A New Heuristic Method for Solving Joint Job Shop Scheduling of Production and Maintenance

N. Fnaiech [*] **C. Fitouri** [***] **C. Varnier** [**] **F. Fnaiech** [*] **N. Zerhouni** [**]

[*] *Research team in the Signal, Image and Energy Management (SIME), (ENSIT), University of Tunis, 1008, Tunis, Tunisia (e-mails: fnaiechnader@yahoo.fr, fitouricyrine@yahoo.fr, fnaiech@ieee.org)*
[**] *Femto-st institute, Dept AS2M, Franche-Comté, University/ENSMM/CNRS, 25000, Besançon, France (e-mails: christophe.varnier@ens2m.fr, Noureddine.Zerhouni@ens2m.fr)*
[***] *University of Tunis El Manar, National engineering school of Tunis (ENIT), 1002, Tunis, Tunisia*

Abstract: Many heuristics and intelligent methods have been proposed and applied in order to solve the Job Shop Scheduling Problems (JSSP). Several researches have so far been interested in solving the production planning in JSSP and few of them have focused on solving production scheduling with the presence of maintenance tasks. This paper presents a new heuristic method (NHGA) that includes two new techniques. The first, is a Modified Genetic Algorithm (MGA) which is inspired from the different steps of standard Genetic Algorithms (GA). Practically, when the GA is used, usually many steps, such as crossover and mutation, are based on random choices. The idea of MGA technique is to enhance the random character of such choices through guiding the steps of GA in a logical procedure, while following at each generation and each step the most plausible solutions to solve the JSS problem with maintenance periods. Henceforth, the new modifications reported in the MGA take into consideration the initial population, selection, crossover, mutation and the running mechanism of the algorithm. This has been sustained by a second technique called Heuristic Displacement of Genes (HDG) such a technique would take as an objective improving the obtained solutions of JSSP. The technique NHGA has been tested on many benchmarks, and compared with standard GA and other recent methods. The obtained results actually shed light on the efficiency of our new heuristic method.

Keywords: Scheduling algorithms, Production, Management, Maintenance, Resource allocation, Availability, Optimization, Genetic algorithms.

1. INTRODUCTION

Scheduling is the process of deciding how to commit resources to each service in an effective way while respecting certain conflicts and aiming to optimize certain objectives. In this paper, the job shop production scheduling problem under machine unavailability is considered.

The job shop scheduling problems (JSSP) are considered among the most complicated problems in industry (NP-hard). It is a combinatorial optimization problem like the problem proposed by Brucker et al. (2007). It has been studied by several researchers using different algorithms and optimization methods such as branch and bound algorithms for Jones et al. (2001); the shifting bottleneck heuristic for Adams et al. (1988); dispatching rules for Chiang and Fu (2007), and other techniques based on tabu search for Glover (1990); simulated annealing for Van L. et al. (1992); neural networks for Yahyaoui et al. (2011) and genetic algorithms for Omar et al. (2006).

[1] This work has been supported by the Labex ACTION project (contract ANR-11-LABX-01-01).

Genetic algorithms (GA) are the most used in solving this kind of problem. Usually, the GA is a search heuristic that mimics the process of natural selection. It starts with a population of individuals, which represents solutions properly marked with a code that identifies them. Evaluation procedure is then necessary to determine the strength of each individual in a population. Then, in selection phase, individuals are selected from which the following population will be created. Indeed, it is a process that allows choosing among the current population of individuals, those are most adapted to present at the crossover and mutation steps. Individuals are randomly assigned in pairs. The parent chromosomes are copied and recombined to produce each one, two sons with characteristics from both parents. Mutation plays a secondary role in relation to crossover. It is used to introduce minor changes to some individuals in the population. So it ensures both local and global search. This helps to generate a new population of individuals who may probably be better than those of the previous generations.

Note that, in the literature, some researchers worked with modified genetic algorithms where in the sequel, some of their works are listed below.

Wang and Zheng (2002) presented a modified GA for JSS whose changes concern, the use of an effective crossover operation for operation-based representation with the purpose to guarantee the feasibility of solutions and, the substitution of the mutation process by the simulated annealing in order to reinforce the neighborhood search and to avoid premature convergence of the algorithm. Omar et al. (2006) used genetic algorithm (GA) with some modifications in the initial population, crossover and mutation to deal with problem of job shop scheduling.

