International Journal of Industrial Engineering & Production Research June 2023 Vol. 34, No. 2: 1-16
DOI: 10.22068/ijiepr.34.2.8

RESEARCH PAPER

Makespan Minimization using Hybrid Heuristic and Evolutionary Genetic Algorithm

Prasad Bari*[1] & Prasad Karande[2]

Received 8 February 2023; Revised 9 May 2023; Accepted 27 May 2023;

ABSTRACT
This paper presents a model for minimizing the makespan in the flow shop scheduling problem. Due to the impact of increased workloads, flow shops are becoming more popular and widely used in industries. To solve the challenge of minimizing makespan, a Hybrid Heuristic and Evolutionary Genetic Algorithm (HHEGA) is proposed. The proposed HHEGA algorithm is tested using the simulation software and demonstrated with steel industry data. The results are compared with those of the best available flow shop problem algorithms such as Palmer's slope index, Campbell-Dudek-Smith (CDS), Nawaz-Enscore-Ham (NEH), Genetic Algorithm (GA) and Particle Swarm Optimization (PSO). According to empirical results and relative differences from the lower bound, the proposed technique outperforms the three heuristics and two metaheuristics algorithms in three of six cases, while the remaining three produce the same results as the NEH heuristic. In comparison to the steel industry's regular job scheduling technique, the simulation model based on HHEGA can save 4642 hours. It was discovered that the suggested model enhanced the job sequence based on the makespan requirements.

KEYWORDS: *Makespan; Scheduling; Heuristic; Metaheuristic; Genetic algorithm; Lower bound.*

1. Introduction

Scheduling is a crucial technique, particularly in the industrial or service sectors, to distribute resources in a way that is more systematic, effective, and efficient. It is thus the process of allocating work, commonly referred to as jobs, to various resources, frequently referred to as machines, to optimize a specified objective function. The scheduling function is understood by using fundamental ideas, models, procedures, and rational conclusions in the process of decision-making [1]. Bari and Karande used the Preference Ranking Organization Method for Enrichment Evaluation- Geometrical Analysis for Interactive Aid (PROMETHEE-GAIA) in evaluating several sequencing rules with varied objective functions linked to flow time and tardy performance measures [2]. Despite the abundance of literature on scheduling, there is a significant gap between the theory and application of developed methodologies. Most of the methods frequently overlook significant real-world factors and concentrate on relatively minor problem occurrences. As a result, manual scheduling remains rather popular in practice. On the other hand, as technology advances and businesses have ever-expanding product ranges, scheduling is becoming a more difficult process.

In a typical Flow Shop Scheduling Problem (FSSP), *n*-jobs are to be handled on *m*- machines in a similar technical sequence. To reduce a certain measure of production cost, a schedule must be created for handling these '*n*' jobs on '*m*' machines [3]. There are *pow*(*n*!, *m*) probable sequences for this problem. This value is exorbitantly big even for a minor problem size. If the order in which the jobs are managed on each machine is assumed to be identical, the problem converts into a Permutation Flow Shop Scheduling Problem (PFSSP), and the number of possible sequences is reduced to *n*! The minimization of the maximum completion time generally referred to as the makespan, is the optimization criterion that is most frequently researched for the FSSP. Many benefits result from time reduction, including lower costs, higher utilization, higher productivity, and less inventory [4]. Azimi and Sholekar developed a model for minimizing the net present value

* Corresponding author: ***Prasad Bari***
prasad.bari@fcrit.ac.in

1. *Department of Mechanical Engineering, Fr. C. Rodrigues Institute of Technology, Navi Mumbai, India.*
2. *Department of Mechanical Engineering, Veermata Jijabai Technological Institute, Mumbai, India.*

and makespan of a scheduling project. They used a simulation-based optimization approach to tackle the problem with non-deterministic duration times for activities [5]. Researchers sequence the jobs that are awaiting processing at various work centres using a wide variety of sequencing rules. Fewest Operations Remaining, First Come First Served, Shortest Processing Time, Earliest Due Date, Critical Ratio, Slack Time, and Next Queue are a few popular rules. These rules produce poor solutions for problems that are comparatively bigger and are quite approximate. Therefore, researchers began developing heuristics to reach approximations with satisfactory accuracy [6]. Further, the FSSP with the objective of minimizing the makespan is nondeterministic polynomial-time hard (NP-hard) and is a highly researched problem [7] [8] [9]. As a result, many of the proposed solution approaches are based on heuristic and metaheuristic algorithms.

Johnson's early work [10] created the interest of researchers working on various scheduling objectives such as optimization of makespan, tardiness, and lateness. A lot of manufacturing environments use flow shop layouts because they have several advantages like fewer materials to be handled, cell activity can be better seen and controlled, cells can use conveyors more easily, tiny transfer batches can be used more effectively, and input/output flow monitoring is made easier; over more traditional job shop arrangement [11]. The early study was constrained because of the restricted computing power and software/programs available. The investigators were simply attempting to expand Johnson's algorithm to '*m*' machines. Gradually many heuristic approaches have been developed and proposed to resolve the FSSP. Palmer recommended assigning a slope index to every job and scheduling by ordering jobs based on it [12]. Hundal and Rajgopal examined and improved this approach by computing two additional slope indexes [13]. Similar to Johnson's algorithms, Campbell et al. proposed a Campbell-Dudek-Smith (CDS) heuristic that chooses the optimal outcome by taking into account two assumed machines and *n* jobs at every stage of processing [14]. They reported that better results are produced but with longer computing times. Nawaz et al. created Nawaz-Enscore-Ham (NEH) heuristic, which is still observed as the outstanding solution to the PFSSP [15]. It is predicated on the indication that jobs with longer processing times should be planned as early in the sequence as practicable on all machines. NEH heuristic is widely acknowledged as an effective heuristic for minimizing the makespan currently available and hence numerous NEH-based versions have been developed over the years [16]. In the literature, many NEH-based heuristics have been suggested, examined, and documented. [17] [18] [19].

Continuous study into the PFSSP is meaningful since novel heuristics, as well as novel enriched methodologies for heuristics or meta-heuristics, can always be anticipated. Furthermore, continuous refinement of heuristic approaches for simple situations may yield better outcomes than for increasingly complex ones. It is worthwhile to attempt to enhance NEH as it is an outstanding heuristic approach for PFSSP [20]. A significant amount of research is still being done to enhance the NEH approach [21] [22] [23] [24].

Good heuristics have considerable advantages for large problems since they typically produce good solutions quickly. The heuristic method, a local search approach embedded in the structure of the heuristic method to minimize the makespan of scheduling problems and also evaluated the performance of the heuristic method, with a lower bound metric [25]. Additionally, there are instances where sufficiently effective heuristics are more accurate than metaheuristics. In comparison to the Genetic Algorithm (GA), this is true of NEH [26]. The NEH method is distinguished for flow shop scheduling. This heuristic is popular since it is not just effective but also easy to compute. This heuristic has served as a benchmark against which many researchers have compared their findings [27] [28]. Among the metaheuristics for FSSP, GA is the most commonly used [29]. Bari et al. applied GA to minimize tardiness and flow time-related performance metrics [30]. Khatami et al. developed iterated local search algorithm to minimize the makespan in the ordered FSSP [31]. Babor et al. analyzed manufacturing lines from small and medium-sized bakeries to discover the best makespan using variants of PSO and GA variables [32]. Allali et al. and Zhao et al. investigated a combination of heuristics and metaheuristics to minimize makespan in FSSP [33] [34]. Umam et al. used the tabu search procedure in conjunction with the GA to address the FSSP and reduce the makespan [35]. GA yields good, quick, and proficient outcomes when exploring multifaceted solution space (global search). However, while undertaking exploitation (local search), it produces poor results because it gets easily locked in the optimal local space. On the other hand, tabu search outperforms local search to keep GA from becoming locked in a local optimum. Heuristics and GA alone do not produce improved results, hence in the present research, heuristics like Palmer's slope index, CDS, NEH and a meta-heuristic such as Particle Swarm Optimization (PSO) based Hybrid Heuristic and Evolutionary GA (HHEGA) is developed to solve the FSSP. The purpose is to curtail the makespan or completion time of the scheduling problem. Based on

data gathered from the steel industry, the HHEGA is modelled. The motivation behind modelling this algorithm is to find the optimal sequence of jobs to minimize the total completion time so that the saved time can be used efficiently by the manufacturing company.

