# An effective hybrid moth flame evolutionary algorithm for multi-objective energy efficient flexible job shop scheduling problem

Vijaya Kumar Manupati[a], Sasi Kanth[a] , Kush Dilip Acharya[b] , Aditya Vikram Satnalika[b], Goran Putnik[c], M.L.R. Varela[c], A. M. Madureira[d] and Ajith Abraham[e]

[a]*School of Mechanical Engineering, VIT University, Vellore, India*

[b]*School of Computer Science Engineering, VIT University, Vellore, India*

[c]*Department of Production and Systems, University of Minho, 4800-058, Guimarães, Portugal*

[d]*Polytechnic Institute of Porto (ISEP-IPP), Porto, Portugal.*

[e]*Machine Intelligence Research Labs, USA.*

Mobile: +91 9775627564

e-mail: manupativijay@gmail.com, mademsasikanth.reddy2014@vit.ac.in, kushdilip.acharya2015@vit.ac.in, adityavikram.satnalika2015@vit.ac.in leonilde@dps.uminho.pt, putnikgd@dps.uminho.pt, amd@isep.ipp.pt, Machine Intelligence Research Labs, USA, abraham.ajith@gmail.com

* Corresponding author: manupativijay@gmail.com

**Abstract**

Current manufacturing enterprises facing the challenge of increasing energy prices and its associated environmental effects to reduce their emissions. In order to reduce the manufacturing energy consumption, the system-level energy reduction with the operational method can be employed as an energy-saving approach. This paper presents an energy efficient Flexible Job-shop Scheduling Problem (FJSSP) that offer several advantages in the current competitive atmosphere by way to improve the system performance. Therefore, a multi-objective scheduling method is developed in this paper, with reducing total completion time of jobs, processing cost of operations and energy consumption of machines as objectives. Since the problem is strongly NP-hard, a novel algorithm based on an improved Moth Flame Optimization (MFO) is adapted to search for the optimal/near-optimal solution in Flexible Job-shop Scheduling problems (FJSSP). Numerical experiments are conducted with modified ten different instances of FJSSP is presented to show the effectiveness of algorithm and to prove the feasibility of the model. Finally, the performance of the proposed MFO algorithm is compared with Non-Dominated Sorting Genetic Algorithm (NSGA-II). Experimental results show that the proposed MFO algorithm performed better in comparison to NSGA-II.

***Keywords:*** Flexible Job Shop Scheduling, Multi-objective Optimization, Energy Consumption, Makespan, Moth Flame Optimization Algorithm.

## 1. Introduction

In the technically advanced manufacturing environment of today's world, an important aim of current manufacturing companies is to meet the requirements of the rapid increase in energy consumption and its associated environmental affect (Giret, 2016). In order to achieve the cost-effective, environmentally friendly, and efficient manufacturing processes the current manufacturing systems have to be flexible enough and highly efficient. But, due to the increase in the price of unit energy and stringent environmental protection awareness schemes to meet the mentioned requirements is a challenge. However, out of the mentioned objectives, reducing the energy consumption of manufacturing process has become an important issue, thus recently many researchers drew their attention towards it (Jiang, 2014). In this paper, we have considered the production-scheduling problem as a Flexible Job-shop Scheduling Problem (FJSSP) due to its realistic and constructive approach for manufacturing industries (Yang, 2016). Flexible Job-shop Scheduling has several applications in various industries such as factories of plastic injection machines, fastener manufacturing companies, To investigate the above mentioned sustainable parameters in FJSSP, the objective functions such as energy consumption, completion time and processing cost has been considered for the development of a sustainable manufacturing.

FJSSP is an NP-hard problem and it is an extension of classical job-shop scheduling problem (JSSP), it usually deals with the real-time manufacturing situations and it is more complex scheduling problem than JSSP (Jiang, 2014) as the number of jobs increases finding the optimal schedule in short time becomes very difficult. In general, there are two different approaches for solving FJSSP, namely Hierarchical and Integrated approach. In Hierarchical approach, the problem is solved by dividing it into sub-problems and solving them independently (Brandimarte, 1993). But by doing so we may lose the best possible solutions and settle down with a compromising solution for multiple objectives. In addition, the considered problem has multiple objectives such as maximum completion time i.e. makespan and minimum energy consumption, where its actual production scenario is complex as it is difficult to find a compromising solution between two or more conflicting objectives that need to be solved to optimality by scheduling of operations to determine the optimal processing sequence. Thus, it can be well portrayed as multiple objective FJSSP (MOFJSSP). Therefore,

efficient optimization algorithms with outstanding computational effort are required to deal with multiple objective functions.

Taking all the above-mentioned characteristics into account and the requirements of sustainable manufacturing, it is required to merge the considered multi-objectives maximum completion time and minimum energy consumption functions more tightly for the improvement of the manufacturing system performance. In this regard, we focus on responding to the following questions.

1) What kind of mathematical model must be developed with the considered performance measures and how this model can be solved to optimality?
2) What are the effects of the proposed multi-objective evolutionary algorithm i.e., MOO-HMF on the considered FJSSP problem and how these effects influence the considered objectives?
3) How do these three conflicting objectives trade-off?

## 2. Literature review

The discussion in this section will focus on the three identified production scheduling problems i.e., Job-shop Scheduling Problem, Flow Shop Scheduling Problem and Flexible Job-shop Scheduling Problem. The detailed literature on these problems is mentioned as follows.

### *2.1 Job shop scheduling problem (JSSP)*

Fang et al. (1993) proposed a new promising genetic algorithm approach for finding the makespan of job-shop scheduling problem and open-shop scheduling problem. Pezzella et al. (2000) developed a new Tabu search method for the Job-shop scheduling problem which is guided by shifting bottleneck model that considered makespan as the objective to attain the best solution. Goncalves et al. (2005) addressed a scheduling model in a job-shop production system, with proposed hybrid genetic algorithm approach an optimal solution related to makespan has been obtained. Lin et al. (2010) addressed a makespan related multi-objective scheduling model, machine idle time and total tardiness in a job-shop system and achieved most optimal solutions using multi-objective PSO. Liu et al. (2014) developed a scheduling model for the job-shop problem that minimizes the total non-processing energy consumption and total weighted tardiness to acquire the optimal solution. May et al. (2015) developed a multi-objective model considering makespan, energy consumption of a job-shop scheduling problem and obtained a set of various Pareto front solutions using green genetic algorithm. Escamilla et al. (2016) proposed a multi-objective scheduling model for JSSP, involving

makespan and energy consumption as objectives and obtained a set of solutions based on a multi-objective genetic algorithm. Zhang et al. (2016) developed a scheduling model that minimizes the energy requirement and weighted tardiness of the job-shop problem and derived the exact solution by employing the genetic algorithm with an enhanced local search. Yin et al. (2017) formulated a multi-objective optimization model that considers productivity, energy efficiency and noise reduction with flexible spindle speed for a job-shop scheduling problem and proposed a multi-objective genetic algorithm based on simplex lattice design to solve to optimality.

### *2.2 Flow shop scheduling problem (FSSP)*

Tang et al. (2005) suggested a neural network model and proposed an algorithm for the hybrid flow shop scheduling prototype, which addressed the average flow time, average tardy time and percentage of tardy jobs as multi-objective optimization functions that depicted the performance of the neural network approach as superior to the traditional dispatching rules in evaluating the optimum solution. Yagmahan et al. (2008) proposed an ant colony optimization algorithm to minimize makespan, total flow time and total machine idle time of flow shop scheduling problem. Sayadi et al. (2010) proposed a new discrete firefly meta-heuristic model for the flow shop scheduling model that deemed makespan as an objective in order to establish the accurate result. Lionel et al. (2010) formulated a multi-objective scheduling model involving the utilisation rate of the bottleneck and the total completion time for a hybrid flow shop model and used L-NSGA algorithm to obtain an accurate answer. Pan et al. (2011) addressed an IOT streaming flow shop scheduling problem that took into supposition the weighted earliness and tardiness penalties with discrete artificial bee colony (DABC) algorithm to evaluate the performance of the proposed DABC algorithm with the other best-performing algorithms known from the literature. Fang et al. (2011) developed a new mixed integer programming model which considered peak power consumption, the carbon footprint and the makespan of flow shop scheduling model to examine computationally tractable approaches for finding near-optimal schedules. Bruzzone et al. (2012) addressed an energy-aware scheduling model with the help of mixed integer programming formulation that accounts for energy consumption of a flexible flow shop problem, where the initial assignment of jobs and sequencing were required to be kept fixed. Liu et al. (2017) proposed a hybrid multi-objective backtracking search algorithm for permutation flow shop scheduling problem that considered makespan and energy consumption as objective functions to attain makespan related definite

results. Marichelvam et al. (2017) developed a multi-objective scheduling problem to evaluate makespan and total flow time in a flow-shop system and solved it with a hybrid monkey search algorithm.