All the works described above have dealt with classical scheduling problems. The authors have used GA as optimization method. Nevertheless, they all consider problems where machines are always available. In real application case it is often necessary to maintain machines in order to ensure a nominal performance of the production system. Then, maintenance periods for which machines are not available should be taken into account in the scheduling phase. Several maintenance policies can be considered. The simplest case is fixed period of maintenance.

In this framework, the problem is known as scheduling problem with availability constraints. Several works were carried out in this field. A comprehensive review paper was published by Schmidt (2000). He listed for each typology of scheduling problem (single machine, flow shop, job shop, parallel machines, etc.) the results already obtained. Few works were done on the job shop case. We can cite Gao et al. (2006). They proposed a hybrid genetic algorithm for the flexible job shop. One of the most recent works is proposed by Lei (2011) who presented a swarm based search for solving job shop problem with preventive maintenance, but with presumable jobs and fuzzy objective.

In this paper, the scheduling problem of production and maintenance tasks is addressed. The main drawback of proposed approaches for solving this problem with standard GA is the computation time needed to converge towards the best solution. The objective of the proposed method in this paper, is to reduce the number of generations to reach the solution and consequently the convergence time. We present here a new heuristic technique based on the steps of the standard GA. At each step of the approach we propose to guide the method toward the optimal solution.

The remaining sections of this paper are organized as follows: section 2 presents the considered problem. A first modification of GA is described in section 3. Another complementary technique (Heuristic Displacement Genes) is presented in section 4. Section 5 summarizes the running mechanism of the global approach (called NHGA). The results are discussed in section 6. Finally, section 7 gives some concluding remarks.

2. NOTATIONS AND PROBLEM STATEMENT

The Job shop is a workshop production where tasks of jobs have to be scheduled. This problem is stated as follows: consider n jobs. Each job has to be processed on m machines following different sequence. Each machine can perform only one job at a time. The problem consists in defining the execution order of tasks on each machine, while respecting the sequence constraints. The different tasks of each job are denoted by O_{ijk}, which is the j^{th} operation of job J_i that is processed on the machine M_k, and the processing time of O_{ijk} is noted P_{ijk}.

Machine are subjected to preventive maintenance periods, which are fixed according to the maintenance planning. During a maintenance period the machine is not available for a job task. Tasks are not able to be pre-empted neither by another task nor by maintenance operation. Consequently, the problem addressed in this paper is the JSSP with fixed unavailability constraints.

2.1 Notations

The different common notations and abbreviations used along the paper for mathematical modelling and for the proposed NHGA method for the scheduling problem with availability constraints are given as follows:

Job-shop problem notations:

- $J = \{J_1, J_2, \ldots, J_n\}$: Set of n jobs;
- $M = \{M_1, M_2, \ldots, M_m\}$: Set of m machines (resources);
- i, y: Indexes of a job $(i, y = 1, \ldots, n)$;
- j, l: Indexes of an operation $(j, l = 1, \ldots, m)$;
- k, q: Indexes of a machine $(k, q = 1, \ldots, m)$;
- O_{ijk}: The j^{th} operation of job J_i to be processed on machine M_k;
- P_{ijk}: Processing time of the operation O_{ijk};
- ST_{ijk}: Starting time of the operation O_{ijk}.

Fixed maintenance period notations:

- I_k^x: Starting date of the x^{th} unavailability on machine M_k;
- F_k^x: Ending date of the x^{th} unavailability on machine M_k;
- $[I_k{}^x, F_k{}^x]$: Interval of the x^{th} unavailability on machine M_k.

Modified genetic algorithm notations:

- u, v: Indexes of a gene.

2.2 Constraints

Sequence Constraint :

It is necessary in a feasible schedule that:

$$ST_{ijk} \geq ST_{i(j-1)q} + P_{i(j-1)q} \quad (1)$$

Resource Constraint :

In order to avoid the resource conflict, for all couple of operations O_{ijk} and O_{ylk} processed by the same machine M_k, it is required that:

$$[ST_{ijk}, ST_{ijk} + P_{ijk}] \cap [ST_{ylk}, ST_{ylk} + P_{ylk}] = \emptyset \quad (2)$$

Maintenance Constraint :

The machines are not always available; they are subject to maintenance operation during fixed periods (see figure 1). During these periods, they are not able to process any job.