The article is further structured into subsequent sections. Section 2 covers methodology, addressing heuristics, metaheuristics, and the proposed HHEGA algorithm. Section 3 outlines the hypothesis testing, including dataset descriptions, lower bound and makespan computations. It also shows a comparative analysis of the algorithms used for scheduling. Section 4 contains a case study and discussion of the results of the steel industry, as well as the model that was constructed for it. Finally, section 5 concludes the article.

2. Methodology

The PFSSP is a combinatorial search problem with $n!$ likely sequences. The sequence with the shortest total completion time could be found by computing all $n!$ sequences, but this process is both expensive and impracticable. The exact approaches such as the branch and bound methodology have two unavoidable drawbacks. Firstly, for big problems, the computing demands can be very high. Second, the search effort is dependent on the problem's data, therefore even for moderately sized problems, there is no guarantee that the answer can be found soon. Heuristic algorithms circumvent these two disadvantages by being able to solve complex problems with little computer effort and by having predictable computational needs for problems of a certain size. Heuristic methods have the disadvantage of not ensuring optimality, and in some cases, it may even be challenging to assess their efficacy. In this study, to overcome these disadvantages, a hybrid technique incorporating heuristics and GA is proposed to discover the near optimal sequence of jobs. The famous heuristics such as Palmer's slope index, CDS and NEH, and metaheuristic techniques like PSO are joined with GA and their procedures are described below in detail.

2.1. Heuristic approaches

2.1.1. Palmer's slope index

A job significance function computes a value depending on job processing times and can help in determining the job processing sequence in FSSP. Palmer adopted this approach in the design of a job significance function, which he termed "slope index" for the job [12]. The significance function was purposefully preferred to prioritize jobs that tend to go from smaller to longer processing times as they pass through the machines. Equation (1) below defines the slope index.

$$SI_j = \sum_{i=1}^{m} P_{ji} * (m - 2 * i + 1) \quad (1)$$

where P_{ji} is the processing time (PT) of job j on machine i

A schedule is created by arranging the jobs in increasing order of their slope indexes. The pseudo-code of the algorithm is provided below.

Algorithm Palmer's slope index

```
For job j in range n
do
   Calculate the slope for job 1 as
   Slope1=Slope1+(m-2*j+1)* PT of job j on machine 1
   Calculate the slope for job 2 as
   Slope2=Slope2+(m-2*j+1)* PT of job j on machine 2
   Calculate the slope for job 3 as
   Slope3=Slope3+(m-2*j+1)* PT of job j on machine 3
   and so on for all jobs
end

Arrange jobs according to the computed slopes (Slope1, Slope2, Slope3, …) for all jobs in increasing order to obtain optimal sequence

Output = optimal sequence
```

2.1.2. CDS

The next heuristic technique considered for determining the makespan of the scheduling problem is the CDS algorithm. This technique generates multiple sequences from which the best sequence is chosen heuristically considering Johnson's rule. The algorithm is a multi-pass application of Johnson's rule, pseudo problems are taken from the original problem, with processing times (PTM_1 and PTM_2) for two hypothetical machines, m_1 and m_2 respectively. The original problem is divided into m-1 sub-problems using equation (2) and equation (3) for each pass.

$$PTM_1 = \sum_{i=1}^{pass} P_{ji} \quad (2)$$

$$PTM_2 = \sum_{i=m-pass+1}^{pass} P_{ji} \quad (3)$$

For iteration 1, $PTM_1 = P_{1\,j}$ and $PTM_2 = P_{m\,j}$ which means the jobs' PT on the first and last machine are included in the sub-problem. For iteration 2, $PTM_1 = P_{1\,j}$

$+ P_{2j}$ and $PTM_2 = P_{m-1\,j} + P_{m\,j}$ which means, aggregated PT for each job on the first two and last two machines, included in the sub-problem. Generate the m-1 sub-problem for m-1 passes, find the job sequence of jobs from the sub-problem and makespan is calculated for each sub-problem with the original PT. Now, the optimal makespan with the sequence is selected. The algorithm's detailed procedure is given below.

Algorithm CDS

For m-1 pass
do
For pass 1:
PTM_1 is the PT of the first machine for all jobs j
PTM_2 is the PT of the last machine for all jobs j
Find a sequence of the jobs using Johnson's rule
Calculate the makespan ($makespan_{Pass1}$) of sequence with the original PT
For pass 2:
PTM_1 is the summation of PT for the first two machines
PTM_2 is the summation of PT for the last two machines
Find a sequence of the jobs using Johnson's rule
Calculate the makespan ($makespan_{Pass2}$) of sequence with the original PT
For pass 3:
PTM_1 is the summation of PT for the first three machines
PTM_2 is the summation of PT for the last three machines
Find a sequence of the jobs using Johnson's rule
Calculate the makespan ($makespan_{Pass3}$) of sequence with the original PT
and so on for the remaining passes.
end

Compare the makespan of all passes ($makespan_{Pass1}$, $makespan_{Pass2}$, $makespan_{Pass3}$, …) and choose a sequence having minimum makespan with optimal sequence

Output = optimal sequence

2.1.3. NEH

Like CDS, the NEH algorithm does not produce sub-problems. It takes three phases to tackle the problem. In Step 1, a sequence called initial sequence S_0 is created by organizing the n jobs as per the summation of their processing times across the m machines in a non-increasing manner. Step 2 selects the first two jobs from S_0, and for the two jobs with the shortest makespan, a sequence is created. The remaining jobs from sequence S_0 are then sequentially added to create a complete sequence in Step 3. Each job is added to a location that minimizes the makespan of the provided partial sequence among all viable positions. The algorithm below provides detailed steps.

Algorithm NEH

For each machine 1 to m
do
Perform the sum of PT of jobs j
Initial Sequence = Place the jobs in decreasing order of their sum
end
For job j in the initial sequence
do
first job=Job which is at position 1
Calculate makespan
second job=Job which is at position 2
Place the second job before and after the first job
Calculate the makespan of both sequences
Compare the makespan and choose a sequence with less makespan
third job=Job which is at position 3
Create three sequences by adding the third job to every available location in the sequence selected in the preceding step
Calculate the makespan of three sequences
Compare the makespan and choose a sequence with less makespan
and so on for all jobs in the initial sequence
end
A sequence with less makespan is optimal sequence
Output = optimal sequence

2.2. Metaheuristic approaches

2.2.1. GA

The GA is an evolutionary process-based optimization method of searching, that uses a population of sequences. In this study, a population of random sequences is taken, and the survival of each sequence is assessed using smaller the best makespan after applying both crossover and mutation operations. Then the best sequence out of all the sequences is chosen, promising a better result. The phases of GA are as follows -

Representation of chromosome - A sequence to the n-jobs is represented as a chromosome.

Fitness function - It evaluates the performance measure to be enhanced. A makespan is calculated for all chromosomes and lower the best policy is used for the survival of the fittest.