### *2.3 Flexible job shop scheduling problem (FJSSP)*

In modern production systems of FJSSP, due to its real-life applications, simultaneous optimization of multiple objectives need to be done. Therefore, many researchers have started working on multi-objective optimization of FJSSP (Reddy, et al., Silva, et al., Varela, et al., 2017; Santos, et al., 2015). Gen et al. (2007) built a multi-objective optimization model considering makespan, maximum machine workload and total workload of a flexible job-shop scheduling problem and designed a hybridised genetic algorithm with an innovative local search procedure (bottleneck shifting) to solve it for acquiring accurate results. Madureira et al. (2008) proposed a Multi-Agent Autonomic and Bio-Inspired framework with self-managing capabilities to solve the extended job-shop scheduling problem. Zhang et al. (2009) addressed a multi-objective scheduling model related to completion time, workload of critical machine and total workload of machines in a flexible job-shop system and obtained most optimal results based on hybrid PSO. Gen et al. (2009) employed genetic algorithm with the crossover and mutation methods for flexible job-shop scheduling problem with the objectives of minimizing makespan, total workloads of machines and maximum workloads of machines to obtain accurate solutions. Madureira et al. (2010) developed a BioSched system to support the dynamic and distributed scheduling of extended job-shop scheduling problem. Jun-Qing et al. (2011) developed a Pareto-based discrete artificial bee colony algorithm to elucidate multi-objective flexible job-shop scheduling problems to obtain faultless results. Jiang et al. (2014) developed a multi-objective flexible job-shop scheduling model involving makespan, processing cost, energy consumption and cost-weighted processing quality and solved it with the proposed on-dominant sorting genetic algorithm with the target of accomplishing the best results. Madureira et al. (2014) proposed a negotiation mechanism for the extended job-shop scheduling problem that considers makespan and machine occupation rate as objective functions. Karthikeyan et al. (2015) proposed a hybrid discrete firefly algorithm for the flexible job-shop scheduling problem that considers maximum completion time, the workload of a critical machine and total workload of all machines to finally obtain the required results. Giret et al. (2016) confronted a multi-objective scheduling model related to energy consumption and makespan in a flexible job-shop system and obtained a set of Pareto optimal solutions using improved particle swarm optimization algorithm. Yang et al. (2016) presented a novel method

for the optimization of bi-objective flexible job-shop scheduling model that considers total completion time and total energy consumption under stochastic processing times using NSGA-II algorithm to be applied to the manufacturing industry. Deng et al. (2017) proposed a bee evolutionary guiding non-dominated sorting genetic algorithm II for the multi-objective flexible job-shop scheduling problem that considered the total completion time, workload of the most loaded machine and the total workload of all machines to obtain optimum results by comparing the experimental results and the results of some pre-defined algorithms.

Although much work has been done on Multi-objective based FJSSP to the author's knowledge, till date very few literature are available with the consideration of multiple objectives as energy consumption of machines to process the job's, processing costs of operations and maximum completion time of jobs in the context of FJSSP. Inspired by the complexity of the above problem and its importance known from the literature, the issues related to the multi-objective problem have been taken into account to find the efficient and feasible solutions.

In this paper, taking environmental concerns into account the performance measures such as makespan, processing cost, and energy consumption based on the actual production environment in the context of FJSSP. Thereafter, a multi-objective mathematical model has been developed by considering the above-mentioned objectives. Further, in order to obtain good approximate solutions, we proposed a multi-objective based hybrid Moth Flame Optimization Algorithm (MOO-HMF). Numerical experiments are conducted with various complex scenarios having different job complexities to assess the performance of the proposed hybrid multi-objective algorithm. With Non-Dominated Sorting Genetic Algorithm (NSGA-II) we try to compare the proposed algorithm effectiveness and efficiency in order to obtain good approximate solutions. From the results, the performance of the proposed evolutionary algorithm is evaluated and found the feasible and efficient scheduling plans for the proposed FJSSP. In later sections the problem description, proposed framework, and its experimentation have been detailed.

In section 3, the detailed description of the problem and its basic assumptions with a developed mathematical model along with the constraints. In section 4, we presented a framework of the proposed MOO-HMF algorithm. In section 5, numerical experiments with different instances having various complex scenarios are explained and their respective results are discussed in section 6. The paper concludes with section 7.

## 3. Problem description

We addressed a flexible job shop scheduling problem (FJSSP) and it is defined as a set of $n$ jobs with $v_i$ number of operations where $v_i$ varies between ($O_{j1,}$ $O_{j2}$, ..., $O_{jl}$). Each operation $O_{jl}$ of job $j$ is to be processed on one machine $k$ from a set of eligible machines $m$. We assume that the processing time of the operations is known and all the machines are available at time zero i.e. before the scheduling of operations. Each machine can process only one operation at a time, and the consecutive jobs can wait at buffers until its preceding job finishes its process. Here, we have considered the buffer sizes are unlimited. To assess the performance of the system, most frequently considered performance measure such as the minimization of makespan for the total completion time of the manufacturing processes is considered. Thereafter, the sustainability parameters such as power and time have been considered due to the fact that as the working speed of a machine increase, the energy consumption also increases despite the shorter processing time. Therefore, the performance measure i.e. minimization of energy consumption has been considered as a second objective function. In addition to the above performance measures, minimization of total processing cost is considered as the third objective function which is also an important factor to be considered in scheduling. As the problem is a multi-objective in nature a mathematical model has been developed with the consideration of the above mentioned three objectives. The above-discussed problem makes several assumptions that are noteworthy to mention.

### *3.1. Assumptions*

(a) The considered machines before scheduling must be available at time zero.

(b) All products can be started at time zero.

(c) At a time it is possible to process only one operation on one machine.

(d) Processing of operations on the machines should not be interrupted.

(e) The sequence of operations of each job for further processing has to be pre-defined.

(f) Release times and due dates are not specified.

(g) Job transportation time among machines is not considered.

Based on the above-mentioned problem and considering its assumptions, the proposed model and its mathematical model is shown below and their notations are explained in Table 1.

**Table 1**. Notations used in mathematical model

| | |
|---|---|
| $N$ | Total number of jobs. |
| $M$ | Total number of machines. |
| $v_i$ | Number of operations of job $j$ |
| $C_{jo}^{k}$ | Completion time of $o^{th}$ operation of job $j$ on machine $k$. |
| $T_{jo}^{k}$ | Duration of $o^{th}$ operation of job $j$ on machine $k$. |
| $S_{jo}^{k}$ | Beginning time of $o^{th}$ operation of job $j$ on machine $k$. |
| $mc_j$ | Denotes raw material cost of job $j$. |
| $pc_k$ | Denotes process cost of machine $k$ per unit time. |
| $e_k$ | Denotes energy consumption of machine $k$ per unit time. |

*Objectives:*

Targeting this problem, minimization of total completion time, total processing cost minimization, and minimization of energy consumption are the three objectives considered during simulation-based optimization. These objectives can be formulated using the following equations:

Minimization of Total completion time $$T = \text{Max}\,(T_{jo}^{k}) \quad (1)$$

Total processing cost minimization $$TPC = \sum_{j=1}^{n} mc_j + \sum_{j=1}^{n}\sum_{o=1}^{v_i} O_{jo}^{k} T_{jo}^{k} pc_k \quad (2)$$

Minimization of energy consumption $$E = \sum_{j=1}^{n}\sum_{o=1}^{v_i} O_{jo}^{k} T_{jo}^{k} e_k \quad (3)$$

*Subject to constraints:*

$$T_{jo}^{k} \geq T_{j(o-1)}^{k} \quad (4)$$

$$T_{jo}^{k} \geq T_{(j-1)o}^{k} \quad (5)$$

$$T_{jo}^{k} \geq 0 \quad (6)$$

The objective of this problem is to mainly focus on scheduling of the jobs so as to minimize the maximum of their total completion time, i.e., makespan as mentioned in equation (1); to minimize the Total processing cost of operations which is mentioned in equation (2); and to minimize the Energy consumption of machines which is mentioned in equation (3). The constraints are in the equation (4 -6). Equation (4) and (5) ensures that the precedence constraints between the operations of a job and jobs are not violated i.e., $o^{th}$ operation cannot

be started unless $(o\text{-}1)^{th}$ operation is completed and $j^{th}$ job cannot be processed before the previous job is completed. Equation (6) indicates processing constraints, which the preceding constraints among operations of the same job should follow so that each machine is available to other operations only if the concerned operations are complete.

