11e CONGRES INTERNATIONAL DE GENIE INDUSTRIEL – CIGI2015
Québec, Canada
26-28 octobre 2015

An effective Genetic Algorithm for the Flexible Job Shop Scheduling Problems

IMEN DRISS[1] , KINZA NADIA MOUSS [2] , ASSIA LAGGOUN[3]
AUTOMATCS &MANUFACTURING ENGINEERING LABORATORY
University of Batna, Algeria.
Idrissamina@hotmail.fr
kinzmouss@yahoo.fr
a.laggoun12@yahoo.fr

Abstract

This present paper will deal with a flexible job shop scheduling problem in order to minimize the makespan. We will propose actually a Genetic algorithm method. In that effective algorithm: a new chromosome representation will be used to conveniently represent a solution for the FJSP, an advanced crossover and mutation operators will be suggested to adapt to chromosome structures. The performance of our proposed approach will be evaluated by a lot of benchmark instances taken from literature. The experimental results will show that the proposed algorithm is a feasible and effective approach for the flexible job shop scheduling problem.

Keywords - FJSP; Scheduling ;Genetic Algorithm ; Chromosome representation.

1. Introduction

Job-shop scheduling problem (JSP) is a branch of production scheduling and combinatorial optimization problems. The classical JSP consists of scheduling a set of jobs on a set of machines under the constraint; each job has a specified processing order. The flexible job-shop problem (FJSP) is anextension of the classical job shop problem in which each operation must be processed on a machine chosen among a set of available ones.

The problem of scheduling job in FJSP can be composed of sub-problems: assigning the operation to machine (the routing problem) and sequencing the operations on the machines (the sequencing problem) in order to minimize the performance indicators. Therefore, the combination of two decisions presents additional complexity.Thus, the FJSP problem is an NP-hardtoo since it is an extension of the job shop scheduling problem (JSP) is proven to be NP-hard[1].

Last years, meta-heuristics were used to solve FJSP namely, tabu search, simulated annealing, genetic algorithm and particle swarm optimization.

In this research, we suggest a new genetic algorithm in order to solve flexible job shop scheduling problems to minimize the Makespan. We create a new chromosome representation called 'permutation job". In fact, this method leads us to find a new coding scheme of the individuals that respects all constraints of the FJSP. At the same time, we employ different strategies for crossover and mutation operators. Computational results show that the proposed algorithm could provide good results.

The paper is organized as follows: the problem definition and formulation are presented in section 2. Literature review is mentioned in section 3. The effective genetic algorithm (eGA)is expanded in section 4. Computational results are presented in section 5. Conclusion and future work are dealt within section 6.

2. Definition and formalization problem

2.1 Problem Description

We focus on flexible job-shop scheduling problems composed of the following elements:

- Jobs. $J = \{J_1 \ldots\ldots J_n\}$ is a set of n jobs to be scheduled. Each job J_i consists of set of predetermined operations. $O_{ij}$is the operationj ofjob J_i. All jobs are released at time 0.
- Machines. $M = \{M_1 \ldots \ldots M_m\}$ is a set of m machines. Each machine can process only one operation at a given time; and each operation can be processed without interruption. All machines are available at time 0.
- Flexibility. The FJSP is classified in two types as follows: [2]
  - Total FJSP (T-FJSP): each operation can be processed on any machine of M existing machines in the shop floor.

- Partial FJSP (P-FJSP): each operation can be processed on one machine of a subset of M existing machines in the shop floor.

➢ Constraints: Are rules that limit the possible assignments of the operations. They can be divided mainly into thefollowing situations:

  - Each operation can be processed by only one machine at a time (disjunctive constraint).
  - Each operation, which has started, runs to completion (non-preemption condition).
  - Each machine performs operations one after another (capacity constraint).
  - Although there are no precedence constraints among operations of different jobs, the predetermined sequence of operations for each job forces each operation to be scheduled after all predecessor operations (precedence/conjunctive constraint)
  - The machine constraints emphasize the operations can be processed only by the machine from the given set (resource constraint).