Initial population - The superiority of the final result is significantly influenced by the start sequences or population. The population is randomly selected using a permutation of the number of jobs in the data set.

Crossover - A couple of child sequences are produced from a couple of parent sequences using the crossover operation. By selecting a position at random, the

crossover operator switches the sub-sequences before and after that position across two sequences. As a result, two-parent sequences are crossed over to create two new children sequences.

Mutation- In this work, the mutation is achieved by switching two randomly chosen jobs in sequence.

2.2.2. PSO

The PSO is one of the metaheuristic techniques to determine a sequence for scheduling problems. The PSO has various steps; first, randomly generate the population (particles). In the search space, assign them a random velocity. Then, to determine each particle's optimum position individually and the best position discovered throughout the population as a whole, a makespan is computed for each particle or sequence. Finally, throughout each cycle, each particle updates its current velocity, which it utilizes to migrate to a new point. This method lingers until a stopping criterion is met and finds the optimal solution with the makespan value, lower the best. Step-by-step instructions for the algorithm are provided below.

Algorithm PSO

Parameters initialization
The inertiaweight(w0)=1.2, Change in inertiaweight (delta) = 0.975
//The inertia weight is the decisive factor in the PSO's convergence behaviour.
For each iteration
do
Initialize the population (particles)
Update position (X) of the particle for job j on machine i
temp=X[j][i]
X[j][i]+= V[j][i]
Update velocity (V) of the particle for job j on machine i for the next iteration
V[j][i]= w0*V[j][i]+c1*Math.random()*(P[j][i]-temp)+c2*Math.random()*(G[i]-temp)
//*c*1 as the coefficient of cognitive acceleration, *c*2 as the coefficient of social acceleration, G be the best-known position of the entire population
w0 = w0*delta;
end
Compute makespan for each row and choose the smallest value as the bestP with optimal sequence
Insert the row into population if the completion time of the row is less than all other rows and assign the value to bestP
Move to bestP from completion time to the best
Find local best
Determine global best using local best to obtain the optimal sequence
Output = optimal sequence

2.3. Hybrid heuristic and evolutionary-GA (HHEGA)

HHEGA is a combinatorial algorithm that combines GA with Palmer's slope index, CDS, and NEH heuristics and the PSO metaheuristic algorithm. The sequences produced by these heuristics and metaheuristics are taken into account by the proposed method as a part of the initial population. Additionally, in the initial population of the following iteration, the best sequence from each iteration is inserted as a chromosome. The near-optimum solution is then found using the GA approach.

The formulation of the HHEGA algorithm is presented below.

Algorithm HHEGA

For each iteration
do
Select the initial population by doing permutation over a number of jobs
Save as parent list
Add sequence generated with Palmer's slope index, CDS, and NEH heuristic in the parent list
Add sequence generated with PSO metaheuristic in the parent list
Apply crossover operation on the parent list
Select two random places
Exchange the sub-sequences before and after the chosen places and produce the offspring
Save offspring list
Apply mutation operation on offspring list
Choose two jobs
Exchange the jobs at the position and produce the offspring
Save offspring list
Merge parent and offspring lists to obtain the total chromosomes
Evaluate total chromosomes
Compute the makespan for the chromosomes
Compare the makespan of chromosomes
Select the fittest chromosome as the best sequence
Add the best sequence in the parent list for the next iteration
If stopping criteria are met, then stop
and optimal sequence=best sequence with the lowest makespan
end
Output = optimal sequence

Fig. 1 presents the components of the proposed HHEGA algorithm. The three components of the proposed technique are sequences generation, evaluation, and selection of the best solution. In the sequences

generation element, generate sequences with heuristics such as Palmer's slope index, CDS, NEH and metaheuristic algorithm such as PSO. To avoid any potential bias brought on by particular heuristic rule sequences, the remaining sequences are produced randomly. Following the generation of the initial population using heuristics, metaheuristics, and randomness, these sequences go through crossover and mutation operations in series, leading to the evolution of additional new sequences as the offspring population. The two-job simple crossover and a job flip mutation are performed. The crossover points are selected randomly. After performing crossover and mutation, the total sequences are then obtained by merging the sequences produced in the preceding steps with the initial population. In the sequences evaluation part of HHEGA, based on each sequence's makespan, the sequences are assessed. The sequence with minimum makespan is selected as the best. The best sequence from this iteration has now been added as one of the members of the initial population of the succeeding iteration. If the stopping requirements are satisfied, the chosen sequence is considered a near-optimal sequence, otherwise the method described above is used for the following iteration.

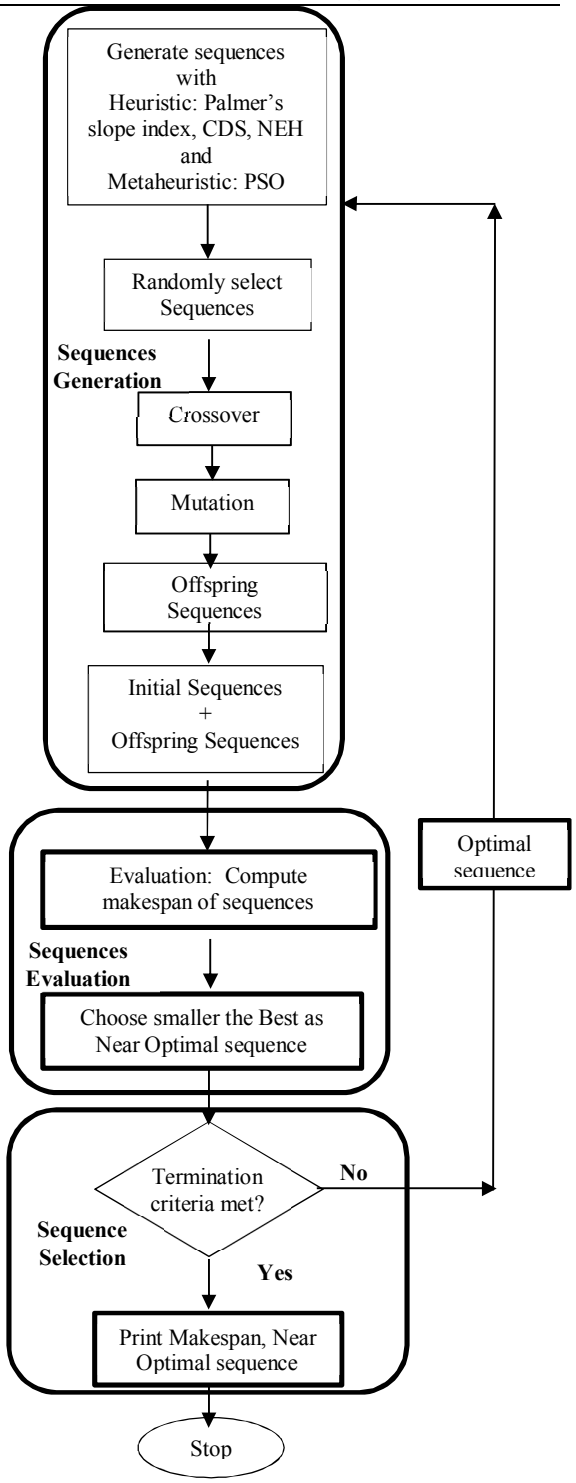


Fig. 1. Components of HHEGA

3. Hypothesis Testing

This section analyzes six randomly generated data sets to assess the performance of the heuristic techniques such as Palmer's slope index, CDS, NEH, metaheuristics GA, PSO and the proposed HHEGA

algorithms for makespan minimization. Comparisons are made between the results obtained through these algorithms.