## 4. The NSGA-II and Proposed multi-objective hybrid moth flame evolutionary algorithm (MOO-HMFA)

Recent years due to the advancements in information and communication technology the manufacturing sector gain its advantage to fulfil the customized customer orders in less time. These advances make the system and process more complex. They also make the scheduling problem NP-hard in nature and hence it cannot be solved in a polynomial amount of time. Thus, there is a need for new optimization techniques more than before. Although, many algorithms available in the market, the nature-inspired evolutionary algorithms to gain their advantages due to their merits by obtaining the optimal/near optimal solutions in computationally reasonable time. Of course, according to no free lunch theorem, no algorithm solves better for all optimization problems (Wolpert, 1997). Due to their demerits, it is necessary to have a right balance of the operators in the algorithm that can generate a very accurate approximation of the global optimum.

### 4.1 NSGA-II

The problem considered in this paper is first solved by applying the non-dominated sorting genetic algorithm-II (NSGA-II) developed by Deb et al. (2002a) and then we propose the recently emerged modified Moth Flame Optimization (MFO) algorithm (2015). The above mentioned two meta-heuristics are adapted to generate the near-optimal process plans for three complex examples in the context of Flexible Job-shop Scheduling problem.

#### *4.1.1. Initial Population Generation*

This method is carried out by generating the initial population for a given population size in a random order. Given the defined problem, randomly a job is nominated and verified for its predecessor constraints. If the predecessor constraint is satisfied, then it is added to the chromosome, otherwise a new job is randomly nominated and verified for its predecessor constraint. Until all the chromosomes are completely occupied with the jobs, the process will continue. Fig 1 clearly depicts encoding scheme of the chromosome. The first array depicts the

operations which will be performed by the machines; here fourteen different operations have been randomly allocated. Likewise, the second array depicts the machines that have been utilized for respective operations in the first array. The next two arrays depict the processing cost and the energy consumption of machines in accordance with the corresponding operations.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $O_{41}$ | $o_{21}$ | $o_{42}$ | $o_{23}$ | $o_{43}$ | $o_{31}$ | $o_{33}$ | $o_{22}$ | $o_{11}$ | $o_{13}$ | $o_{44}$ | $o_{12}$ | $o_{32}$ | $o_{14}$ |
| $m_2$ | $m_3$ | $m_3$ | $m_3$ | $m_2$ | $m_2$ | $m_2$ | $m_2$ | $m_4$ | $m_3$ | $m_2$ | $m_4$ | $m_3$ | $m_3$ |
| $pc_2$ | $pc_3$ | $pc_3$ | $pc_3$ | $pc_2$ | $pc_2$ | $pc_2$ | $pc_2$ | $pc_4$ | $pc_3$ | $pc_2$ | $pc_4$ | $pc_3$ | $pc_3$ |
| $e_2$ | $e_3$ | $e_3$ | $e_3$ | $e_2$ | $e_2$ | $e_2$ | $e_2$ | $e_4$ | $e_3$ | $e_2$ | $e_4$ | $e_3$ | $e_3$ |

**Figure 1.** Encoding scheme of chromosome for NSGA-II.

*4.1.2. Evolutionary operators*

Newly developed solutions are carried over to operators namely mutation and crossover for generating a child population of size N. This will preserve the diversity in the population N. Diversification is achieved with the help of crossover operator. Moreover, it also helps to search in the unevaluated solution space so as to generate a solution which will be suggestively different from previously determined solutions. An appropriate value for probability of crossover plays a vital role so as to make sure that the performance of the proposed algorithm is maintained. Hence, a one-cut point order-based crossover operator Davis (1991) with a probability of crossover (*Pc*) as 0.6 is considered. The mutation operator plays a vital role in safeguarding stronger individuals from diversification and to adjust the weaker ones.

A new type of population of size *2N* is identified by merging the parent and child population each of which has a size of N. This is to make sure that there is elitism which produces a huge diversity of population, so as to attain better convergence. The new size of population which is *2N* now goes through non-dominated sorting procedure which divides the set of individuals into different non-dominated regions named non-dominated fronts. In the field of non-dominated sorting procedure and usage of diversity preservation operators an extension has been proposed and detailed Deb et al. (2002).

**4.2. Hybrid Moth Flame Optimization (MFO) Algorithm**

In this section, we adopted a well-known bio-inspired evolutionary algorithm i.e., moth flame optimization (MFO) algorithm to find the best possible solutions of the considered problem. The adopted algorithm has several applications in various diverse fields, but in this paper, we

have used the algorithm and then proposed a modified version of Moth Flame Algorithm that suits the need of the considered problem which covers all the possible solution set to give an optimized result.

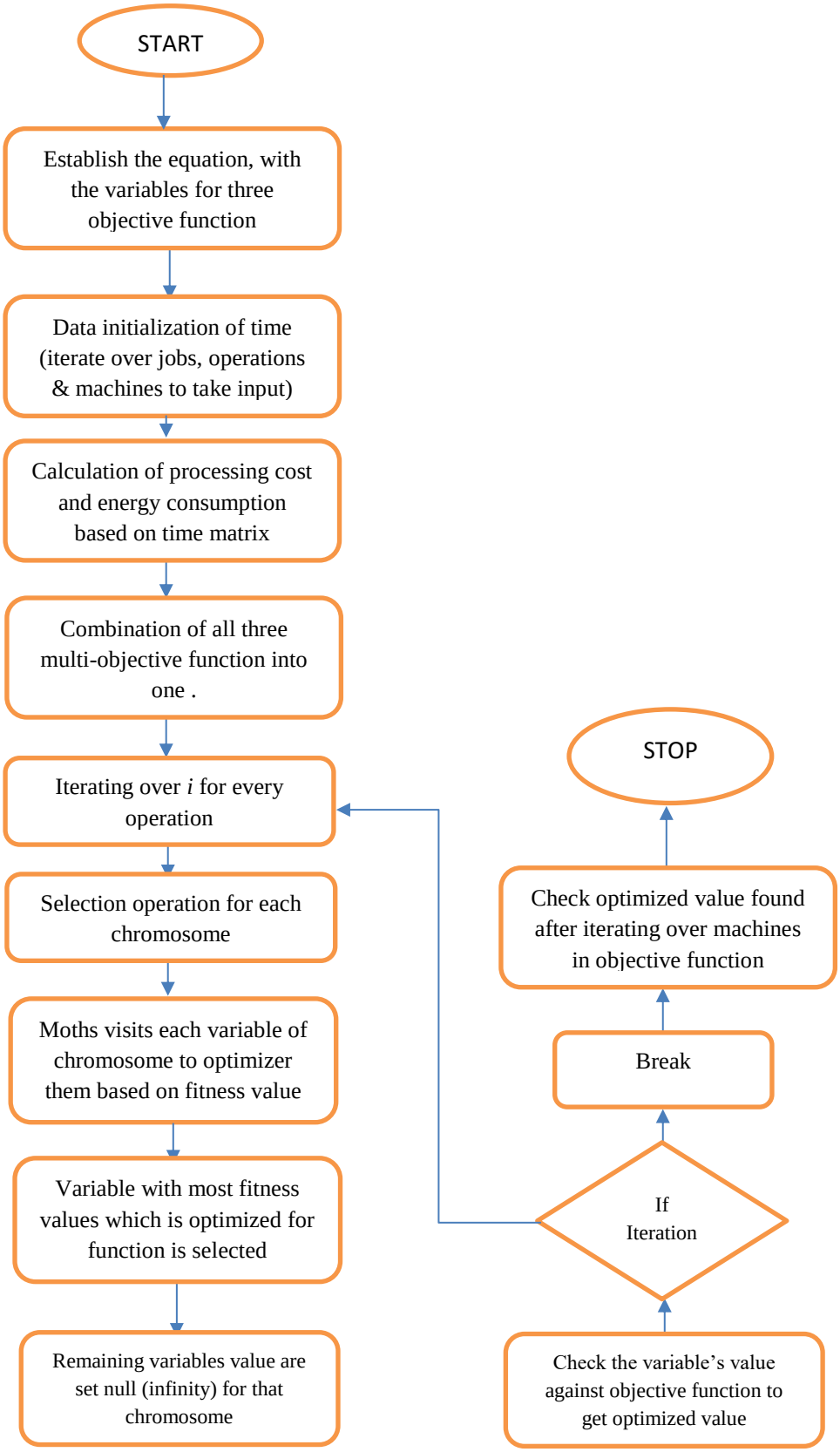



**Figure 2.** Flowchart of the proposed MOO-HMF.

In Fig.2 the flowchart of the proposed MOO-HMFO is depicted where the moths in the algorithm are defined as possible solutions and their position in the space are considered as variables of the problem. The basic assumption for performing the algorithm is, the moths can fly in all dimensions together and individually or even in hyper dimensional space with changing their position vectors. The detail description of the proposed algorithm and its step wise procedure is as follows:

**Step 1** Initially we define the space for moths to explore in the form of time matrices and their related inputs. An encoding scheme for the initialisation of processing time in the form of a chromosome is shown in Fig 3.