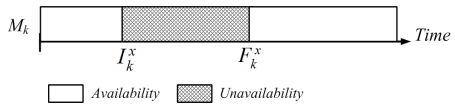

Fig. 1. Maintenance periods representation

The deal is to execute operations on these machines in an optimal way. This said that there wont be any loss of time.

2.3 Objective

The main goal is to minimize the makespan (C_{max}), which is the cumulative time to complete all operations on all machines with respecting sequence, resource and unavailability constraints. Any JSSP of n jobs and m machines aiming to minimize the makespan C_{max} can be written as $n/m/J/C_{max}$. To solve this problem, we would propose a new heuristic method called NHGA, which includes two new complementary techniques. These techniques have as an aim ensuring the convergence towards the best global optimal types of solution of joint JSSP:

- The first technique is a Modified Genetic Algorithm (MGA)
- The second technique is Heuristic Displacements of Genes (HDG)

3. THE FIRST PROPOSED TECHNIQUE: MODIFIED GENETIC ALGORITHM (MGA)

The standard genetic algorithm depends on arbitrary choices including the creation of the initial population and selection of chromosomes for the crossover and mutation. Additionally, standard GA hide many drawbacks, it is based on arbitrary choice of a given population, which asks for many generations so that it converges towards the desired solutions, even if such solutions do not display any certainty that they are the best. For this reason, the same process is applied on another population, in order to get sure that best solution is already achieved.

This section presents the steps of the proposed MGA applied to the job shop production scheduling problem in juxtaposition with fixed periods of maintenance. The MGA is inspired from different steps of the standard genetic algorithm while adding some new modifications.

So, the problem here is to find the best solution among several other solutions obtained through the different steps. The most organized chromosome, leading to a minimum C_{max} is considered as the solution of the problem.

3.1 Organized Chromosomes

Job-based Representation :

The machine sequences and job sequences are both necessary to form this representation. Here, the first job is scheduled first. Moreover, the sequence of operations of each job is determined from the machine sequence. For example, if the number 1 occurs three times. And the corresponding values in the machine field are 3, 2 and 5. It means that the sequence of the job J_1 is 3 2 5 (Ponnambalam et al. (2001)). So, the classification of operations into the chromosome is performed while respecting the

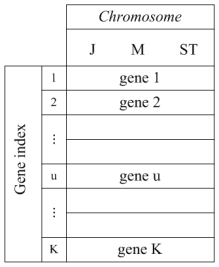

Fig. 2. Representation of a chromosome

sequence constraint. This representation shown in figure 2 is customized. In fact, J represents the job numbers, M represents the required machines for that job, and ST the starting time is represented here in order to explain the run mechanism of the proposed NHGA in the sequel of the paper. Note that K is the number of genes (operations).

Computation of Starting Times ST :

In order to obtain only feasible solutions, the ST of different operations must be also computed with respecting the resource and unavailability constraints. The proposed algorithm considers only semi-active schedule, it leads for each chromosome to a unique solution. Six rules are then used. Notably, the chromosome shall be built from top to down.

For each gene, the following rules should be hold :

- Rule 1: The determination of the ST is performed starting by evaluating the first gene then the second gene and so on up to the last gene. The starting time of the first gene is automatically initialized to 0.
- Rule 2: If a gene has J_i and M_k that appear for the first time then, the ST will be set to 0.
- Rule 3: For two consecutive genes u (corresponding to the task O_{ijk}) and $(u+1)$ (for the task O_{ylq}),
$$ST_{ylq} \geq ST_{ijk} \tag{3}$$
- Rule 4: Two consecutive genes u (O_{ijk}) and v ($O_{i(j+1)q}$) with ($u < v$) including the same job J_i must respect the sequence constraint:
$$ST_{i(j+1)q} \geq ST_{ijk} + P_{ijk} \tag{4}$$
- Rule 5: Two consecutive genes u (O_{ijk}) and v (O_{ylk}) with ($u < v$) including the same machine M_k must respect the resource constraint:
$$ST_{ylk} \geq ST_{ijk} + P_{ijk} \tag{5}$$
- Rule 6: Maintenance constraint (See figure 3 and figure 4:
$$If\ ST_{ijk} \leq {F_k}^x\ and\ ST_{ijk} + P_{ijk} \geq {I_k}^x \quad then\ ST_{ijk} = {F_k}^x \tag{6}$$

In conclusion, the lowest value of ST which satisfies equations 3, 4, 5 and 6 will be the ST considered.