3.1. Dataset

The datasets are generated randomly for three jobs and four machines, four jobs and four machines, four jobs and five machines, four jobs and six machines, nine jobs and five machines, and twenty jobs and five machines. These datasets are used to evaluate the six different algorithms.

3.2. Computation of lower bound (LB) for makespan

The FSSP includes allocating n jobs to m machines for processing with the same sequence of machines. The lower the makespan of sequence means the better the solution for allocating n jobs to m machines. The summation of the PT of all jobs on each machine is calculated and the maximum sum among all the machines is the LB for makespan. The makespan can be made more precise or accurate by involving other machines in the estimation of the LB. Following are the steps to find the precise LB for makespan.

- LB for corner machines which means the first and last machine is computed using equation (4) and equation (5) respectively.

$$LB\ for\ first\ machine = \min_{i=2\ to\ m}(P_{j2} + P_{j3} + P_{j4} + \cdots + P_{jm}) + \sum_{j=1}^{n} P_{j1} \quad (4)$$

$$LB\ for\ last\ machine = \min_{i=1\ to\ m-1}(P_{j1} + P_{j2} + P_{j3} + \cdots + P_{jm-1}) + \sum_{j=1}^{n} P_{jm} \quad (5)$$

- LB for all intermediate machines is computed using equation (6).

$$LB\ for\ specific\ machine = \sum \min(processing\ time\ for\ jobs\ on\ machine\ other\ than\ specific\ machine) + \sum_{j=1}^{n} P_{ji}(specific\ machine) \quad (6)$$

where specific machine m = 2, 3, 4, …, m-1

- LB for the makespan of the dataset is computed using equation (7).

LB makespan for dataset = max (LB for first and last machine, LB for intermediate machines) (7)

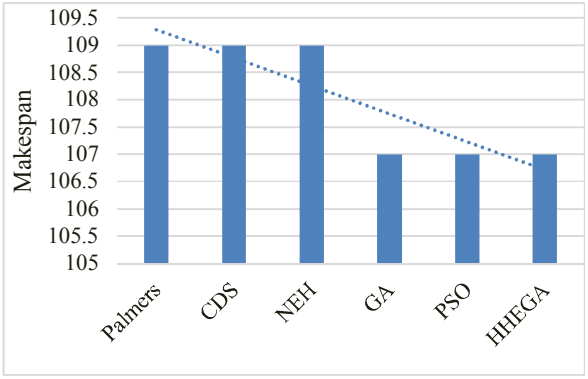
3.3. Application of heuristics, metaheuristics and proposed algorithm on datasets

The makespan is computed for each dataset using different heuristics, metaheuristics and the proposed HHEGA algorithm which are recorded in Tab. 1. The computed LB is also recorded in Tab. 1. The first column shows the name of the dataset, for example, Test01_3x4, where, 01 is the first dataset, and 3x4 means three jobs and four machines scheduling problems. 100 iterations are used for GA, PSO, and HHEGA to discover the best sequence for the FSSP. Each dataset is examined with different parameters such as initial population size 200, crossover rate 0.8, and mutation rate 0.1, The crossover rate increases the number of developing sequences when it is high and mutation which causes sequence properties to change randomly.

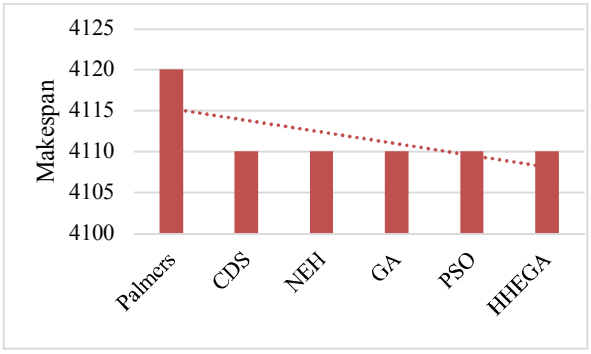
Fig. 2 (a-f) shows a graphical representation of trends in the makespan of each dataset for six different algorithms. From Fig. 2, it is observed that the HHEGA does better than all the other algorithms.

Tab. 1. LB and makespan

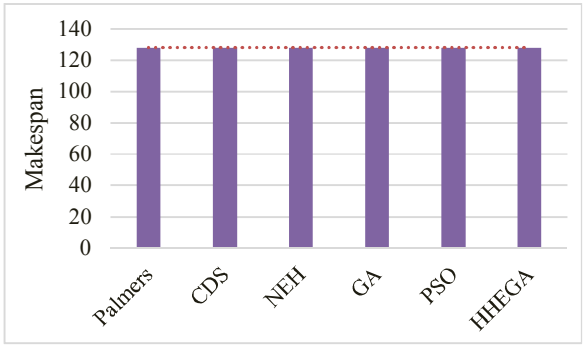
Dataset	LB	Palmer's Slope Index	CDS	NEH	GA	PSO	HHEGA
Test01_3x4	107	109	109	109	107	107	107
Test02_4x4	4070	4120	4110	4110	4110	4110	4110
Test03_4x5	121	128	128	128	128	128	128
Test04_4x6	1231	1260	1260	1243	1231	1231	1231
Test05_9x5	228	241	238	234	241	240	232
Test06_20x5	1232	1384	1390	1286	1334	1310	1286



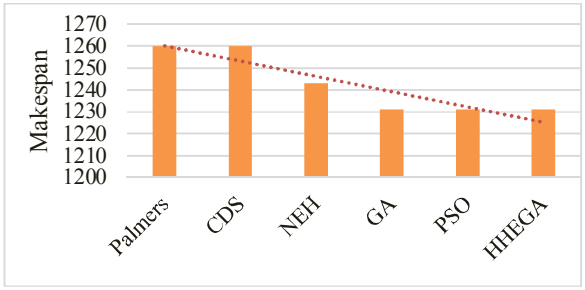
(a) Test01_3x4



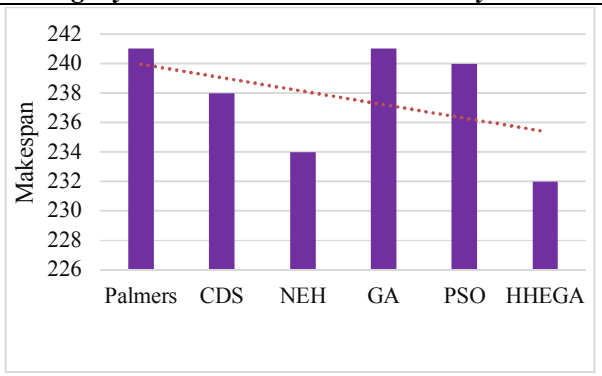
(b) Test02_4x4



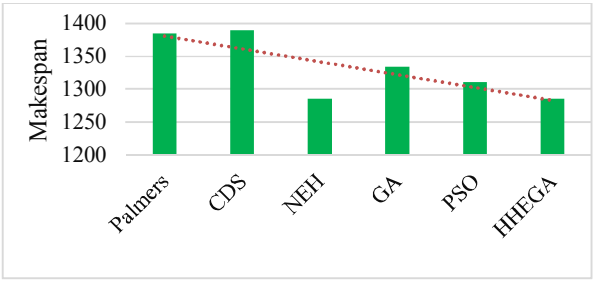
(c) Test03_4x5



(d) Test04_4x6



(e) Test05_9x5



(f) Test06_20x5

Fig. 2. Makespan for datasets

Relative difference from the LB is computed with equation (8) and recorded. A smaller value for the relative difference from the LB indicates better algorithm performance. Tab. 2 shows a comparative analysis of considered heuristics, metaheuristics and HHEGA technique.