The hypotheses considered in this paper are summarized as follows:

  - Each operation can be processed without interruption on one of a set of available machines;
  - Jobs are independent and no priorities are assigned to any job type;
  - All jobs are released at time 0, and all machines are available at time 0 too;
  - Breakdowns are not considered;
  - The order of operations for each job is predefined and cannot be modified

➢ Objective: Is to find aschedule that has minimum time required to complete all operations (minimum Makespan).

In order to simplify the presentation of the algorithm; we designed a sample instance of FJSP which will be used a long this paper. Table 1gives the dataset of a P-FJSP including 2 jobs operated on 4 machines where rows correspond to operations and columns correspond to machines. Each cell denotes the processing time of that operation on the corresponding machine. We can represent this system as a graph figure 1

Table1: Processing time table of an instance of P-FJSP

Job	*Operations*	M_1	M_2	M_3	M_4
J_1	O_{11}	5	2	3	1
	O_{12}	8	-	-	8
	O_{13}	-	-	4	3
J_2	O_{21}	4	5	1	-
	O_{22}	3	2	-	1

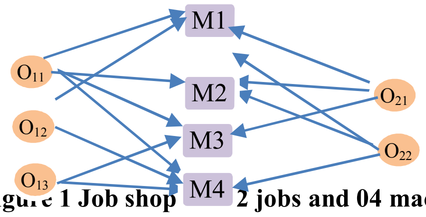


Figure 1 Job shop 2 jobs and 04 machines

2.2 Problem formulation

Some symbols used in our work are listed as follows.

n :Number of the total jobs.

m: Number of the total machines.

i : Index of i^{th} job.

j : Index of the j^{th} operation of job j_i.

J_{io} : Number of the total operation of job J_i.

O_{ij} : j^{th}Operation of job j_i.

p_{ijk} : Processing time of operation O_{ij}on machine k.

S_{ij} :Start time of operation O_{ij}

E_{ijk} : End time of operation O_{ij}on machine k.

$L=\sum_1^n J_{io}$:The sum of all operation of all jobs.

H : Very large positive integer.

Objective function:

$$\text{Minimize}\left[Max\left(C_{1J_1}, C_{2J_2}, \dots C_{nJ_n}\right)\right] \quad (1)$$

Subject to:

$$C_{ij}\text{-}S_{ij}\text{-}\sum_{k:O_{ij\in\Omega_{ij}}}\left(P_{ijk}.X_{ijk}\right) = 0 \quad \forall\, i,j \quad (2)$$

$$C_{\grave{i}\grave{j}}\text{-}C_{ij}\text{+H}\left(1 - Y_{ij\grave{i}\grave{j}k}\right) + H\left(1 - X_{ijk}\right) + \left(1 - X_{\grave{i}\grave{j}k}\right) \geq P_{\grave{i}\grave{j}k}(3)$$

$$C_{ij}\text{-}C_{\grave{i}\grave{j}}\text{+H}\left(Y_{ij\grave{i}\grave{j}k}\right) + H\left(1 - X_{ijk}\right) + \left(1 - X_{\grave{i}\grave{j}k}\right) \geq P_{ijk}(4)$$

$$S_{ij} \geq 0 \;\forall i,j \quad (5)$$

$$S_{ij+1}\text{-}C_{ij} \geq 0 \forall i,j \;\; j=1\ldots J_{i-1}(6)$$

$$\sum_{k:O_{ij\in\Omega_{ij}}}\left(X_{ijk}\right) = 1 \text{ , } \forall i,j \quad (7)$$

$$X_{ijk} = \begin{cases} 1 & \text{if operation } O_{ij} \text{is assigned to machine k} \\ 0 \text{ otherwise} \end{cases} \quad (8)$$

$$Y_{ij\grave{i}\grave{j}k} = \begin{cases} 1\; if operation\; O_{ij} procede\; O_{\grave{i}\grave{j}} on\; machine\; k \\ \quad 0\; otherwise \end{cases} \quad (9)$$