Initial Population :

The initial population includes a number of chromosomes with different classifications of genes. At each chromosome, this classification of genes (operations) must respect the sequence, resource and unavailability constraints in order to ensure a feasible solution. Here, the initial population is composed of two sub-populations; sub-population 1 of

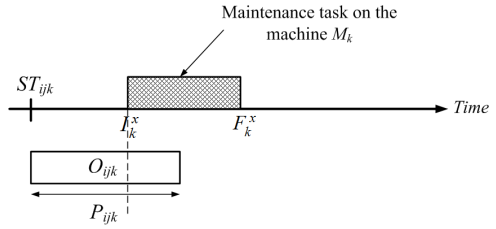

Fig. 3. Situation of conflict

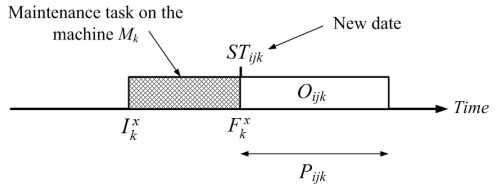

Fig. 4. Offset of the starting time of the operation towards the nal date of the maintenance task

random chromosomes and sub-population 2 of specific chromosomes whose at least one operation is classified in its best ST.

3.2 Fitness Value

The step of evaluation of the genetic algorithm consists in calculating a fitness function for every chromosome and assigning it a weight in the population. Our method of assessment consists in evaluating each chromosome by two significant parameters with which we can decide whether or not this chromosome can lead to an optimal solution. The first parameter is known as makespan C_{max} which indicates the time required for a production cycle to end. It corresponds to our main objective. The second parameter is defined as the time lost between operations on each machine rated T_k where k is the index of the machine and $k = 1, \ldots, m$. This second characteristic is only used to guide the next phase of our heuristic algorithm. So each chromosome is evaluated by:

- Makespan: C_{max} ;
- Time lost on each machine k: T_k

The later parameter is given by the following formula:

$$T_k = \max_{i,j} (ST_{ijk} + P_{ijk}) - \min_{i,j} (ST_{ijk}) - \sum_{i,j} P_{ijk} \quad (7)$$

We seek the optimal solution, which has a minimum C_{max} and the best combination of T_k for all machines.

3.3 Smart Selection

In the literature, there are several methods of chromosome selection such as the selection by Roulette Wheel, the selection by rank, the selection by elitism, etc. In order to enhance the randomness of the selection usually used in standard genetic algorithms, we propose a complementary new method of selection, which is defined as follows:

- Firstly, the chromosomes are sorted in ascending order of their C_{max} to determine the list of best ones denoted by $\mathscr{E}$ which will include those with the same lowest C_{max} values but with different classification of operations.

Table 1.

Resource and processing time assignment of $7/5/J/C_{max}$

Jobs	Resource (Processing time) Operations				
	1	2	3	4	5
1	3(5)	2(4)	5(8)	-	-
2	2(6)	4(3)	-	-	-
3	1(4)	5(3)	4(6)	2(4)	-
4	3(6)	4(2)	1(3)	5(4)	2(9)
5	1(8)	3(4)	2(3)	-	-
6	5(7)	3(2)	4(4)	1(3)	-
7	4(10)	-	-	-	-

Table 2.

Maintenance periods of $7/5/J/C_{max}$ problem

Machine M_k	$[I_k{}^1 \quad F_k{}^1]$	$[I_k{}^2 \quad F_k{}^2]$
M_1	[8 15]	[36 43]
M_2	[12 18]	[30 38]
M_3	[12 18]	[25 26]
M_4	[18 26]	[40 43]
M_5	-	-

- Secondly, each chromosome which belongs to $\mathscr{E}$ that has the lowest C_{max}, chooses one or more chromosome(s) coming from the N chromosomes of the population that has at least one T_k value better than its own T_k to form one or more couples.

Once a couple at least is built, the crossover procedure can be performed. The proposed crossover method is illustrated using a benchmark of $7/5/\mathrm{J}/C_{max}$ problem given in table 1 with adding maintenance periods presented by table 2.

3.4 Crossover Procedure

Unlike conventional crossover methods, here a new crossover procedure is given. For each couple, the main idea of the new method is illustrated as follows:

- Make a scanning of time on both chromosomes father and mother.
- At each time t, if there are several genes resulting from two chromosomes (mother and father) and having the same starting time equal to t, thus they have to be placed in the child chromosome in the order as follows: gene coming from mother before gene coming from father, making sure to not add a gene already placed. Indeed, in figure 5, the gene index coming from the mother chromosome is marked by a bold figure and the one coming from the father chromosome by an italic and grey figure.
- Make sure that all the operations are present in the child chromosome without redundancy.