$$Relative\ difference\ from\ LB\ = \frac{makespan\ computed\ with\ algorithm - LB}{makespan\ computed\ with\ algorithm} \quad (8)$$

Tab. 2. Relative difference from LB

Dataset	Palmer's	CDS	NEH	GA	PSO	HHEGA
Test01_3x4	1.83486	1.83486	1.83486	0	0	0
Test02_4x4	1.21359	0.97323	0.97323	0.97323	0.97323	0.97323
Test03_4x5	5.46875	5.46875	5.46875	5.46870	5.46875	5.46875
Test04_4x6	2.30158	2.30158	0.96540	0	0	0
Test05_9x5	5.39419	4.20168	2.56410	2.14592	5	1.72413
Test06_20x5	10.98266	11.36691	4.19906	7.64617	5.95419	4.19906

Fig. 3a shows the relative difference from the LB for all datasets. The figure shows that Palmer's method performs poorly when measured against the other five algorithms. The proposed HHEGA algorithm is giving minimum makespan as compared with other algorithms. It can be seen that by finding the average relative difference from the LB, HHEGA in attaining the LB was better related to others and it is presented in Tab. 3 and Fig. 3b.

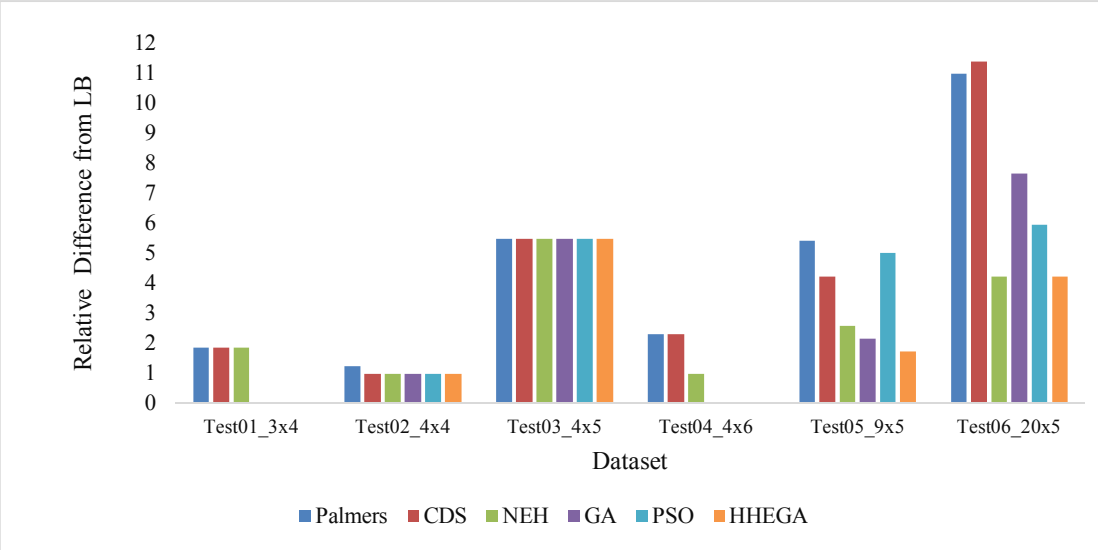


Fig. 3a. Visual representation of relative difference from LB

Tab. 3. Average relative difference from LB

Algorithm	Palmers	CDS	NEH	GA	PSO	HHEGA
Average	4.53260	4.35783	2.66757	2.70568	2.89936	2.06086

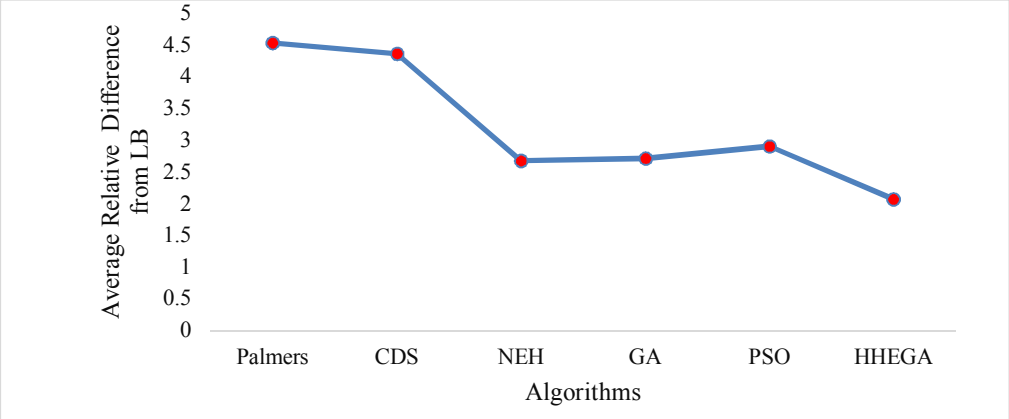


Fig. 3b. Visual representation of average relative difference from LB

4. Case Study: Application of HHEGA in A Steel Industry

In cooperation with a manufacturing company that specialized in the steel industry, a case study was carried out, to verify the methodology and contrast findings with a real-world scenario. The records from the company's datasheet are shown in Tab. 4.

Tab. 4. Dataset for steel company

Job Description				Machines and Processing time (in hours)					
Job No.	Drg. No.	Qty.	Job Name	CNC-Zayer	VTL-1	CNC-Union	VTL-2	CNC-Doosan	Drilling
1	110 995	32	B/U Chock Top DR	1120	1088	1920	1472	2240	640
2	110 994	32	B/U Chock Top OP	960	1152	1984	1344	3040	704
3	110 997	32	B/U Chock Bottom DR	1024	1120	1920	1536	2400	640
4	110 996	32	B/U Chock Bottom OP	960	1024	2112	1440	3072	800
5	1145 89	4	B/U Chock Top DR	100	120	200	192	120	80

6	1145 88	4	B/U Chock Top OP	120	128	216	208	140	72
7	1145 91	4	B/U Chock Bottom DR	112	120	200	180	140	88
8	1145 90	4	B/U Chock Bottom OP	140	140	240	220	160	80
9	61933	4	B/U Chock Top DR	80	100	160	152	140	64
10	61932	4	B/U Chock Top OP	100	112	180	168	168	80
11	61935	4	B/U Chock Bottom DR	92	100	160	160	180	60
12	61934	4	B/U Chock Bottom OP	112	120	200	180	224	72
13	20420 625	8	Work Roll Top DR	200	600	560	200	800	160
14	20420 624	8	Work Roll Top OP	240	520	640	224	880	144
15	20420 627	8	Work Roll Bottom DR	224	592	608	192	760	176
16	20420 626	8	Work Roll Bottom OP	256	624	704	240	1000	200
17	20421 604	8	Work Roll Top DR	144	440	480	360	560	120
18	20621 603	8	Work Roll Top OP	192	360	544	400	512	144
19	20421 606	8	Work Roll Bottom DR	176	416	608	480	480	120
20	20421 605	8	Work Roll Bottom OP	224	480	672	520	576	128
21	20421 071	8	Work Roll Top DR	160	320	400	280	440	200
22	20421 070	8	Work Roll Top OP	200	360	560	320	800	240
23	20421 073	8	Work Roll Bottom DR	240	336	448	336	496	256
24	20421 072	8	Work Roll Bottom OP	280	400	640	360	880	240
25	58153	6	IMR Top DR	90	72	180	48	300	90
26	58152	6	IMR Top OP	108	240	300	180	420	108
27	58155	6	IMR Bottom DR	96	90	150	60	336	84
28	58154	6	IMR Bottom OP	132	270	372	210	480	120
29	64615	4	IMR Top DR	40	72	160	48	232	60
30	64614	4	IMR Top OP	64	140	220	128	300	80
31	64617	4	IMR Bottom DR	48	80	128	56	240	60
32	64616	4	IMR Bottom OP	80	200	260	168	340	80