Objective (1) minimizes. The constraint set (2) imposes that the difference between the completion time and the starting time of an operation is equal to its processing time on the machine assigned to. Constraint set (3) and (4) ensure that no two operations can be processed simultaneously on the same machine. This disjunctive constraint (3) becomes inactive when $Y_{ij\grave{i}\grave{j}k}$=0 and the disjunctive constraint (4) becomes inactive when $Y_{ij\grave{i}\grave{j}k}$=1. Constraint set (5) ensures that the start time of an operation is always positive. Constraint set (6) represents the precedence relationship among various operations of a job. Constraint set (7) imposes that operation can only be assigned to one machine.[16]

3. Literature review

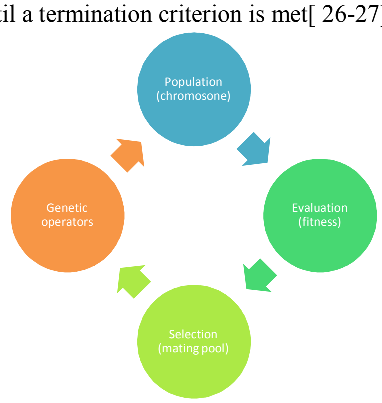
Bruker and Schlie were the first to consider this problem. They developed a polynomial algorithm for solving the flexible job shop scheduling problem with two jobs. However, exact algorithms are not effective for solving FJSP and large instances

[3] Brandimarte was the first to apply the decomposition approach into the FJSP [4] .He solved the routing sub-problem by using some existing dispatching rules and then focused on the scheduling sub-problem, solved by using a tabu search heuristic, Paulli applied hierarchical approach results, but it is rather difficult to be implemented in real operations. [5] and Mesghouni et al were the first to model GA known as parallel job representation for solving FJSP [6]. Chen et al proposed a GA that uses an A-B string representation to solve FJSP for minimum Makespan time criterion [7]. Kacem, Hammadi, and Borne proposed a genetic algorithm controlled by the assignment model generated by the approach of localization (AL) to mono-objective and multi-objective FJSP. [8] Ho and Tay proposed a GA based tool, namely GENACE, for solving the FJSP for minimum Makespan time criterion. The chromosome representation consists of two components, one component for machine selection and the other for operation sequence. Their methodology first generates an initial population using composite dispatching rules. A cultural evolution is then applied to preserve knowledge of schemata and resource allocations learned over each generation. The knowledge or belief spaces in turn influence mutation and selection of individuals. [9]. Zhang and Gen proposed a method called multistage based genetic algorithm to solve FJSP problem [10]. Mehrabad and Fattahi presented a mathematical model and tabu search algorithm to solve the flexible job shop scheduling problem with sequence dependent on setups for minimizing the Makespan[11]. Ho et al proposed an architecture for learning and evolving of flexible job shop schedules for minimum Makespan criterion called learnable genetic architecture (LEGA), a generalization of their previous approach GENACE (Ho and Tay 2004) The population generator module generates a set of feasible schedules equal to the population size using composite dispatching rules and then encodes it into chromosomes of initial population for subsequent evolution in the EA module. During genetic evolution, the SL module modifies the offspring schedules to improve solution quality and to preserve feasibility based on a memory of conserved schemas resolved from sampled schedules sent dynamically from EA module [12]. Gao and Gen developed a hybrid optimization strategy for FJSP multi objective (min Makespan, min maximal machine workload and min total workload) by combining the genetic algorithm and the bottleneck shifting [13]. Tay and Ho proposed a genetic programming (GP) based approach for evolving effective composite dispatching rules for solving the multi-objective FJSP [14]. Girish and Jawahar (2008) proposed a GA for the FJSP for minimum makespan time criterion. The chromosome representation of their proposed GA consists of two strings: one string for machine assignment and the other string for sequencing the operations on the assigned machines using Giffler and Thompson schedule generation procedure (Giffler and Thompson 1960). The chromosomal representation of their proposed GA does not require a repair mechanism and is capable to rummage through the entire search space. [15]. Ponnambalam et al Genetic algorithm (GA) based heuristics that have adopted Giffler and Thompson (GT) procedure, an efficient active feasible Schedule for Makespan time criterion [16] . Giovanni and Pezzella proposed an Improved Genetic Algorithm to solve the Distributed and Flexible Job-shop Scheduling problem to minimize the Makespan [17]. Sun et al presented a research on flexible job shop scheduling problem based on a modified GA [18]. Motaghedi et al presented an effective hybrid genetic algorithm to solve the multi-objective flexible job shop scheduling problems [19]. Guohui et al proposed an effective genetic algorithm for solving the flexible job-shop scheduling problem (FJSP) to minimize Makespan criteria[20] Zhang et al proposed a genetic algorithm with tabu search procedure for Flexible Job Shop Scheduling Problem (FJSP) with transportation constraints and bounded processing times to minimize the Makespan and the storage of solutions [21]. Chen et al proposed Genetic Algorithm (GA) and Grouping Genetic Algorithm (GGA) for job shop scheduling problem with parallel machines and reentrant process [22].