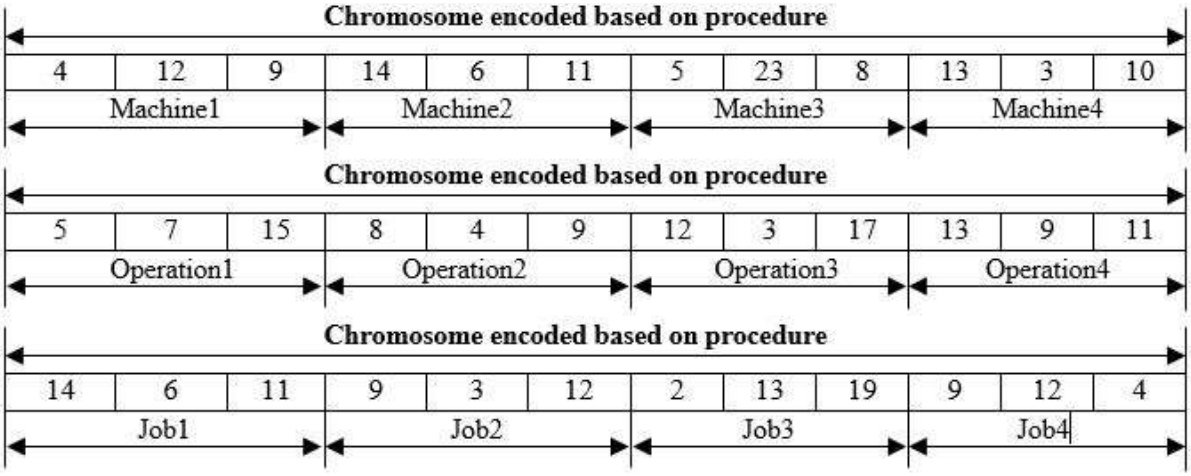



**Figure 3.** Encoding scheme of chromosome for initialisation.

**Step 2** Next, we multiply the time matrices with our processing cost inputs and energy consumption inputs to get our remaining search spaces as energy and processing cost.

$$P\_cost\ (j, k, i) = ProcessingCost(k) * time(j, k, i) \quad (7)$$

$$Energy(j, k, i) = Energy\_consumption(k) * time(j, k, i) \quad (8)$$

Equation (7) and (8) represent processing cost of jobs and energy consumption of machines. The procedure for the calculation of processing cost and energy consumption matrix is shown in Algorithm 2.

**Step 3** We then convert time and energy matrices in terms of money by multiplying it by wages and cost per unit energy and add all three matrices along with their weights. This mutation is applied so as to dimensionally match the matrices of time, cost and energy.

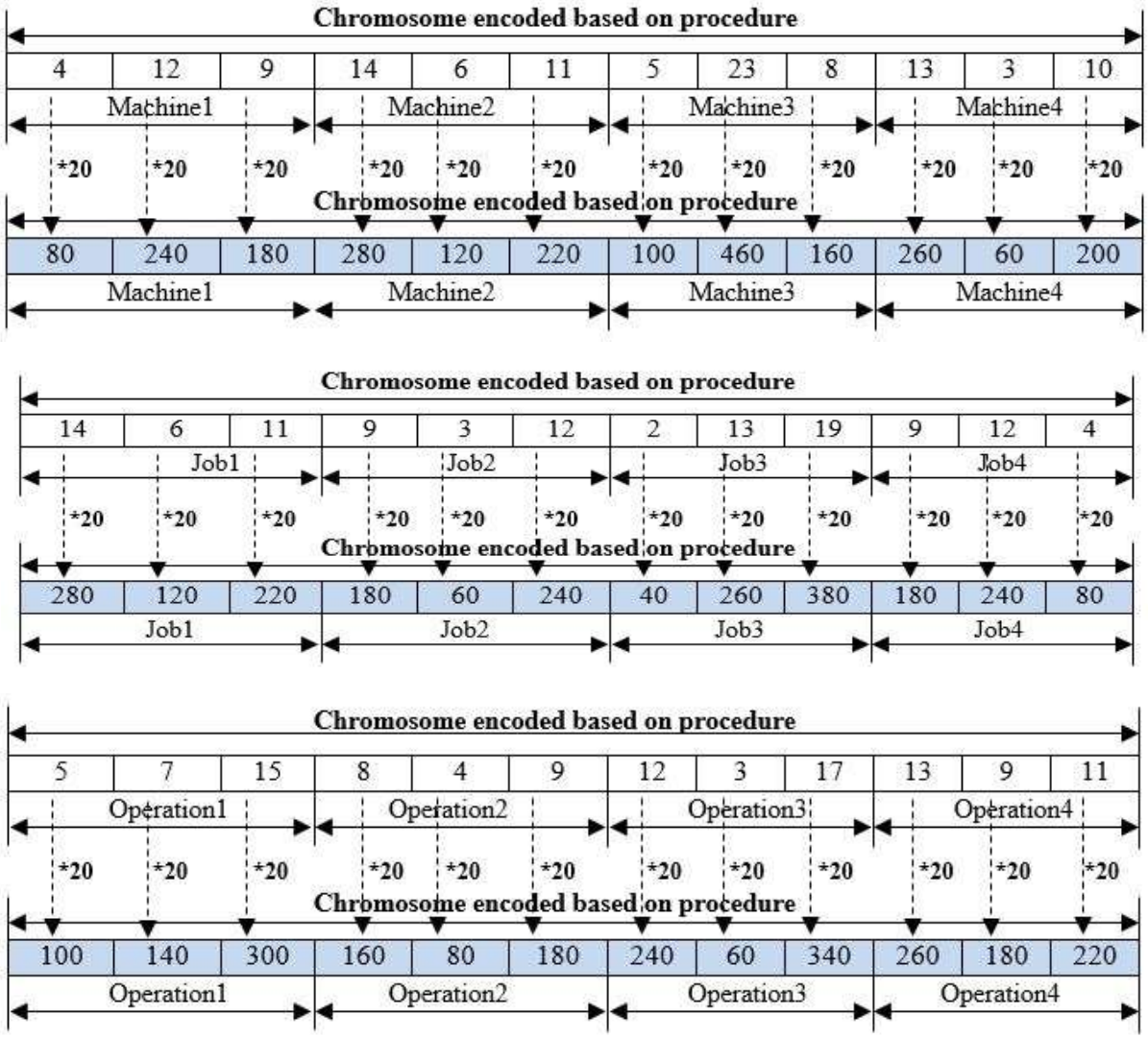
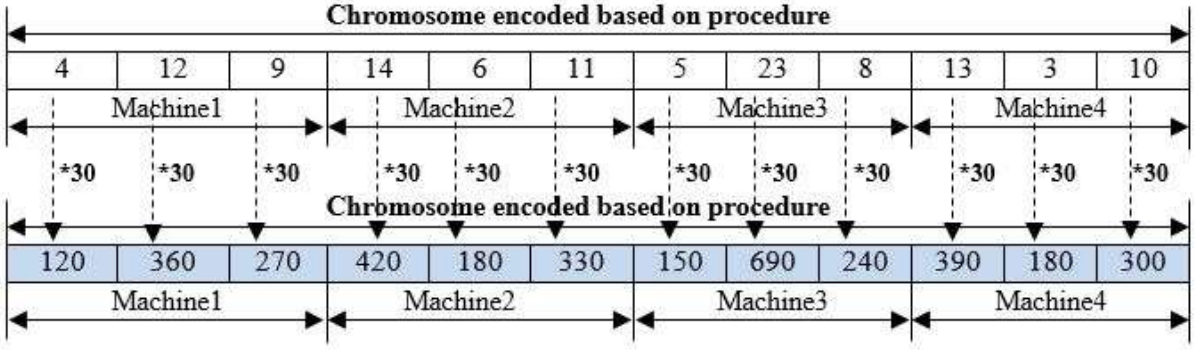