After the crossover, a new classification of genes which respects automatically the sequence constraints is obtained. To respect also the resource constraints and the unavailability constraints, starting times of genes within the child chromosome are computed again. The second child chromosome is obtained by inverting the position of

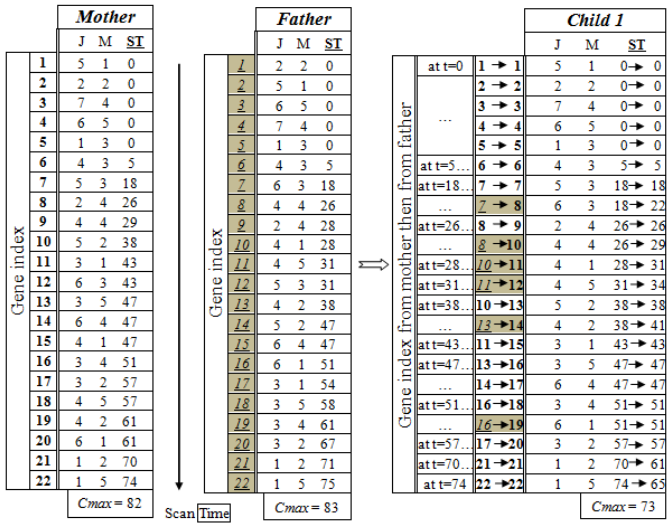

Fig. 5. Crossover procedure proposed

the parents.
This modified crossover procedure is illustrated in figure 5 where:

$$\left.\begin{array}{l} C_{max} \text{ of chromosome mother} = 82 \\ C_{max} \text{ of chromosome father} = 83 \end{array}\right\} \; C_{max} \text{ of chromosome child} = 73 \quad (T.U)$$

3.5 Special mutation

Usually, the mutation process is necessary in GA when the algorithm stack in local optimum or does not converge. So, mutation is used to diversify the GA population. Moreover, it improves drastically the GA convergence and leads good results.

In addition to the standard mutation method as commonly used in standard genetic algorithms, a new mutation method is proposed. Here, after the crossover took place, the chromosomes with the lowest C_{max} were selected. The new mutation method consists on permutation of two consecutive genes from two different jobs using the same machine which necessarily have a time lost between them. These two consecutive genes will be selected so as to reduce the time lost.

3.6 Stop criteria

The proposed program will stop either when there are no couples to create or when the lowest C_{max} does not change during a well determined number of last generations. Therefore, the best chromosomes obtained will be considered as local optima.

3.7 Running Mechanism of the First Technique MGA

The running mechanism of the proposed MGA is illustrated by figure 6 which summarizes its different steps.

4. THE SECOND PROPOSED TECHNIQUE: HEURISTIC DISPLACEMENT OF GENES (HDG)

The second technique is used to fine further the best chromosomes obtained by the first technique MGA. It is