4. An effective Genetic Algorithm for flexible job shop scheduling problem

4.1 Basis of genetic Algorithm

Genetic algorithms (GAs) are search methods based on principles of natural selection and genetics[23-24].The interest in heuristic search algorithms with underpinnings in natural and physical processes began as early as the 1970s,when Holland' first proposed genetic algorithms[25]. The fundamental underlying mechanism to start the search GAs is initialized with a population of individuals. The individuals are encoded as chromosomes in the search space. GAs use mainly two operators namely, crossover and mutation to direct the population to the global optimum.Crossover allows exchanging information between different solutions (chromosomes) and mutation increases the variety in the population. After the selection and evaluation of the initial population, chromosomes are selected the crossover and mutation operators are applied. Then the new population is formed. This process is continued until a termination criterion is met.This process is continued until a termination criterion is met[26-27].



-

Figure.2 .The cycle GAs

4.2 Chromosome representation

Our chromosome representation has one component which is Job Permutation (JP) We use array of integer, the length equals to L,

and each integer value equals the index of array of job correspondent [30] .
From chromosome values we deduct three components the first is operations for each job in order to verify operation sequence constraint, the second is machine assignment and processing time for each operation as third. we start with first one, in fig.3 we can see operation[1] =1 because chromosome[1]=1 is the first instance of job 1 and operation[2]=1 because chromosome[2]=2 is the first instance of job 2 then operation[3] =1 because chromosome[3]=2 is the second instance of job 1 and so on. Now we have job and operation , we can choose randomly from table 1 one machine to complete the machine vector and time processing.

According to table 1 we can take these two examples:

Chromosome 1=	1	2	1	2	1

Input data

Operation sequence	1	1	2	2	3
Machine assignment randomly	M1	M3	M4	M2	M4
Processing time	5	5	1	2	3

Schedule 1 = O_{11},O_{21},O_{12},O_{22},O_{13}

Chromosome 2=	1	2	1	1	2

Input data

Operation sequence	1	1	2	3	2
Machine assignment randomly	M2	M3	M1	M3	M4
PTime= Processing time	2	5	8	4	1

Schedule 2 = O_{11},O_{21},O_{12},O_{13},O_{23}

Figure 3. Two different permutations for chromosome with input data

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>M4</td><td></td><td></td><td></td><td></td><td></td><td>O12</td><td colspan="3">O13</td></tr>
<tr><td>M3</td><td colspan="5">O21</td><td></td><td></td><td></td><td></td></tr>
<tr><td>M2</td><td></td><td></td><td></td><td></td><td></td><td colspan="2">O22</td><td></td><td></td></tr>
<tr><td>M1</td><td colspan="5">O11</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table>