On a processor with the Windows 10 operating system and equipped with 8 GB of RAM and a 500 GB hard drive, the simulation model for the HHEGA algorithm is developed using Python programming shown in Fig. 4. Additionally, this model is compatible with a number of operating systems, such as Windows 7, Windows 8, and others. Other than this, there are no more unique hardware or software requirements. The dataset shown in Tab. 4 has job descriptions and machines with the PT of jobs. For computing makespan for the dataset job number, machines and PT of jobs on each machine are considered. The "Browse File" button is used to select an Excel sheet of the dataset and the number of iterations to be entered as stopping conditions for the algorithm as input to the model shown in Fig. 5. After clicking on the "Data" button the preview of data is visible as shown in Fig. 6. Once data is given as input clicking on the "Submit" button gives the optimal sequence with the makespan as shown in Fig. 7.

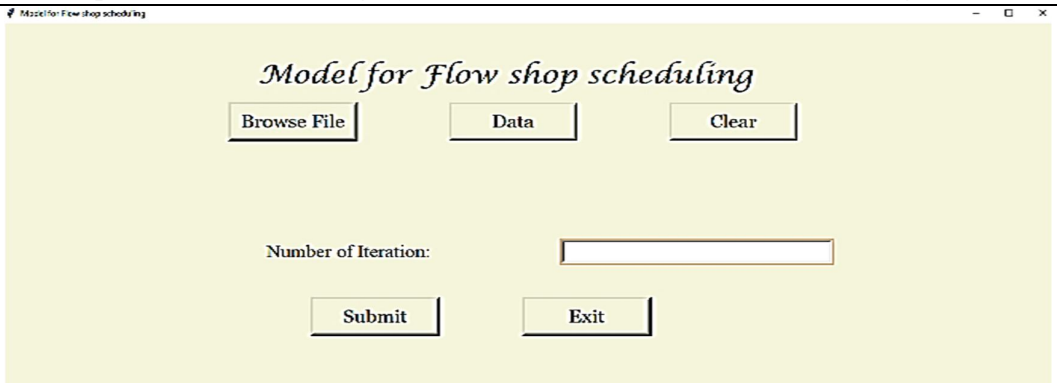


Fig. 4. Model for makespan in flow shop scheduling for *n* jobs, *m* machines

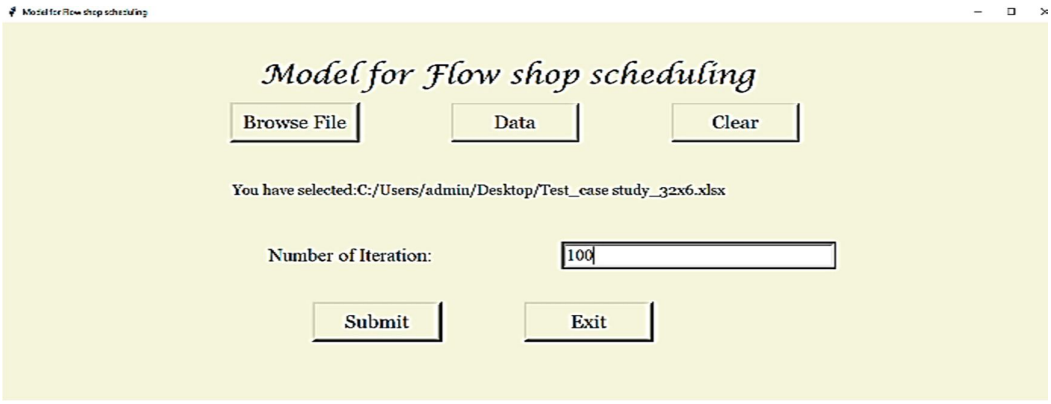


Fig. 5. Input data

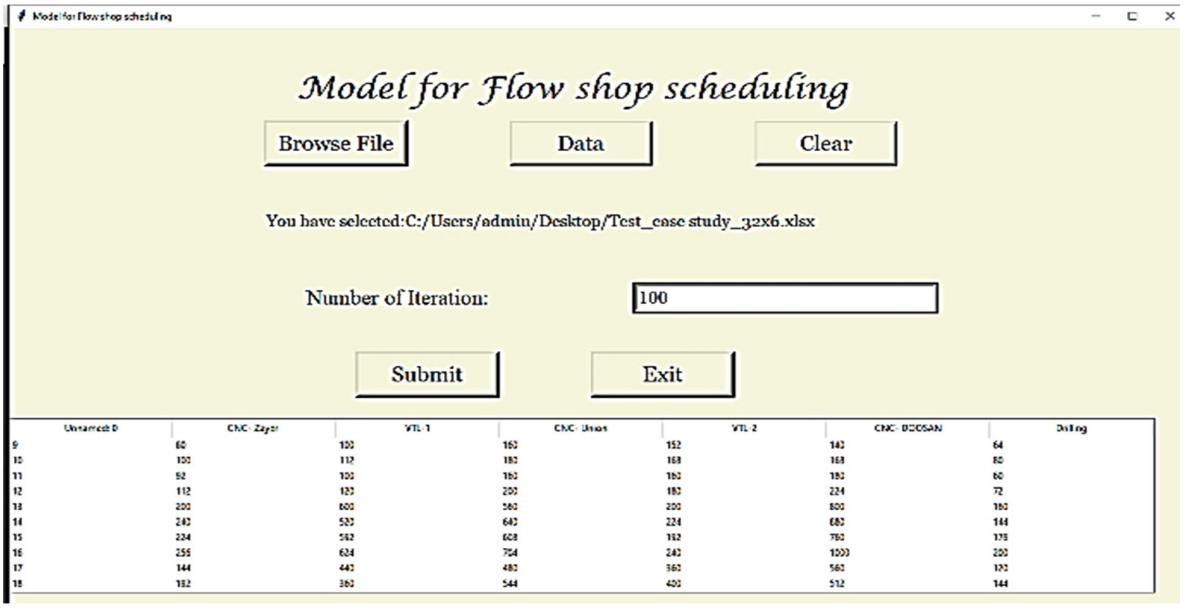


Fig. 6. Data preview

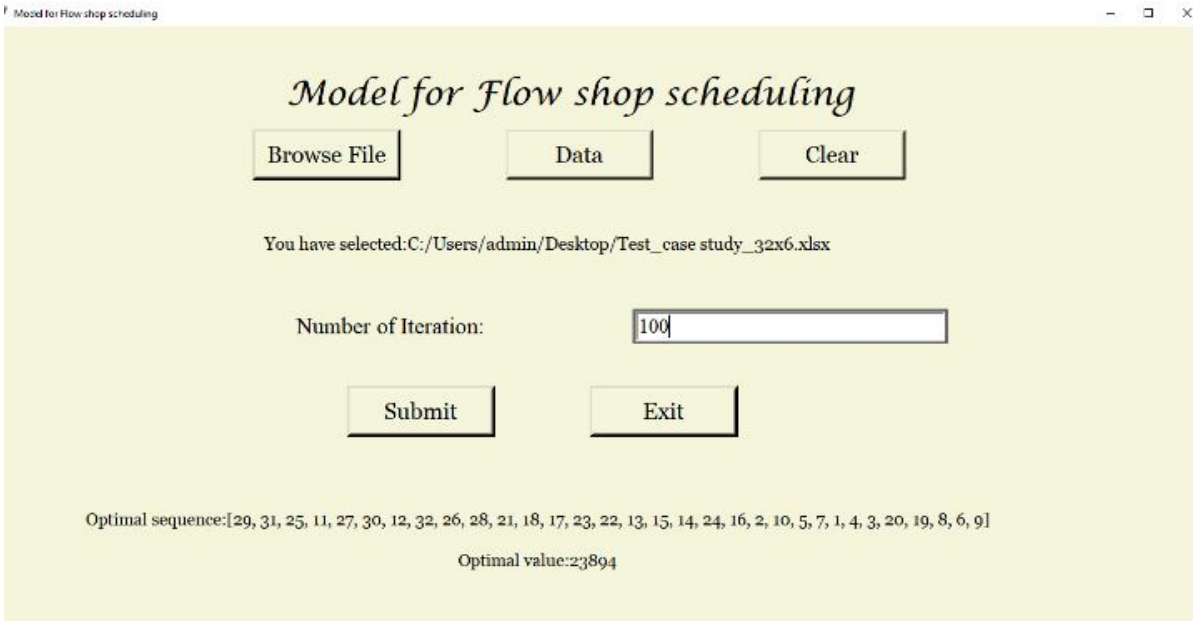


Fig. 7. Output window

Palmer's slope index, CDS, NEH, GA, PSO and the proposed HHEGA algorithms are applied to the company dataset to compute makespan. The makespan for the company's traditional first come, first served job technique is 28536, while the makespan after applying HHEGA is 23894. As a result, the company can save 4642 hours and utilize it for better use. The case study makespan for heuristics, metaheuristics and HHEGA is shown in Fig. 8.

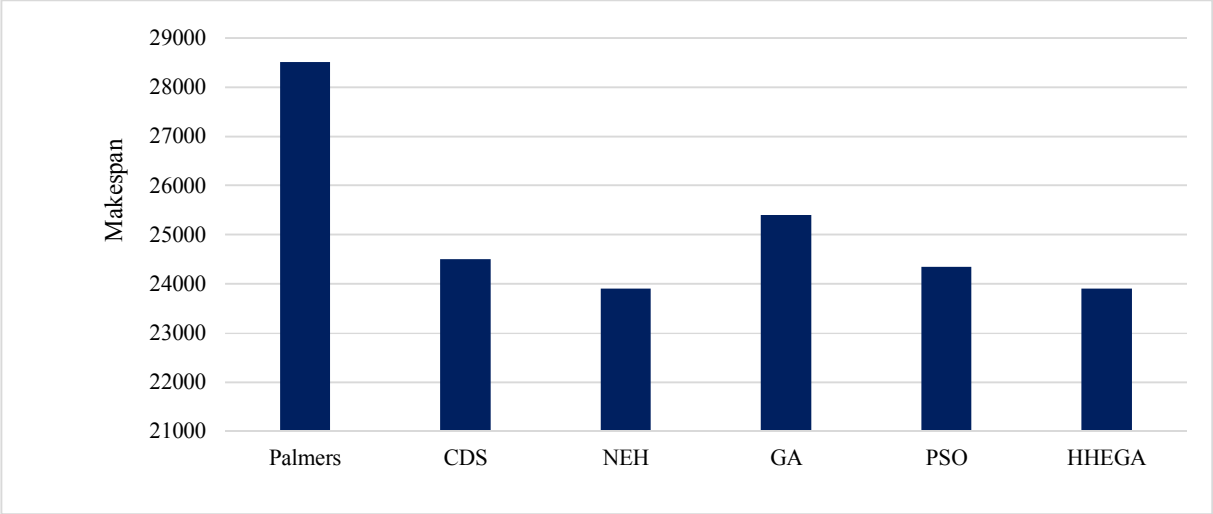


Fig. 8. Makespan for case study

The LB for the company case study is estimated as 23204, and the relative difference is calculated and recorded in Tab. 5 using equation (7) and equation (8) respectively. It can be seen that by finding relative differences from the LB for all algorithms, HHEGA and NEH in attaining the LB was better related to others and it is displayed in Tab. 5 and Fig. 9.

Tab. 5. Relative difference from LB for case study

Dataset name	Palmers	CDS	NEH	GA	PSO	HHEGA
Case Study	18.63955	5.29752	2.88775	8.66005	4.72201	2.88775

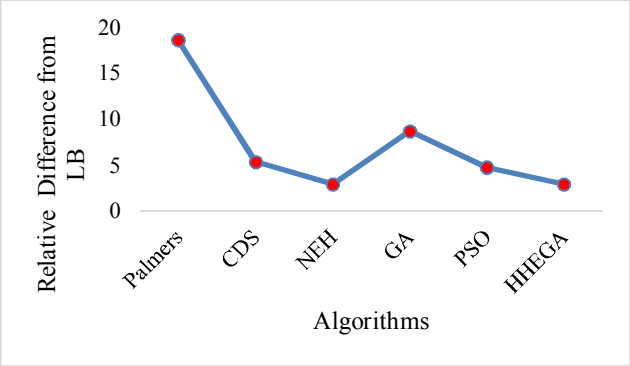


Fig. 9. Case study - relative difference from LB