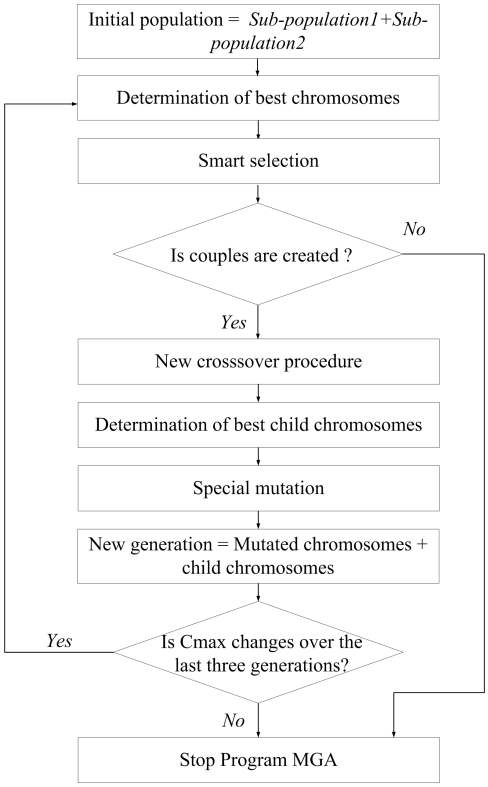

Fig. 6. Flowchart of the running mechanism of the proposed MGA

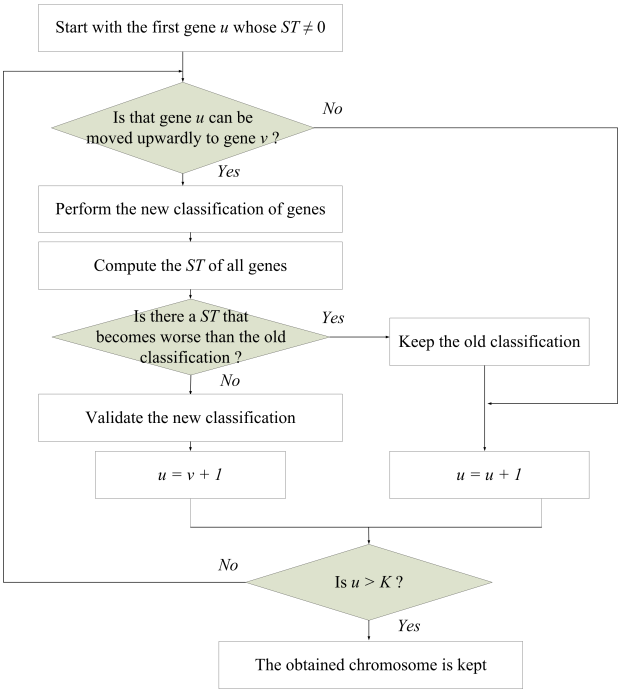

Fig. 7. Flowchart of the proposed HDG

based on performing the possible displacements of genes into these obtained chromosomes in order to improve their stating time. Note that previous respected constraints, indeed, have been taken into consideration. The choice of genes to be moved is determined by the chromosomes themselves. In fact, from top to bottom, the possible displacements of each gene will be tested so that we may

decide whether to keep or not the new position of each gene displaced. In HDG technique, the movement of the gene is submitted to a couple of criteria. First, its starting time is already improved.Second, the remaining genes in the chromosome will not have their starting time negatively changed. Consequently, C_{max} can either remain the same or enhanced.

Figure 7 represents the general flowchart of the second technique HDG.

5. THE PROPOSED NHGA RUNNING MECHANISM

The NHGA have been run on two phases. In the first phase, we applied the first technique MGA, followed by the second technique HDG on several initial populations. This is dare in order to determine the best solutions, and these later ones will be considered as local optima. Then, in the second phase, the resulting local optimal chromosomes are grouped together and again the MGA and the HDG are applied to reach the global optimal solutions for the joint JSSP.

6. EXPERIMENTAL RESULTS

This section is presented into two parts. The first part gives a simulation of an example explaining the characteristics of the proposed NHGA. The second one presents a comparative study of our NHGA with other algorithms such as conventional genetic algorithms, neural networks and some heuristics.

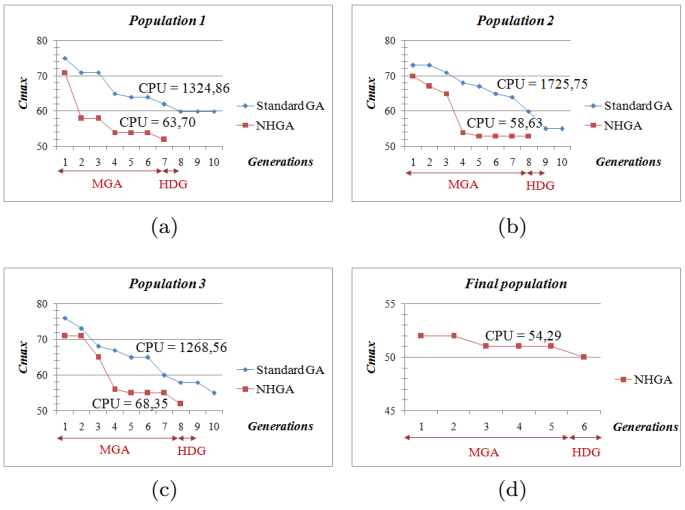

Fig. 8. Experimental results of $7/5/J/C_{max}$ problem

6.1 Simulation Example

The example will be considered is the $7/5/J/C_{max}$ problem with maintenance periods given by the corresponding Table 1 and Table 2.