Figure 4 .Gantt char for schedule 1

These components must have the same length. Now we have all we need to calculate makespan.
In this case we use some intermediate data structures such as processing which is matrix of n_j rows and no columns, and STime ETime presenting start and end time for each operation and finally EMachine, end time for machines to calculate makespan for Schedule 1(O_{21},O_{22},O_{11},O_{23},O_{12}) we need the following matrix

$$Processing = \begin{array}{c} \\ J1 \\ J2 \end{array} \left[\begin{array}{ccc} O1 & O2 & O3 \\ 5 & 1 & 3 \\ 5 & 2 & \end{array}\right] \text{ matrix of processing time}$$

$$STime = \begin{array}{c} \\ J1 \\ J2 \end{array} \left[\begin{array}{ccc} O1 & O2 & O3 \\ 0 & 5 & 9 \\ 0 & 5 & \end{array}\right] \text{ matrix of operation start time}$$

$$ETime = \begin{array}{c} \\ J1 \\ J2 \end{array} \left[\begin{array}{ccc} O1 & O2 & O\,3 \\ 5 & 6 & 9 \\ 5 & 7 & \end{array}\right] \text{ matrix of operation end time}$$

$$EMachine = \left[\begin{array}{cccc} M1 & M2 & M3 & M4 \\ 5 & 7 & 5 & 9 \end{array}\right] \text{ matrix of end time}$$

machine
Makespan = max(ETime) = max (EMachine)= 9

It is time to apply the genetic algorithm with selection, mutation and crossover operators on chromosome in our case is permutation of jobs.

We can not apply genetic operators on operations sequences and machine because we lose constraints and we will be obliged to add another algorithm to verify them.

The procedures of job permutation encoding chromosome and decoding used in this paper are described in Figure 5 and Figure 6

Procedure: Job Permutation Encoding

Input: Total number of state (machine) **m**, total number of jobs **nj**, total number of operation for each job **no**$_j$ processing time matrix **Time**.
Output: Digit permutation chromosome CHROM; assign machine, processing time
Step 1: Generate randomly a vector CHROM of sequencing operations**.**
Step 2: Set indexl=1.
Step 3: Choose randomly fromTime matrix a machine such two operations cannot be processed simultaneously on the same machine.
Set machine(l)
Set processing(l)
Permutation; assignment machine and processing time for each operation is built and read from left to right.
Setl = l + 1.
step 4: repeat step 3 until l = K
Output CHROM, machine, processing.
End

Figure 5. Procedure of Job Permutation Encoding

Procedure: Job Permutation Decoding

Input: Total number of state (machine) **m**, total number of jobs **nj**, total number of operation for each job **no**$_j$, digit permutation **CHROM**; assign machine**Machine**; processing time matrix **Time**.
Output: Best schedule, Makespan
Step 1: Deduct from CHROM operation's order
Step 2: Choose machine for each operation
Step 3: Calculate Makespan
Best schedule, Makespan
End

Figure 6. Procedure of Job Permutation Decoding

4.3 Genetic operators

Selection operator

Choosing individuals for reproduction is the task of selection [28] [30].The chosen selection approach is adopted. Detailed steps are in figure.7.

Procedure: Wheel Selection Job

Input: Total number of state (machine) **m**, total number of jobs **nj**, total number of operation for each job **noj**, total-number of individuals in one population **np**, Makespan for each individual **Mspan(np)**, Population
Output:SelectedpopulationNewpopulation
Step 1:SortMspan; set indexl=1.
Step 2:Prob(l)=Mspan(l)/(sum(Mspan))
l = l + 1.
Step 3: Repeat step 2 until l = np
Step 4:Probp=cumsun(Prob)
Step 5:Select best individuals
Perc= np*30%
Keep 50% of best individuals
rdn random probability
Set index l=1, repeat
Set index k=1, repeat
If rdn<Probp(k)
Newpopulation(l)=Population(k)
Break
Endif
k=k+1
Until k=np
l=l+1
Until l=Perc
OutputNewpopulationWith 30% of best individuals.
End

Figure.7. Procedure of Wheel Selection Job

Crossover operator

The goal of the crossover is to obtain better chromosomes to improve the result by exchanging information contained in thecurrent good ones. In our work we carried out two kinds of crossoveroperator for the chromosomes.

