A hybrid genetic algorithm for parallel machine scheduling problem with consumable resources

F. BELKAID and Z. SARI
Manufacturing Engineering Laboratory of Tlemcen
University of Tlemcen
Tlemcen, Algeria
f_belkaid@yahoo.fr and z_sari@mail.univ-tlemcen.dz

F. YALAOUI
Institut Charles Delaunay, LOSI, STMR (UMR CNRS 6279)
University of Technology of Troyes
Troyes, France
farouk.yalaoui@utt.fr

Abstract—**This paper deals with the scheduling problem on identical parallel machines when each job depends on the amount of consumed resource and is characterized by different resource requirements. A typical workshop configuration is chosen for detailed study and analysis under several assumptions. This problem is known as NP-hard. To solve it, an integer linear programming based position variables and a genetic algorithm are proposed. A local search procedure is proposed to provide improved solutions. Since small instances of the problem can be solved optimally, the genetic algorithm (with and without local search) were compared to an exact resolution method which enumerates all possible solutions determined from a mathematical model. However, for medium or large instances, the proposed approaches effectiveness is checked on the basis of a heuristic. The analysis of results reveals that the hybrid genetic algorithm performs the best for different structure.**

Keywords— Scheduling, parallel machines, consumable resources, hybrid genetic algorithm, local search, makespan

I. INTRODUCTION

In the current context of international competition, it has become necessary for manufacturing enterprises to exploit efficiently all potential resources to remain competitive. The production scheduling is unquestionably one of the most important parameters that affect the performance achieved. In production system, scheduling problem consists in organizing the execution time of interdependent operations using available resources in limited quantities to achieve a production plan [1].

Production scheduling problems have different characteristics; they can be classified into several families based on several aspects. Among them, the most widely studied are single and parallel machine scheduling, flow shop, job shop.

In an industrial context, parallel machine scheduling represents an important research field. This importance is further amplified by the variety of complex configuration that can be reduced to this type of problem. Further details about this class are given of manufacturing model in [2] [3] [4].

An important part of scheduling problems studies are placed in the context where resources are always available, but this constrain cannot be satisfied in many practical situations, various resources both material and component may be unavailable for various reasons. This situation appears frequently in production environments where productivity is significantly affected.

Parallel machine scheduling problems with the presence of consumable resources are very typical in industrial production practice and have wide application background in textile industry, shoes industry and so on.

Due to the almost limitless number of different production environments, many constrains can be defined and several parameters can be considered under various perturbations. Therefore, in many production scheduling problems we often have to deal with difficult situations. Consequently, scheduling problems in these production systems are generally difficult and there are not universal methods making it possible to solve all the cases optimally [5].

Metaheuristics are a family of stochastic algorithms dedicated to solving difficult optimization problems and attempt to improve the resolution of complex situations. The success of these methods depends on their ability to find good solutions to NP-hard problems in reasonable time.

Since the selected job involve the behaviour of the system, the optimal scheduling strategy used to determine the sequence of the jobs assigned to each machine under resource constraint becomes an important issue and affect the efficiency of scheduling. The aim of this paper which extends and improves the research conducted in [6] is to propose an integer linear programming and to develop an efficient hybrid genetic algorithm for scheduling a set of jobs on identical parallel machines with consumable resources to minimize the makespan.

The paper has the following structure. Section 2 presents a literature review about scheduling problems with consumable resource. Section 3 describes our problem. Section 4 provides the proposed mathematical formulation. In section 5, a hybrid genetic algorithm is developed. Experiments are made in Section 6. Finally, section 7 is devoted to conclusions and discussions on possible extensions.