**Figure 4.** Proposed mutation by considering operator wages.



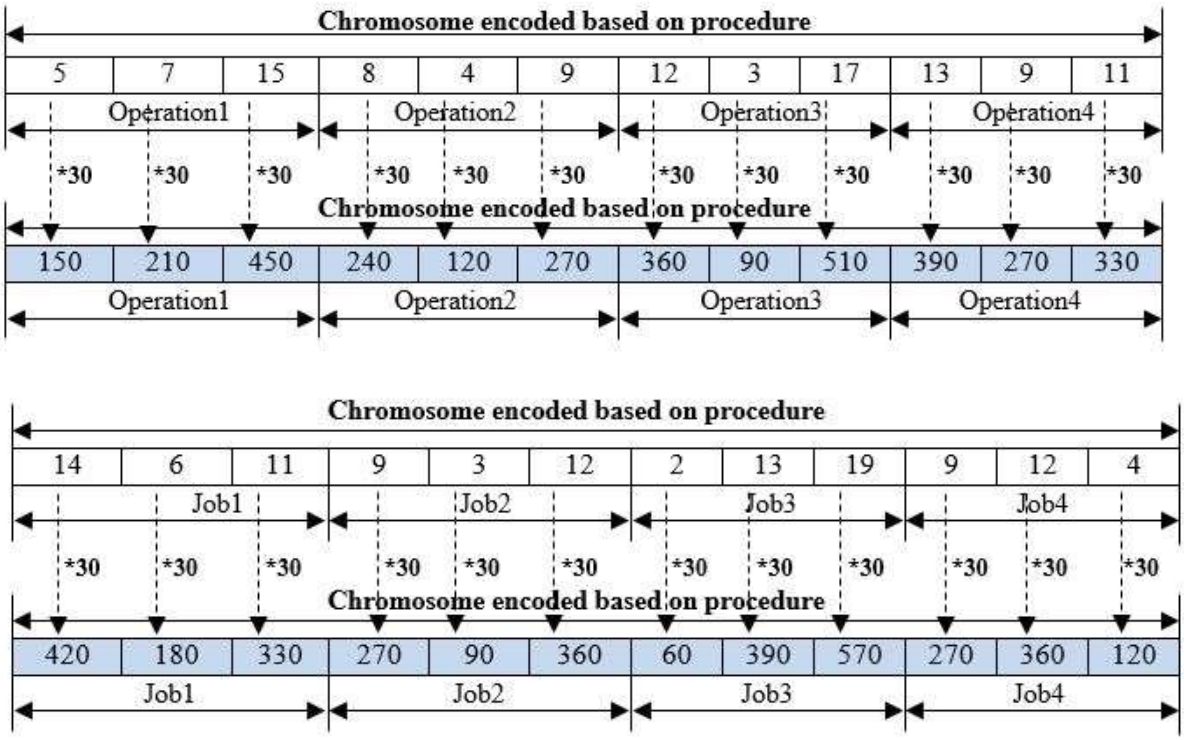



**Figure 5.** Proposed mutation by considering unit cost of energy.

A different type of mutation is proposed in this paper and this is shown in Fig 4 and 5. This was done because of the difference in dimensions of the matrices that are added to form a matrix with minimum values. Hence, we multiplied the energy by its unit cost and time with operator wages.

**Step 4** All the rows are considered as individual light sources in which the minimum value is to be found.

**Step 5** We then start exploring the matrices (search spaces) to find the minimum of the sum so as to obtain a Gantt chart and analyse the results.

**Step 6** After the inputs are received and our search area has been properly defined, we then start exploring the matrices row by row in search for the minimum entry in their respective rows.

**Step 7** For the same we finally find the sum of all resultant matrices column-wise to gain a single-valued function after converting all ∞'s to zeros.

**Step 8:** Checking the single-valued function is optimized or not.

Based on the fitness values we select the highest probable solution for our problem. Hence, in the example, as shown in Fig 3, we can see that second chromosome has best fitness value and hence we are selecting them.

## 5. Experimental Evaluation

In this section, the performance of the proposed algorithm is assessed by considering different modified FJSSP benchmark instances incorporating processing costs and energy consumption profiles for a different set of jobs and machines. First, an FJSSP is proposed and according to our requirements, we constructed ten FJSSP benchmarks. These constructed benchmarks are based on the instances proposed by Brandimarte (1993), Fisher and Thompson (1963) and Lawrence (1984) and it is shown in Algorithm 2. Thereafter, processing cost matrix is generated by multiplying the unit processing cost matrix with processing time matrix. Similarly, the energy consumption matrix is formed by multiplying the unit energy consumption matrix with the processing time matrix. It can be inferred from these matrices that, with a decrease in processing time, processing cost decreases and the energy consumption also decreases.

The proposed algorithm has been coded in MATLAB software and data is provided through input. The computation for the above data was conducted on an HP Notebook (522TX) with Intel Core i5 -6200U CPU (2.30GHz, 3MB L3 Cache) running under Windows 10 Professional as Operating System with 8 GB RAM. All of the instances considered to determine the results were benchmark instances whose references in mentioned in the paper. The number of iterations for each instance are: 10X6X6, 15X10X10, 20X5X5, 10X10X10, 10X5X5, 10X9X9, 6X6X6, and 15X5X5 (number of jobs X number of machines X number of operations).

In each instance, there are *n* jobs and *m* machines where each job has a different number of operations having pre-defined operating times. Unspecified operating time implies that the concerned operation cannot be processed on that machine. The operation times on the machines are kept different. For example, in Table 2 the 10 by 6 data from Brandimarte instances (Brandimarte, 1993) is shown where 10 jobs and 6 machines indicate each job has a maximum of 6 operations, at a time only one job can be processed on one machine. The operational time of different operations for jobs on different machines, the processing cost and energy consumption of respective machines are also given.

**Table 2.** Modified instance "mk1" by Brandimarte (1993)

| $J$ | $O_{jo}$ | $M_1$ | $M_2$ | $M_3$ | $M_4$ | $M_5$ | $M_6$ | $Mc_j$ |
|---|---|---|---|---|---|---|---|---|
| $J_1$ | $O_{11}$ | 5 | - | 4 | - | - | - | 120 |
| | $O_{12}$ | - | 1 | 5 | - | 3 | - | |
| | $O_{13}$ | - | - | 4 | - | - | 2 | |
| | $O_{14}$ | 1 | 6 | - | - | - | 5 | |
| | $O_{15}$ | - | - | 1 | - | - | - | |
| | $O_{16}$ | - | - | 6 | 3 | - | 6 | |
| $J_2$ | $O_{21}$ | - | 6 | - | - | - | - | 100 |
| | $O_{22}$ | - | - | 1 | - | - | - | |
| | $O_{23}$ | 2 | - | - | - | - | - | |
| | $O_{24}$ | - | 6 | - | 6 | - | - | |
| | $O_{25}$ | 1 | - | - | - | - | 5 | |
| $J_3$ | $O_{31}$ | - | 6 | - | - | - | - | 65 |
| | $O_{32}$ | - | - | 4 | - | - | 2 | |
| | $O_{33}$ | 1 | 6 | - | - | - | 5 | |
| | $O_{34}$ | - | 6 | 4 | - | - | 6 | |
| | $O_{35}$ | 1 | - | - | - | 5 | - | |
| $J_4$ | $O_{41}$ | 1 | 6 | - | - | - | 5 | 100 |
| | $O_{42}$ | - | 6 | - | - | - | - | |
| | $O_{43}$ | - | - | 1 | - | - | - | |
| | $O_{44}$ | - | 1 | 5 | - | 3 | - | |
| | $O_{45}$ | - | - | 4 | - | - | 2 | |
| $J_5$ | $O_{51}$ | - | 1 | 5 | - | 3 | - | 150 |
| | $O_{52}$ | 1 | 6 | - | - | - | 5 | |
| | $O_{53}$ | - | 6 | - | - | - | - | |
| | $O_{54}$ | 5 | - | 4 | - | - | - | |
| | $O_{55}$ | - | 6 | - | 6 | - | - | |
| | $O_{56}$ | - | 6 | 4 | - | - | 6 | |
| $J_6$ | $O_{61}$ | - | - | 4 | - | - | 2 | 85 |
| | $O_{62}$ | 2 | - | - | - | - | - | |
| | $O_{63}$ | - | 6 | 4 | - | - | - | |
| | $O_{64}$ | - | 6 | - | - | - | - | |
| | $O_{65}$ | 1 | 6 | - | - | - | 5 | |
| | $O_{66}$ | 3 | - | - | 2 | - | - | |
| $J_7$ | $O_{71}$ | - | - | - | - | - | 1 | 120 |
| | $O_{72}$ | 3 | - | - | 2 | - | - | |
| | $O_{73}$ | - | 6 | 4 | - | - | 6 | |
| | $O_{74}$ | 6 | 6 | - | - | 1 | - | |
| | $O_{75}$ | - | - | 1 | - | - | - | |
| $J_8$ | $O_{81}$ | - | - | 4 | - | - | 2 | 100 |
| | $O_{82}$ | - | 6 | 4 | - | - | 6 | |
| | $O_{83}$ | 1 | 6 | - | - | - | 5 | |
| | $O_{84}$ | - | 6 | - | - | - | - | |
| | $O_{85}$ | - | 6 | - | 6 | - | - | |
| $J_9$ | $O_{91}$ | - | - | - | - | - | 1 | 65 |
| | $O_{92}$ | 1 | - | - | - | 5 | - | |
| | $O_{93}$ | - | - | 6 | 3 | - | 6 | |
| | $O_{94}$ | 2 | - | - | - | - | - | |
| | $O_{95}$ | - | 6 | 4 | - | - | 6 | |
| | $O_{96}$ | - | 6 | - | 6 | - | - | |
| $J_{10}$ | $O_{101}$ | - | - | 4 | - | - | 2 | 100 |
| | $O_{102}$ | - | 6 | 4 | - | - | 6 | |
| | $O_{103}$ | - | 1 | 5 | - | 3 | - | |
| | $O_{104}$ | - | - | - | - | - | 1 | |
| | $O_{105}$ | - | 6 | - | 6 | - | - | |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $O_{106}$ | 3 | - | - | 2 | - | - | |
| | $pc_k$ | 6 | 8 | 7 | 4 | 5 | 6 | |
| | $e_k$ | 8 | 10 | 7.5 | 12 | 9 | 10.5 | |

Although various optimization algorithms are available to solve this problem, we considered a newly developed Moth Flame Optimization algorithm (Mirjalili, 2015) for solving the above-described problem. The proposed algorithm has been coded in MATLAB software and data is provided through input. The computation for the above data was conducted on a PC with Intel Core i5 -6200U CPU (2.30GHz, 3MB L3 Cache) running under Windows 10 Professional Operating System with 8 GB RAM. The operations of jobs that are assigned to the machines are in an order to get the minimum values of performance measures i.e. makespan, processing cost, and energy consumption.