From hypothesis testing and case study, it can be seen that HHEGA, is a hybrid algorithm that combines the benefits of the NEH, PSO, and GA algorithms. It can be shown that combining heuristics and metaheuristics improves performance and speeds up convergence.

5. Conclusions

The research investigated the minimization of makespan for explaining the flow shop scheduling problem. Here six random instances are generated and tested with a simulation model based on the proposed HHEGA. The steel industry case study has been tested and determined to be acceptable. The solutions of experiments revealed the proficiency and effectiveness of the proposed algorithm. In summary, the optimal solutions for 50% of instances are obtained, and that too in a short amount of period. Additionally, the average relative difference from the LB of the HHEGA is shown to be 2.06086 which is significantly less than that of Palmer's slope index, CDS, NEH, GA and PSO. It is seen that the makespan employing NEH and HHEGA are the same for the case study, however, after applying HHEGA there is an improvement for the three instances Test01_3x4, Test04_ 4x6, and Test05_9x5 taken in this research. The simulation model based on HHEGA can save the company 4642 hours as compared to their traditional approach. In terms of minimizing makespan, an improvement of approximately 16% was noticed in the case study. Thus, it is concluded that the newly proposed HHEGA outperforms other approaches.

Future work on this topic might incorporate other objectives such as tardiness, and it could be explored for open job shop problems, offering the company a wide range of possibilities. Changing the stand-alone design settings for web apps also enables the operator to use it remotely on any device.

References

[1] M. F. G. Charista Elliani, L. Gozali, F. J. Daywin, and C. O. Doaly, "Flowshop scheduling using CDS algorithm, bat algorithm, and tabu search algorithm at PT. Dynaplast Jatake," *Proc. Int. Conf. Ind. Eng. Oper. Manag.*, No. 2006, (2021), pp. 2733-2742.

[2] P. Bari and P. Karande, "Application of PROMETHEE-GAIA method to priority sequencing rules in a dynamic job shop for single machine," *Mater. Today Proc.*, Vol. 46, No. 17, (2021), pp. 7258-7264.

[3] V. S. Tanaev, V. S. Gordon, and Y. M. Shafransky, "Scheduling Theory. Single-Stage Systems," *Springer Science & Business Media,* Vol. 284, (2012).

[4] R. Vanchipura, R. Sridharan, and A. S. Babu, "Improvement of constructive heuristics using variable neighbourhood descent for scheduling a flow shop with sequence dependent setup time," *J. Manuf. Syst.*, Vol. 33, No. 1, (2014), pp. 65-75.