In figure 8, three initial populations POP 1, POP 2 and POP 3 that have initially as best C_{max} values respectively (71, 70 and 71) were used during the first phase.

After applying the MGA and HDG on each population separately, three different C_{max} values (52, 53 and 52) of local chromosomes were obtained respectively for POP 1, POP 2 and POP 3 (fig. 8(a), fig. 8(b) and fig. 8(c)). These three local chromosomes were grouped into a final population to be used in the second phase in which MGA then HDG are again applied. This second phase leads to a "global optimal" solutions with a C_{max} value equal to 50 (fig. 8(d)).

Compared to standard GA, simulations show the effectiveness of NHGA with respect to the results of the initial populations, in particular the role of the sub-population of specific chromosomes. Thus, it provides best C_{max} values. Then, the first technique MGA displays its effectiveness to converge quickly towards a solution and the HDG can further improve the C_{max} of such solution. Also, the second phase showed its efficacy. Indeed, after grouping the solutions obtained in the first phase and applying MGA and HDG, the C_{max} value is even lower.

6.2 Comparison with Existing Algorithms

In order to evaluate the performance of the proposed NHGA versus existing methods and techniques, various tests have been performed on some known benchmarks reformulated with maintenance data (wm) in order to compare the NHGA with the other algorithms. By inspecting the recorded values on Table 3, NHGA has been compared with existing Neural Network (NN) algorithms, a heuristic for Fnaiech et al. (May 2011) and conventional GA on the $7/5/J/C_{max}$ problem and four other problems which are $2/3/J/C_{max}$ and $3/3/J/C_{max}$ given by Yahyaoui et al. (2011), $5/5/J/C_{max}$ given by Omar et al. (2006) and $6/6/J/C_{max}$ given by (Mut (1963)) and Table 4. The NHGA obtained results are clearly better than those given by NN algorithm, the heuristic for Fnaiech et al. (May 2011) and conventional GA in term of reduced C_{max} and CPU time.

In table 3, it was found that more the size of benchmark is large, the more NHGA performs better than other algorithms.

For Mt06, whereas our NHGA reaches 63 after 35 cumulative generations using five different initial populations in the first phase, standard GA could not reach after 100 generations best than 83 C_{max} value.

One of the obtained solutions of the different benchmarks is presented on the Gantt chart given by figure (9).

7. CONCLUSION

In this paper, a new heuristic method NHGA has been proposed and put into practice to solve the joint job shop scheduling problem of production and maintenance. The proposed heuristic technique consists in guiding the conventional genetic algorithm from the beginning and during all its steps with the help of choosing the suitable chromosomes in each generation and performing suitable guidance of the standard GA while trying to enhance the random choices along all the selection, crossover and mutation steps. This allows the NHGA to converge towards optimal solution in reduced number of generations and CPU time.

[2] Willems and Brandts (1995)
[3] Yahyaoui et al. (2009), Yahyaoui et al. (2011)
[4] Fnaiech et al. (May 2011)

Table 3.

With adding maintenance tasks, comparative results with standard GA, recent methods and heuristics

Examples	NN [2]		NN [3]		Heuristic [4]		Standard GA		Proposed NHGA	
	Cmax	CPU	Cmax	CPU	Cmax	CPU	Cmax	CPU	Cmax	CPU
2/3 wm	34.4	0.30	27	0.15	27	1.08	27	21.20	27	1.32
3/3 wm	26	18.56	20	0.74	21	1.40	18	55.91	18	4.40
5/5 wm	-	-	67	28.62	50	3.45	53	1354.87	50	269.08
7/5 wm	-	-	54	105.1	50	3.30	51	1268.56	50	244.97
Mt06 wm	-	-	120.2	291.1	78	6.17	83	1863.65	63	227.01

Table 4.

Maintenance periods of $6/6/J/C_{max}$ problem

Machine M_k	$[I_k{}^1 \quad F_k{}^1]$	$[I_k{}^2 \quad F_k{}^2]$	$[I_k{}^3 \quad F_k{}^3]$
M_1	[15 19]	[44 48]	-
M_2	[12 18]	[33 39]	[54 60]
M_3	[18 21]	[39 42]	[60 63]
M_4	-	-	-
M_5	[15 20]	[40 45]	[65 70]
M_6	-	-	-

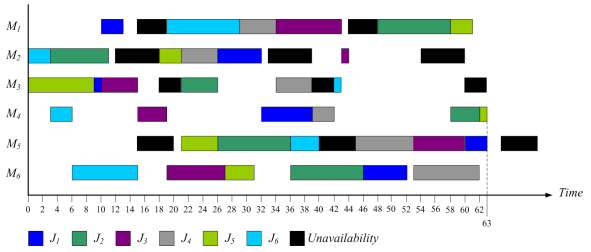

Fig. 9. Gantt chart of 6/6 wm

The obtained results display the efficiency of the proposed NHGA with respect to comparison of existing evolutionary algorithms.