The above procedure is repeated for all the considered instances to find the robustness of the proposed algorithm. In Table 4, the first column indicates different instances, the second column represents a number of jobs and the respective number of machines are represented in column 3, makespan, processing cost, and energy consumption are represented in columns 4,5, and 6 respectively. The results of the proposed MOEA based HMFO algorithm is compared with most popular meta-heuristic NSGA-II.

## 6. Results and Discussion

The performance of the proposed MOO-HMFO algorithm is tested on different benchmark instances with objective functions as minimization of makespan, processing cost and energy consumption. This modified MOO-HMFO Algorithm helps us to find and determine what must be the best sequence of the jobs that should be scheduled on the given machines for a particular job to meet the criterion of the objective function. This helps to maximize profits without any resource or time wastage.

**Table 3.** The results of Instances with different iteration number

| Instance | Iteration number | Best fitness | Worst fitness | Average fitness | Average deviation |
|---|---|---|---|---|---|
| mk1 | 100 | 389.5 | 61.35 | 225.425 | 72.78474 |
| | 300 | 985.5 | 155.4 | 570.45 | 72.75835 |
| | 500 | 1906.5 | 299.1 | 1102.8 | 72.87813 |
| | 700 | 1906.5 | 299.1 | 1102.8 | 72.87813 |
| | 1000 | 1906.5 | 299.1 | 1102.8 | 72.87813 |
| mk2 | 100 | 834 | 131.3 | 482.65 | 72.79602 |
| | 300 | 2153.5 | 338.85 | 1246.175 | 72.80879 |
| | 360 | 2836 | 447 | 1641.5 | 72.76881 |
| | 700 | 2836 | 447 | 1641.5 | 72.76881 |
| | 1000 | 2836 | 447 | 1641.5 | 72.76881 |
| mk3 | 100 | 605.5 | 96.6 | 351.05 | 72.48255 |
| | 300 | 1689.5 | 267.1 | 978.3 | 72.69754 |
| | 500 | 2608.5 | 410.2 | 1509.35 | 72.82274 |
| | 1000 | 5323 | 834.85 | 3078.925 | 72.88502 |
| | 1500 | 7223 | 1130.7 | 4176.85 | 72.92936 |
| mk5 | 100 | 257.5 | 40.25 | 148.875 | 72.96390 |
| | 300 | 515.5 | 81.05 | 298.275 | 72.82709 |
| | 500 | 791 | 126.2 | 458.6 | 72.48147 |
| | 810 | 2005 | 316.2 | 1160.6 | 72.75547 |
| | 1000 | 2005 | 316.2 | 1160.6 | 72.75547 |
| mk7 | 100 | 2357 | 374.1 | 1365.55 | 72.60445 |
| | 300 | 7320.5 | 1166.7 | 4243.6 | 72.50683 |
| | 500 | 10635.5 | 1691.7 | 6163.6 | 72.55338 |
| | 700 | 10635.5 | 1691.7 | 6163.6 | 72.55338 |
| | 1000 | 10635.5 | 1691.7 | 6163.6 | 72.55338 |
| mt06 | 100 | 808.5 | 127 | 467.75 | 72.84874 |
| | 216 | 1400.5 | 220.15 | 810.325 | 72.83189 |
| | 500 | 1400.5 | 220.15 | 810.325 | 72.83189 |
| | 700 | 1400.5 | 220.15 | 810.325 | 72.83189 |
| | 1000 | 1400.5 | 220.15 | 810.325 | 72.83189 |
| mt10 | 100 | 6629 | 1046.5 | 3837.75 | 72.73142 |
| | 300 | 9255 | 1474.7 | 5364.85 | 72.51181 |
| | 500 | 17081 | 2699.55 | 9890.275 | 72.70501 |
| | 700 | 19101.5 | 3004.4 | 11052.95 | 72.81812 |
| | 1000 | 32048 | 5031.6 | 18539.8 | 72.86055 |
| mt20 | 100 | 9991 | 1572.15 | 5781.575 | 72.80758 |
| | 300 | 28001.5 | 4415.65 | 16208.575 | 72.75732 |
| | 500 | 47799 | 7505.5 | 27652.25 | 72.85754 |
| | 700 | 47799 | 7505.5 | 27652.25 | 72.85754 |
| | 1000 | 47799 | 7505.5 | 27652.25 | 72.85754 |
| la01 | 100 | 15100.5 | 2392.55 | 8746.525 | 72.64571 |
| | 250 | 21651 | 3421.75 | 12536.375 | 72.70543 |

| | | | | | |
|---|---|---|---|---|---|
| | 500 | 21651 | 3421.75 | 12536.375 | 72.70543 |
| | 700 | 21651 | 3421.75 | 12536.375 | 72.70543 |
| | 1000 | 21651 | 3421.75 | 12536.375 | 72.70543 |
| la06 | 100 | 9466 | 1480.75 | 5473.375 | 72.94631 |
| | 375 | 41704 | 6656.25 | 24180.125 | 72.47223 |
| | 500 | 41704 | 6656.25 | 24180.125 | 72.47223 |
| | 700 | 41704 | 6656.25 | 24180.125 | 72.47223 |
| | 1000 | 41704 | 6656.25 | 24180.125 | 72.47223 |

In Table 3, the comparative analysis of different fitness values of all instances was investigated under the same algorithm parameters. The results of best, worst and average fitness values for each iteration and for all instances are shown in Table 3. It also shows the average deviation value, which is a measure of the deviation of best/worst fitness from the average fitness value. We can observe that after a certain number of iterations few values are becoming constant in other words the termination criteria have achieved after a significant number of iterations indicated by constant fitness values. For example, consider the first instance i.e., mk1 after five hundred iterations the best, worst and average fitness values are same indicating that the number of iterations required for getting optimal values is reached. One can observe from the instances that the number of iterations required for obtaining optimum values is different. This is the reason why the time for finding optimized value differs for most of the instances.

**Table 4.** Optimal values of makespan, processing cost, and energy consumption of MOO-HMFO and NSGA-II

| **Instances** | **Jobs** | **Machines** | **Proposed HMFO** | | | **NSGA-II** | | |
|---|---|---|---|---|---|---|---|---|
| | | | Makespan | Processing cost | Energy consumption | Makespan | Processing cost | Energy consumption |
| mk1 | 10 | 6 | 113 | 700 | 1093.5 | 124 | 714 | 1124 |
| mk2 | 10 | 6 | 172 | 1032 | 1632 | 190 | 1068 | 1658 |
| mk3 | 15 | 8 | 425 | 2623 | 4185 | 462 | 2654 | 4203 |
| mk5 | 15 | 4 | 121 | 741 | 1143 | 138 | 762 | 1168 |
| mk7 | 20 | 5 | 660 | 3983 | 5992.5 | 669 | 4013 | 6023 |
| mt06 | 6 | 6 | 84 | 510 | 806.5 | 102 | 533 | 849 |
| mt10 | 10 | 10 | 2092 | 9340 | 20616 | 3018 | 9402 | 20645 |
| mt20 | 20 | 5 | 2914 | 16630 | 28255 | 3128 | 16734 | 28288 |
| la01 | 10 | 5 | 1319 | 7986 | 12346 | 1350 | 8034 | 12386 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| la06 | 15 | 5 | 1239 | 33610 | 6855 | 1298 | 33814 | 6876 |

Table 4 shows the values of makespan, processing cost and energy consumption for 10 different instances. These values represent the minimized values that can be obtained when the inputs were given to each algorithm. For instance, for HMFO, consider mk1 benchmark we got a makespan value of 113, processing cost value 700 and energy consumption value 1093.5. These are the optimum values that can be obtained by the simultaneous optimization of all the three objective functions. To verify the effectiveness of the proposed algorithm, ten different instances are used and their results are shown in Table 4. It can be inferred from Table 4, that makespan, processing cost and energy consumption values are independent of the number of jobs, machines and depends mostly on the type of operations. Taking into consideration not only the number of generation (iteration value) as the deciding factor for the algorithm but also considering the time complexity of order O ($n^2$) makes the algorithm more enhanced. From the results, we have realized that the optimization power of MOO-HMF can project optimal results which help to explore and exploit full capabilities of machines. This also tells us the sequence in which the machines should be supplied with jobs in order to achieve this optimality hence making the production quite profitable.