In accordance with the adopted representation two crossover operators are used in this work[30].. Uniform crossover and a Precedence preserving order-based crossover (POX). Uniform Crossover Operation is described in Figure.8

Procedure:Uniform Crossover Job

Input:Total number of indivuduals in one population np, Makespan for each individual Mspan(np), Newpopulation
Output:Newpopulation
Step 1:Choose randomly 2 individuals parent1;parent2
Step 2: Generate randomly a binary vector unif(np)
Step 3:Set index l=1.
Step 4:If unif(l)=1
Offset1(l)=parent1(l)
Offset2(l)=parent1(1)
With Precedence preserving order
Endif
l = l + 1.
Step 5: Repeat step 3 until l = 50% of population
Replace parent1,2 with offset1,2
Newpopulation
End

Figure 8. Procedure of Uniform Crossover Job

Mutation operator

Mutation introduces some extra variability into the population to enhance the diversity of population. Usually, mutation is applied with small probability. Large probability may destroy the good chromosome. In our research work we carried out one kind of mutation operator which is the mutation by values for the chromosome PJ (values mutation Job is described in Figure 9).

Procedure:Values Mutation Job

Input:Total number of indivuduals in one population np, Makespan for each individual Mspan(np), Newpopulation
Output: Newpopulation
Step 1:Set index l=1
Step 2:Choose randomly an individual from Newpopulation
Step 3:Generate randomly 2 values in an individual
Permute these values if precedence order
Step 4: Repeat step 1 until l 50% of population
Output Newpopulation
End

Figure 9. procedure of values mutation job

5. Experimental results

To obtain meaningful results, we ran our algorithm five times on the same instance. The parameters used in the eGA were chosen experimentally to obtain a satisfactory solution within an acceptable time span.
The effective GA is tested on Brandimarte's data set (BR data). The data set consists of ten problems with number of jobs ranging from 10 to 20, number of machines ranging from 4 to 15 and number of operations for each job ranging from 5 to 15.

The proposed effective algorithm is compared to the following algorithms[17-20-29] M&G (2000) ,GENACE (2004), Zhang etal (2011),Chen et al (1999), Pessella et al (2008), HGTS (2014).

The proposed eGA algorithm for FSJ problem was coded in Matlab and run on 2.3 GHz PC with 4GO memory with the following parameters: popsize =100,P_c= 0.8,P_m= 0.05, selection perc =30%.
Table 2 summarizes the experimental results. It lists problem names, problem dimension (number of jobs × number of machines), the best known solution (C_m), the solution obtained by our algorithm (eGA) and the solution obtained by each of the other algorithms. C_m denotes makespan time. *indicates the best known solution to date and denotes makespan time .** indicate the best known solution best to date The computational results show that

the effective genetic algorithm is speed so far of searching optimal solution. Among the ten test problems, Mk01could gain better solution in all the approachs. Mk03 and Mk08 could get the optimal solution in the first generation by using NGA. Mk02,Mk04,Mk05,Mk09 (Four problems) could gain the same good results as in M&G's approach. One problem Mk06 could gain the same results as GENACE and two problems (Mk07,Mk10 problems) could gain the same results as Chen et al In figure 10-11 we draw the decrease of Makespan and Gantt chart for the Mk01 test problem with 10 jobs and 6 machines from BR'sdata.

Table 2: Results of Brandimart's data.

Problem	n*m	M&G	GENACE	Zhang et al	Chen et al	Pezzella et al	HGTS	NGA
		C_m	C_m	C_m	C_m	C_m	C_m	C_m
Mk01	10*6	40	40	40	40	40	40	37**
Mk02	10*6	26	32	26	29	26	26	26*
Mk03	15*8	204	N/A	204	204	204	204	204*
Mk04	15*8	60	67	60	63	60	60	60*
Mk05	15*4	173	176	173	181	173	172	173
Mk06	10*15	58	67	58	60	63	57	67
Mk07	20*5	144	147	144	148	139	139	148
Mk08	20*10	523	523	523	523	523	523	523*
Mk09	20*10	307	320	307	308	311	307	307*
Mk10	20*15	198	229	198	212	212	198	212