As extension of the proposed NHGA, further works have recently been performed to apply it to the JSSP with predictive maintenance tasks.

REFERENCES

(1963). *Industrial scheduling*. Prentice-Hall.

Adams, J., Balas, E., and Zawack, D. (1988). The shifting bottleneck procedure for job shop scheduling. *Management Science*, 34(3), 391–401. doi:10.1287/mnsc.34.3.391.

Brucker, P., Sotskov, Y.N., and Werner, F. (2007). Complexity of shop-scheduling problems with fixed number of jobs: a survey. *Mathematical Methods of Operations Research*, 65(3), 461–481. doi:10.1007/s00186-006-0127-8.

Chiang, T.C. and Fu, L.C. (2007). Using dispatching rules for job shop scheduling with due date-based objectives. *International journal of production research*, 45, 3245–3262.

Fnaiech, N., Yahyaoui, A., and Fnaiech, F. (May 2011). New Shifting Method for Combined Production Scheduling and Maintenance Cost in Job Shop. *International Review on Modelling and Simulations (I.RE.MO.S.)*.

Gao, J., Gen, M., and Sun, L. (2006). Scheduling jobs and maintenances in flexible job shop with a hybrid genetic algorithm. *Journal of Intelligent Manufacturing*, 17(4), 493–507. doi:10.1007/s10845-005-0021-x.

Glover, F. (1990). Tabu search, part ii. *ORSA Journal on Computing*, 2, 4–32.

Jones, A., Rabelo, L., and Sharawi, A. (2001). *Survey of Job Shop Scheduling Techniques*. John Wiley & Sons, Inc. doi:10.1002/047134608X.W3352. URL `http://dx.doi.org/10.1002/047134608X.W3352`.

Lei, D. (2011). Scheduling fuzzy job shop with preventive maintenance through swarm-based neighborhood search. *International Journal of Advanced Manufacturing Technology*, 54(9-12), 1121–1128. doi:10.1007/s00170-010-2989-4.

Omar, M., Baharum, A., and Abu Hasan, Y. (2006). a job-shop scheduling problem (jssp) using genetic algorithm (ga). In *2nd IMT-GT Regional Conference on Mathematics, Statistics and Applications*, Universiti Sains Malaysia, Penang.

Ponnambalam, S.G., Aravindan, P., and Rao, P.S. (2001). Comparative Evaluation of Genetic Algorithms for Job-shop Scheduling. *Production Planning and Control*.

Schmidt, G. (2000). Scheduling with limited machine availability. *European Journal of Operational Research*, 121(1), 1 – 15. doi:10.1016/S0377-2217(98)00367-1.

Van L., P., Aarts, E., and Lenstra, J. (1992). Job shop scheduling by simulated annealing. *Operations research*, 40(1), 113–125. doi:10.1287/opre.40.1.113.

Wang, L. and Zheng, D. (2002). A modified genetic algorithm for job shop scheduling. *International Journal of Advanced Manufacturing Technology*, 20(1), 72–76. doi:10.1007/s001700200126.

Willems, T. and Brandts, L. (1995). Implementing heuristics as an optimization criterion in neural networks for job-shop scheduling. *Journal of Intelligent Manufacturing*, 6(6), 377–387. doi:10.1007/BF00124064.

Yahyaoui, A., Fnaiech, N., and Fnaiech, F. (2009). New Shifting Method for Job Shop Scheduling Subject to Invariant Constraints of Resources Availability. In *IECON: 2009 35TH Annual Conference of IEEE Industrial Electronics*, 3211–3216. IEEE Ind Elect Soc, Porto, Portugal, Nov 03-05.

Yahyaoui, A., Fnaiech, N., and Fnaiech, F. (2011). A Suitable Initialization Procedure for Speeding a Neural Network Job-Shop Scheduling. *IEEE Transactions on industrial electronics*, 58(3), 1052–1060. doi:10.1109/TIE.2010.2048290.