It is evident from the Table 4 that results of the proposed algorithm is better than NSGA-II. For instance, in almost all the instances the value of makespan, processing cost and energy consumption are lesser for our algorithm when compared to NSGA-II. This makes our algorithm more efficient because of the fact that all three functions of makespan, processing cost and energy consumption are supposed to be minimized and our results are having lesser value than NSGA-II. Thus, we can say that on comparing both algorithms we obtain minimum values in the case of HMFO which proves that it is an improved method to determine the optimal values for all three functions.

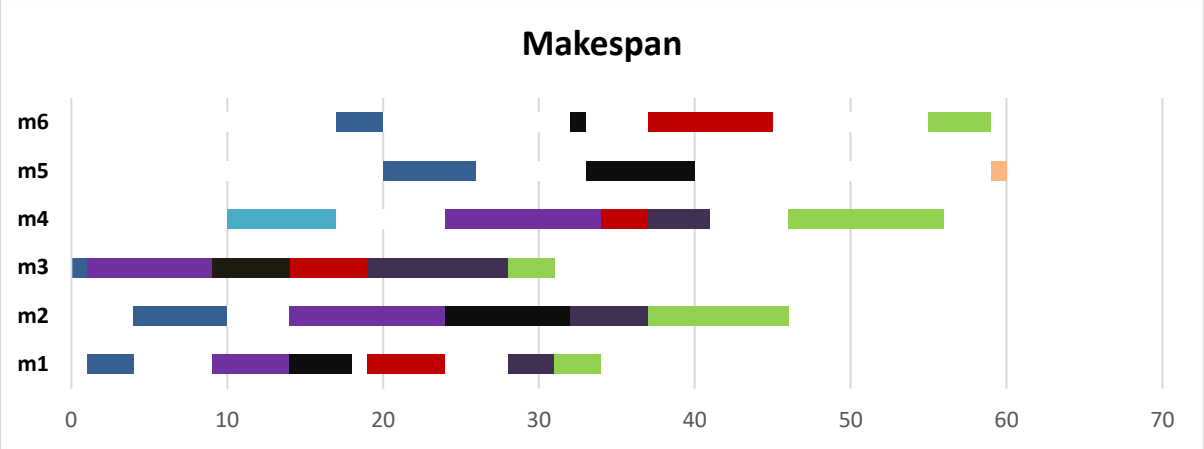



**Figure 6.** Gantt chart for the instance mt06.

A Gantt chart has been popularly used to represent a schedule Fig. 6 illustrates the maximum completion time for the benchmark instance mt06. The x-axis of Gantt chart denotes processing time and the y-axis denotes the machines.

Our program flow model contains complex three-dimensional matrix which covers machines, jobs and operations. The flow of operation starts with an input of time values for respective fields. The time complexity of this operation is O ($n^3$). For the values of processing costs and energy units, the input three-dimensional matrix is multiplied by their respective per unit values for each instance that we have considered. Then, these new-found matrices are mutated to respective normalized values so as to make all of them dimensionally similar to ensure that they are scaled while performing operations on them. Then minimum values are determined row-wise and their sum is computed. This obtained sum is then utilized to form Gantt chart and Hypothesis Test so as to perform a complete analysis of our results.

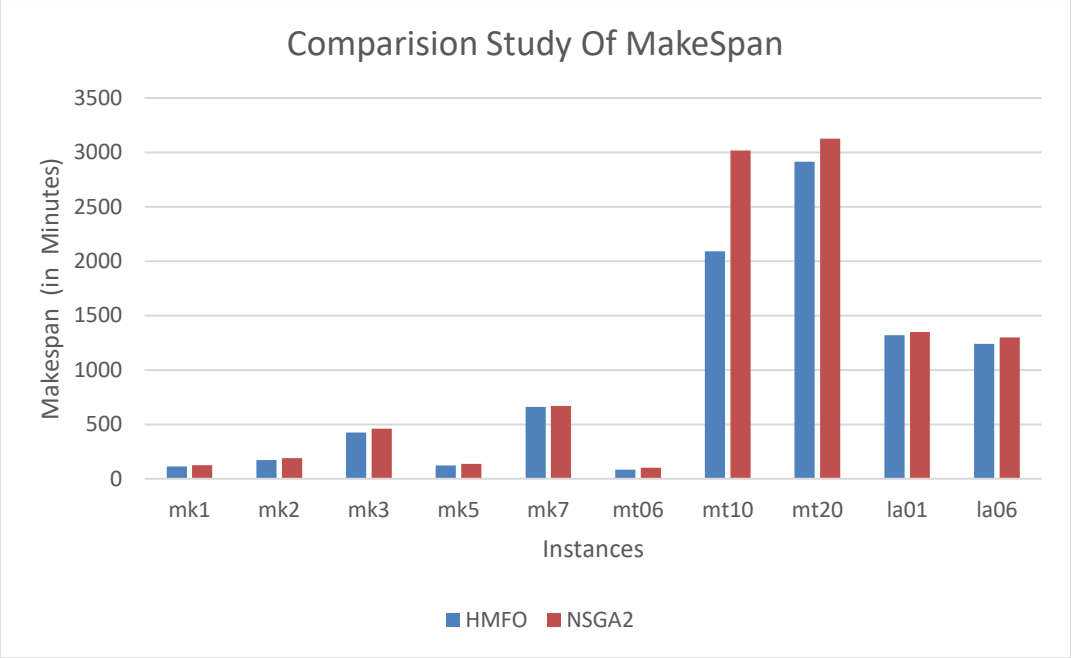



**Figure 7.** Comparison study of makespan for all the instances.

From the graph shown in Fig. 7, we can clearly differentiate the makespan values obtained from the two different algorithms, HMFO and NSGA-II. For the instances with lower makespan, it is observed that the values are approximately same i.e. both the algorithms perform equally well. But as we move towards instances ‘mt10’ and ‘mt20’ a significant difference in makespan values is observed. This difference clearly shows the supremacy of our genetic based algorithm HMFO over a normal algorithm like NSGA-II. So, we can rightfully conclude that our algorithm performs well in both normal as well as stressful environment. So, from the above we can conclude that artificial intelligence can very well find its practical usage in Manufacturing domain for reducing the overall makespan thus reducing processing cost.

## 7. Conclusions and Future Work

In this paper, the authors have developed a mathematical model for the multi-objective problem in the context of dynamic flexible job shop scheduling problem (FJSSP). Due to the problem complexity, it is necessary to develop an efficient multi-objective evolutionary algorithm. Thus, we have used moth flame optimization algorithm and tuned the algorithm operators according to the problem need due to its multi-objective nature by proposing it into a MOO-HMF.

The flexibility in flexible job shop scheduling problem allows the operators to investigate multiple performance measures for the benefit of the manufacturers. Hence, the

authors have investigated the objective functions as makespan, processing cost, and energy consumption. Out of which the performance measure i.e., energy consumption is of utmost important according to the current manufacturing environment. To solve the mentioned multi-objective functions an efficient algorithm needs to be developed to solve to optimality. Henceforth, a hybrid multi-objective moth flame optimization algorithm has been developed. In due course, we have found that some more interdependent objectives like as optimal sequence of jobs, and the number of generations are significant enough for comparing the performance of different algorithms.

While implementing the proposed algorithm an effective process plan has been followed. Thus, we have obtained the optimized solutions. As an extension of this research work, we plan to introduce and evaluate some more sustainable parameters and to develop many objective optimization algorithms.

**Acknowledgement**

This work has been supported by COMPETE: POCI-01-0145-FEDER-007043 and FCT – Fundação para a Ciência e Tecnologia within the Project Scope: UID/CEC/00319/2013, PEst-OE/EEI/UI0760/2014, and PEst2015-2020, and also by Department of Science and Technology, Science & Engineering Research Board (SERB), Statutory Body Established through an Act of Parliament: SERB Act 2008, Government of India with Sanction Order No ECR/2016/001808.

**References**

Brandimarte, P., 1993. Routing and scheduling in a flexible job shop by tabu search. *Annals of Operations research*, *41*(3), pp.157-183.