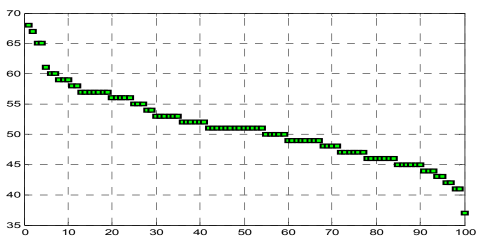


Figure 10.Decreasing of the Makespan (MK01)

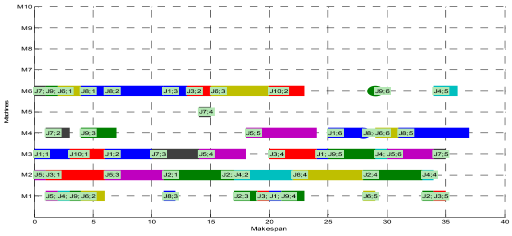


Figure 11.The Gantt chart of MK01

5. Conclusion and future work

Recently, mono-objective flexible job-shop scheduling problem has attracted many researchers' attention. Whereas this problem is well known as NP-hard. The complexity of this problem leads to the appearance of many heuristic approaches, and the research is mainly concentrated on evolutionary algorithms.

In this research, a meta-heuristic algorithm based on GA is developed for solving flexible job shop scheduling problems to minimize makespan . In our algorithm, we proposed a new chromosome representation scheme is applied, and efficient crossover and mutation operators are proposed to adapt. The numerical experiments indicate the effectiveness of the proposed approach. However, there are still a number of further works that need to be considered in the future. can be considered for FJSP but not limited with this objectives There may be some other side constraints and It can also be multi-objective optimization problem : Objective function can be to minimize the makespan, the, tardiness, maximum lateness

Acknowledgment

We would like to thank Pr. A. Driss for checking our English phrasing, Pr. Samai Aitouch for helping us with the organization and the methodology of this paper and finally my dear husband for his great support.

References

[1] M.R.Garey,D.S.Johmson ,andR.Sethi ,The complexity of flow shop and job shop scheduling,*Mathematics of Operational Research* ,1 (1976) 117-129 .

[2] I.Kacem,S. Hammadi,and P.Borne,Pareto-optimality approach for flexible job shop scheduling problems .Hybridization of evolutionary algorithms and fuzzy logic , *Matimatic and Computers in Simulation* ,60 (2002) 245-276.

[3] P.Brucker,R.Schile,Job-shop scheduling with multi-purpose machines, 45(4) (1990) 369–375.

[4] P. Brandimarte, Routing and scheduling in a flexible job shop bytabu search,Annals of Operations Research, 41(1993) 157–183.

[5] J.A.Paulli,Hierarchical approach for the FMS scheduling problem, *European Journal of Operational Research*, 86 (1)(1995) 32- 42.

[6] K.Mesghani , S.Hammadi, and P. Borne, On modeling genetic algorithms for flexible job shop scheduling problems , (1998).

[7] H. Chen, J. Ihlow,and C.A. Lehmann, Genetic algorithm for flexible Job shop scheduling*, IEEE International Conferenceon Robotics and Automation*, Detroit vol 2 (1999) 1120- 1128.

[8] I.Kacem, S.Hammadi,andP. Borne,Approche By Localization And Multiobjective Evolutionary And Optimization For Flexible Job Shop Scheduling Problems , *IEEE Transations Man, and Cybernetrics* , 32 (1) (2002) 1-13.

[9] N.B .Ho, J.C .Tay,GENACE : An Efficient Cultural Algorithm for solving the flexible job shop problem, In: *Proceeding of IEEE Congress on Evolutionary Computation*,Vol .1, (2004) 1759-1766.

[10] H .P Zhang, and M .Gen, Multistage-based genetic algorithm for flexible job shop scheduling problem, *Journal of Complexity International* ,48 (2005)409-425.

[11] P.Fattahi ,M.Saidi Mehrabad,F.Joli , Mathematical Modeling and Heuristic Approaches to Flexible Job Shop Scheduling Problems , *Journal of Inteligent Manufacturing,*vol 18 (2007) 331-342.

