauwels ‘’ Local Search Heuristics for Single Machine Scheduling. (1997).

[15] Okanya, A. ., Asogwa, J., & Onyedikachi, I. . . . (2021). Indoor Environmental Quality (IEQ) in Nigerian Tertiary Institutions: The Effect on Performance of Building Technology Lecturers. *Middle Eastern Journal of Research in Education and Social Sciences*, 2(1), 172-186. https://doi.org/10.47631/mejress.v2i1.143

[16] Pirlot, M., "General local search heuristics in combinatorial optimization: a tutorial", Belgian Journal of Operations Research, Statistics and Computer Science, 32,8-67,(1992).

[17]S., Sharma, p. Bhambu, Artificial Bee Colony Algorithm: A Survey. International Journal of Computer Applications (0975 –

8887). Volume 149 – No.4, September (2016).

[18]Walser, Joachim Paul. Integer optimization by local search: a domain-independent approach. Springer-Verlag, (1999).

[19] Ariyo, S. O., Ogbonnaya, . K. . E., & Bamgboye, S. O. (2020). Strategies Required to improve Students' Industrial Training Program between Local Businesses and Technical Colleges in Enugu State, Nigeria. *Journal of Advanced Research in Economics and Administrative Sciences*, *1*(2), 45-55. https://doi.org/10.47631/jareas.v1i2.95

[20] Z. Saleh Ahmadi, A. Manafi, How Can Bee Colony Algorithm Serve Medicine? Bushehr University of Medical Sciences, [Bushehr, Iran, Vol.3/No.2/July (2014).

[21] Okanya, A. . V., Vincent, D. . A., & Onyekachi, A. J. . (2021). Rationale for Material Selection in Landscaping for Checking Intrusion in Public and Private Primary Schools in Nsukka Urban Area of Enugu State, Nigeria. *Middle Eastern Journal of Research in Education and Social Sciences*, 2(1), 119-129. https://doi.org/10.47631/mejress.v2i1.141

[1]