Bruzzone, A.A.G., Anghinolfi, D., Paolucci, M. and Tonelli, F., 2012. Energy-aware scheduling for improving manufacturing process sustainability: A mathematical model for flexible flow shops. *CIRP Annals-Manufacturing Technology*, *61*(1), pp.459-462.

Deng, Q., Gong, G., Gong, X., Zhang, L., Liu, W. and Ren, Q., 2017. A Bee Evolutionary Guiding Nondominated Sorting Genetic Algorithm II for Multiobjective Flexible Job-Shop Scheduling. *Computational intelligence and neuroscience*, *2017*.

Dugardin, F., Yalaoui, F. and Amodeo, L., 2010. New multi-objective method to solve reentrant hybrid flow shop scheduling problem. *European Journal of Operational Research*, *203*(1), pp.22-31.

Escamilla, J., Salido, M.A., Giret, A. and Barber, F., 2016. A metaheuristic technique for energy-efficiency in job-shop scheduling. *The Knowledge Engineering Review*, *31*(5), pp.475-485.

Fang, H.L., Ross, P. and Corne, D., 1993. *A promising genetic algorithm approach to job-shop scheduling, rescheduling, and open-shop scheduling problems* (pp. 375-382). University of Edinburgh, Department of Artificial Intelligence.

Fang, K., Uhan, N., Zhao, F. and Sutherland, J.W., 2011. A new approach to scheduling in manufacturing for power consumption and carbon footprint reduction. *Journal of Manufacturing Systems*, *30*(4), pp.234-240.

Gao, J., Gen, M., Sun, L. and Zhao, X., 2007. A hybrid of genetic algorithm and bottleneck shifting for multiobjective flexible job shop scheduling problems. *Computers & Industrial Engineering*, *53*(1), pp.149-162.

Gen, M., Gao, J. and Lin, L., 2009. Multistage-based genetic algorithm for flexible job-shop scheduling problem. *Intelligent and evolutionary systems*, pp.183-196.

Gonçalves, J.F., de Magalhães Mendes, J.J. and Resende, M.G., 2005. A hybrid genetic algorithm for the job shop scheduling problem. *European journal of operational research*, *167*(1), pp.77-95.

Jiang, Z., Zuo, L. and Mingcheng, E., 2014. Study on multi-objective flexible job-shop scheduling problem considering energy consumption. *Journal of Industrial Engineering and Management*, *7*(3), p.589.

Karthikeyan, S., Asokan, P., Nickolas, S. and Page, T., 2015. A hybrid discrete firefly algorithm for solving multi-objective flexible job shop scheduling problems. *International Journal of Bio-Inspired Computation*, *7*(6), pp.386-401.

Lawrence, S., 1984. Resouce constrained project scheduling: an experimental investigation of heuristic scheduling techniques (Supplement). *Graduate School of Industrial Administration, Carnegie-Mellon University*.

Li, J.Q., Pan, Q.K. and Gao, K.Z., 2011. Pareto-based discrete artificial bee colony algorithm for multi-objective flexible job shop scheduling problems. *The International Journal of Advanced Manufacturing Technology*, *55*(9), pp.1159-1169.

Liu, Y., Dong, H., Lohse, N., Petrovic, S. and Gindy, N., 2014. An investigation into minimising total energy consumption and total weighted tardiness in job shops. *Journal of Cleaner Production*, *65*, pp.87-96.

Lu, C., Gao, L., Li, X., Pan, Q. and Wang, Q., 2017. Energy-efficient permutation flow shop scheduling problem using a hybrid multi-objective backtracking search algorithm. *Journal of Cleaner Production*, *144*, pp.228-238.

Madureira, A. and Pereira, I., 2010, August. Intelligent bio-inspired system for manufacturing scheduling under uncertainties. In *Hybrid intelligent systems (HIS), 2010 10th international conference on* (pp. 109-112). IEEE.

Madureira, A., Pereira, I., Pereira, P. and Abraham, A., 2014. Negotiation mechanism for self-organized scheduling system with collective intelligence. *Neurocomputing*, *132*, pp.97-110.

Madureira, A., Santos, F., & Pereira, I. 2008, July. Self-managing agents for dynamic scheduling in manufacturing. In *Proceedings of the 10th annual conference companion on genetic and evolutionary computation* (pp. 2187-2192). ACM.

Marichelvam, M.K., Tosun, Ö. and Geetha, M., 2017. Hybrid monkey search algorithm for flow shop scheduling problem under makespan and total flow time. *Applied Soft Computing*, *55*, pp.82-92.

May, G., Stahl, B., Taisch, M. and Prabhu, V., 2015. Multi-objective genetic algorithm for energy-efficient job shop scheduling. *International Journal of Production Research*, *53*(23), pp.7071-7089.

Mirjalili, S., 2015. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-Based Systems*, *89*, pp.228-249.

Muth, J.F. and Thompson, G.L. eds., 1963. *Industrial scheduling*. Prentice-Hall.

Pan, Q.K., Tasgetiren, M.F., Suganthan, P.N. and Chua, T.J., 2011. A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem. *Information sciences*, *181*(12), pp.2455-2468.

Pezzella, F. and Merelli, E., 2000. A tabu search method guided by shifting bottleneck for the job shop scheduling problem. *European Journal of Operational Research*, *120*(2), pp.297-310.

Reddy, M. S., Ratnam, C., Agrawal, R., Varela, M. L. R., Sharma, I., & Manupati, V. K. (2017). Investigation of reconfiguration effect on makespan with social network method for flexible job shop scheduling problem. *Computers & Industrial Engineering*, 110, 231-241. Https://Doi.org/10.1016/j.cie.2017.06.014.

Santos, A. S., Madureira, A. M., Varela, M. L. R., Putnik, G. D., Kays, H. E., & Karim, A. N. M. (2015, June). Scheduling and batching in multi-site flexible flow shop environments. In Information Systems and Technologies (CISTI), 2015 10th Iberian Conference on Information Systems and Technologies (CISTI 2015), pp. 1-6), IEEE/ IEEEXplore. DOI: 10.1109/CISTI.2015.7170525.

Silva, C., Reis, V., Morais, A., Brilenkov, I., Vaza, J., Pinheiro, T., ... & Dias, L. (2017). A comparison of production control systems in a flexible flow shop. Procedia Manufacturing, 13, 1090-1095. https://doi.org/10.1016/j.promfg.2017.09.169.

Sayadi, M., Ramezanian, R. and Ghaffari-Nasab, N., 2010. A discrete firefly meta-heuristic with local search for makespan minimization in permutation flow shop scheduling problems. *International Journal of Industrial Engineering Computations*, *1*(1), pp.1-10.

Sha, D.Y. and Lin, H.H., 2010. A multi-objective PSO for job-shop scheduling problems. *Expert Systems with Applications*, *37*(2), pp.1065-1070.

Tang, D., Dai, M., Salido, M.A. and Giret, A., 2016. Energy-efficient dynamic scheduling for a flexible flow shop using an improved particle swarm optimization. *Computers in Industry*, *81*, pp.82-95.

Tang, L., Liu, W. and Liu, J., 2005. A neural network model and algorithm for the hybrid flow shop scheduling problem in a dynamic environment. *Journal of Intelligent Manufacturing*, *16*(3), pp.361-370.

Varela, M. L., Trojanowska, J., Carmo-Silva, S., Costa, N. M., & Machado, J. (2017). Comparative simulation study of production scheduling in the hybrid and the parallel flow. *Management and Production Engineering Review*, 8(2), 69-80. DOI: 10.1515/mper-2017-0019.

Wolpert, D.H. and Macready, W.G., 1997. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, *1*(1), pp.67-82.

Yagmahan, B. and Yenisey, M.M., 2008. Ant colony optimization for multi-objective flow shop scheduling problem. *Computers & Industrial Engineering*, *54*(3), pp.411-420.

Yang, X., Zeng, Z., Wang, R. and Sun, X., 2016. Bi-Objective Flexible Job-Shop Scheduling Problem Considering Energy Consumption under Stochastic Processing Times. *PloS one*, *11*(12), p.e0167427.

Yin, L., Li, X., Gao, L., Lu, C. and Zhang, Z., 2017. Energy-efficient job shop scheduling problem with variable spindle speed using a novel multi-objective algorithm. *Advances in Mechanical Engineering*, *9*(4), p.1687814017695959.

Zhang, G., Shao, X., Li, P. and Gao, L., 2009. An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem. *Computers & Industrial Engineering*, *56*(4), pp.1309-1318.

Zhang, R. and Chiong, R., 2016. Solving the energy-efficient job shop scheduling problem: a multi-objective genetic algorithm with enhanced local search for minimizing the total weighted tardiness and total energy consumption. *Journal of Cleaner Production*, *112*, pp.3361-3375.