[5] P. Azimi and S. Sholekar, "A simulation optimization approach for the multi-objective multi-mode resource constraint project scheduling problem," *Int. J. Ind. Eng. Prod. Res.*, Vol. 32, No. 1, (2021), pp. 37-45.

[6] A. Baskar and M. A. Xavior, "New idle time-based tie-breaking rules in heuristics for the permutation flowshop scheduling problems," *Comput. Oper. Res.*, Vol. 133, (2021), p. 105348.

[7] J. K. Lenstra, A. H. G. Rinnooy Kan, and P. Brucker, "Complexity of machine scheduling problems," *Ann. Discret. Math.*, Vol. 1, No. C, (1977), pp. 343-362.

[8] V. Fernandez-Viagas and J. M. Framinan, "On insertion tie-breaking rules in heuristics for the permutation flowshop scheduling problem," *Comput. Oper. Res.*, Vol. 45, (2014), pp. 60-67.

[9] J. Berlińska and B. Przybylski, "Scheduling for gathering multitype data with local computations," *Eur. J. Oper. Res.*, Vol. 294, No. 2, (2021), pp. 453-459.

[10] S. M. Johnson, "Optimal two-and three-stage production schedules with setup times included," *Nav. Res. Logist. Q.*, Vol. 1, (1954), pp. 61-68.

[11] A. J. Vakharia and U. Wemmerlov, "Designing a cellular manufacturing system: A materials flow approach based on operation sequences," *IIE Trans. (Institute Ind. Eng.)*, Vol. 22, No. 1, (1990), pp. 84-97.

[12] D. S. Palmer, "Sequencing Jobs Through a Multi-Stage Process in the Minimum Total Time -- A Quick Method of Obtaining a Near Optimum," *OR*, Vol. 16, No. 1, (1965), p. 101.

[13] T. S. Hundal and J. Rajgopal, "An extension of palmer's heuristic for the flow shop scheduling problem," *Int. J. Prod. Res.*, Vol. 26, No. 6, (1988), pp. 1119-1124.

[14] H. G. Campbell, R. A. Dudek, and M. L. Smith, "Heuristic Algorithm for the N Job, M Machine Sequencing Problem," *Manage. Sci.*, Vol. 16, No. 10, (1970), pp. 630-637.

[15] M. Nawaz, E. E. Enscore, and I. Ham, "A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem," *Omega*, Vol. 11, No. 1, (1983), pp. 91-95.

[16] S. F. Rad, R. Ruiz, and N. Boroojerdian, "New high performing heuristics for minimizing makespan in permutation flowshops," *Omega*, Vol. 37, No. 2, (2009), pp. 331-345.

[17] A. Baskar, "Revisiting the NEH algorithm- the power of job insertion technique for optimizing the makespan in permutation flow shop scheduling," *Int. J. Ind. Eng. Comput.*, Vol. 7, No. 2, (2016), pp. 353-366.

[18] P. J. Kalczynski and J. Kamburowski, "An improved NEH heuristic to minimize makespan in permutation flow shops," *Comput. Oper. Res.*, Vol. 35, No. 9, (2008), pp. 3001-3008.

[19] X. P. Li, Y. X. Wang, and C. Wu, "Heuristic algorithms for large flowshop scheduling problems," *Proc. World Congr. Intell. Control Autom.*, Vol. 4, (2004), pp. 2999-3003.

[20] C. Sauvey and N. Sauer, "Two NEH heuristic improvements for flowshop scheduling problem with makespan criterion," *Algorithms*, Vol. 13, No. 5, (2020), pp. 1-14.

[21] V. Fernandez-Viagas, R. Ruiz, and J. M. Framinan, "A new vision of approximate methods for the permutation flowshop to minimise makespan: State-of-the-art and computational evaluation," *Eur. J. Oper. Res.*, Vol. 257, No. 3, (2017), pp. 707-721.

[22] W. Liu, Y. Jin, and M. Price, "A new improved NEH heuristic for permutation flowshop scheduling problems," *Int. J. Prod. Econ.*, Vol. 193, (2017), pp. 21-30.

[23] Y. Wang, X. Li, and Z. Ma, "A hybrid local search algorithm for the sequence dependent setup times flowshop scheduling problem with makespan criterion," *Sustainability (Switzerland),* Vol. 9, No. 12. (2017).

[24] P. J. Kalczynski and J. Kamburowski,

"An empirical analysis of the optimality rate of flow shop heuristics," *Eur. J. Oper. Res.*, Vol. 198, No. 1, (2009), pp. 93-101.

[25] E. A. Gangraj, "Heuristic approach to solve hybrid flow shop scheduling problem with unrelated parallel machines," *Int. J. Ind. Eng. Prod. Res.*, Vol. 28, No. 1, (2017), pp. 61-74.

[26] R. Ruiz and C. Maroto, "A comprehensive review and evaluation of permutation flowshop heuristics," *Eur. J. Oper. Res.*, Vol. 165, No. 2, (2005), pp. 479-494.

[27] G. I. Zobolas, C. D. Tarantilis, and G. Ioannou, "Minimizing makespan in permutation flow shop scheduling problems using a hybrid metaheuristic algorithm," *Comput. Oper. Res.*, Vol. 36, No. 4, (2009), pp. 1249-1267.

[28] R. Ruiz and T. Stützle, "A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem," *Eur. J. Oper. Res.*, Vol. 177, No. 3, (2007), pp. 2033-2049.

[29] D. A. Rossit, F. Tohmé, and M. Frutos, "The Non-Permutation Flow-Shop scheduling problem: A literature review," *Omega (United Kingdom)*, Vol. 77, (2018), pp. 143-153.

[30] P. Bari, P. Karande, and J. Menezes, "Simulation of Job Sequencing for Stochastic Scheduling With a Genetic Algorithm," *Oper. Res. Eng. Sci. Theory Appl.*, Vol. 5, No. 3, (2022), pp. 17-39.

[31] M. Khatami, A. Salehipour, and F. J. Hwang, "Makespan minimization for the m-machine ordered flow shop scheduling problem," *Comput. Oper. Res.*, Vol. 111, (2019), pp. 400-414.

[32] M. Babor, O. Paquet-Durand, R. Kohlus, and B. Hitzmann, "Modeling and optimization of bakery production scheduling to minimize makespan and oven idle time," *Sci. Rep.*, Vol. 13, No. 1, (2023), p. 235.

[33] K. Allali, S. Aqil, and J. Belabid, "Distributed no-wait flow shop problem with sequence dependent setup time: Optimization of makespan and maximum tardiness," *Simul. Model. Pract. Theory*, Vol. 116, No. 2021, (2022), p. 102455.

[34] F. Zhao, H. Bao, L. Wang, T. Xu, N. Zhu, and Jonrinaldi, "A heuristic and meta-heuristic based on problem-specific knowledge for distributed blocking flow-shop scheduling problem with sequence-dependent setup times," *Eng. Appl. Artif. Intell.*, Vol. 116, (2022), p. 105443.

[35] M. S. Umam, M. Mustafid, and S. Suryono, "A hybrid genetic algorithm and tabu search for minimizing makespan in flow shop scheduling problem," *J. King Saud Univ. - Comput. Inf. Sci.*, Vol. 34, No. 9, (2022), pp. 7459-7467.

Follow this article at the following site:

Prasad Bari & Prasad Karande: Makespan minimization using hybrid heuristic metaheuristic genetic algorithm. IJIEPR 2023; 34 (2) :1-16
URL: http://ijiepr.iust.ac.ir/article-1-1713-en.html