[12] N.B Ho,J.C Tay, M.Edmund,andK.Lai,An effective architecture for learning and evolving flexible job shop schedules ,*European journal of Operational Research*,179 (2007) 316-333.

[13] J.Gao,M Gen,L.Sun,Ahybrid of genetic algorithm and bottleneck shifting for multi objective flexible job shop scheduling problems,*Computers and Industrial Engineering* vol 53 (2007) 149-162.

[14] J.C. Tay , N.B. Ho, Evolving dispatching rules using genetic programming for solving multi- objective flexible jobshop problems , *Computers and Industrial Engineering* vol 54 .2008).453-473.

[15] B.S. Girish , N.Jawahar ,Scheduling job shops associated with multiple routings with genetic and ant colony heuristics(2008).

[16] S.G Pannanbalam ,N. Jawahar, and B.S.Girish, Giffler and Thampson Procedure based genetic algorithms for scheduling job shops ,*Springer-Verbag Berlin Heidelberg*(2009)229-259.

[17] F.Pezzela ,G.Margenti ,and G.Ciaschetti,A genetic algorithm for flexible job shop scheduling problem,*Computers and Operations Research,* 35(10)(2007).3202-3212.

[18] W.Sun,Y.Pan ,X.Lu, and Q.Ma , Research on flexible job shop scheduling problem based on a modified genetic algorithm., *Journal of Mechanical Science And Technology* 24(10) (2010) 2115-2119.

[19] A.Motaghedi,K.Sabri-Laghare, and M.Heydari, solving flexible job shop scheduling problem with multi objectives, *International Journal of Industrial Engineering and Production Research* ,vol 21(2010).197-209.

[20] G.Zhang ,L.Gao, and Y.Shi,An effective genetic algorithm for the flexible job shop scheduling problem, *Expert System with Application*,vol 38 (2011). 3563-3573

[21] Q.Zhang ,H Manier,and A.Manier , A genetic algorithm with tabu search procedure for flexible job shop scheduling with transportation constants and bounded proceding times,*Computers and operations Research*,vol 39 (2012) 1713-1723.

[22] J.C.Chen,C.C.Wu, and C.W.Chen, Flexible job shop with parallel machines using Genetic Algorithm and Grouping Genetic Algorithm , *Expert Systems with Application* ,vol 39 (2012)10016-10021.

[23] N.Kim ,H.Kim ,and J.Lee, Damage detection of truss structures using two stage optimization based on micro genetic algorithm,*Journal Of Mechanical Science And Technology*28(9) (2014) 3687-3695 .

[24] R.N.Yadar ,V.Yadar,and G.H.Singh ,Application of non dominated sorting genetic algorithm for multi objective optimization of electrical discharge diamond face grinding process , *Journal of Mechanical Science And Technology*,28(6) (2014) 2299-2306.

[25] L.N.Xing, Y.U.Chen,.and K.W.Yang , Multi. Population interactive coevolutionnary algorithm for flexible job shop scheduling problems ,*Comput Optim Appl*(2009)DOI 10.1007//S 10589-009-9244-7 .

[26] F.N.Defersha,M.Chen , A Coase-Grain Parallel Genetic Algorithm for flexible job shop scheduling with lot streaming ,*In IEEE International Conference on Computational Science and Engineering,* (2009).

[27] Y.K.Park ,J.M.Yang ,Optimization of mixed casting processes considering discrete ingot sizes , Journal Of Mechanical Science and Technology 23 (2009) 1899-1910.

[28] S.F.Hwang,Y.Hsu,and Y.Chen,Agenetic algorithm for the optimization of fiber angles in composite laminates , *Journal of Mechanical Science and Technology*,28(8) (2014) 3163-3169.

[29] J.J.Palacios. ,A. González ,and C.R.González , Genetic tabu search for the fuzzy flexible job shop problem , *Computers & Operations Research*,(2014) 5474–89.

[30] I.Driss, K.N.Mouss,A.Laggoun , A New genetic algorithm for flexible job shop scheduling problems, *Journal of Mechanical Science and Technology 29 (3) (2015) 1273~1281*