II. LITERATURE REVIEW

Recently, there has been an increasing interest in single machine problems with consumable resources. Toker et *al.* [7] treat the single machine problem with a single financial resource, they show that this problem is equivalent to a two

machine flow shop without financial resources; they applied the Johnson algorithm for minimizing the makespan. Xie [8] consider a single machine scheduling problem with multiple financial resource constraints, the author reduce this problem to the two machine flow shop scheduling problem. He shows that the LPT rule gives an optimal solution to the problem if the financial resources are consumed uniformly by all the jobs. Kaspi and Shabtay [9] treat a single machine scheduling problem where the processing time is defined by a convex decreasing resource consumption function and processing times depend on a common limited resource. The objective is to minimize the makespan and to determine the optimal job permutation and the resource allocation. Lazarev and Werner [10] address the scheduling problem on a single machine; they assume that processing times and due dates are oppositely ordered. They give some polynomials solvable special cases for minimizing total tardiness. Janiak et *al.* [11] study a single machine problem, in which the processing times are controllable through the consumption of a non-renewable resource. Carrera et *al.* [12] discuss the problem in a single machine with consumable resources and fixed delivery dates. The authors conduct a study on the complexity of these problems and they show that they are NP-hard. They use a branch and bound to solve them. A recent analysis about the complexity of scheduling problems with a consumable resource is made by Gafarov and Lazarev [13].

Although, scheduling problems were studied in many research articles, little progress has been made for the objective of minimizing makespan in parallel machine scheduling problems with non renewable resources. It is shown that, the problem for minimizing the makespan is NP-hard even for the two-machine problem [14] [15]. Slowinski [16] addresses a scheduling problem on parallel machines with resource constraints (financial constraints) and preemptive jobs. The author proposes an exact procedure composed of two phases using linear programming to minimize the makespan. Ling et *al.* [17] dealt with a scheduling problem on parallel machines to minimize the makespan when the processing time depends on the amount of resource consumed. They propose a heuristic to solve this problem.

These types of problems are relatively complex and several resolution methods may be impractical in many cases of these scheduling problems.

Given the success of metaheuristics in the context of solving NP-hard problems and wherein production systems are complex, many researchers have employed them to solve parallel machine scheduling problems because of their good results proved in the resolution of several complex problems and their particularly successful performances on the makespan objective.

Metaheuristics have been widely used in parallel machines scheduling problem. Min and Cheng [18] proposes a genetic algorithm for minimizing the makespan on identical parallel machines scheduling problem, which is efficient for large scale problems. The obtained results are compared with simulated annealing and other available heuristics to demonstrate the good performance of genetic algorithm. Serifoglu and Ulusoy [19] study a parallel machines scheduling problem with independent jobs to minimize earliness-tardiness. They propose a genetic algorithm to solve this problem. Gupta and Ruiz-Torres [20] provide a LISTFIT algorithm for solving the parallel-machine scheduling problem of minimizing makespan. Mendes et *al.* [21] consider a parallel machines scheduling problem for minimizing the makespan. The authors propose a taboo search and memetic approach based local search procedures to solve this problem. Fowler et *al.* [22] propose a hybrid genetic algorithm for a parallel machines scheduling problem to minimize the makespan with dependent setups. The first step of the hybrid genetic algorithm consists to assign jobs to machines and the second step consists in dispatching rules for scheduling the individual machines. Wilson et *al.* [23] treat a parallel machines scheduling problem with a common batch setup time to minimize the makespan. To solve this problem, the authors propose a batch splitting and a genetic algorithm. A heuristic is integrated in the genetic algorithm to improve the results.

To address the above parallel machines scheduling problem with resources constraints some researchers used metaheuristics. Liu et *al.* [24] use a genetic algorithm for minimizing the makespan for a scheduling problem on parallel machines. Chaudhry [25] proposes a genetic algorithm to minimize the cycle time of a set of jobs and the allocation of workers for a scheduling problem on parallel machines. Kai Li et *al*. [26] propose a simulated annealing algorithm for the identical parallel machine scheduling problem to minimize the makespan with controllable processing times, he assume that the processing times of each jobs are linear decreasing functions of the consumed resource. Belkaid et *al.* [6] propose a genetic algorithm to find a solution to the parallel machine scheduling problem with consumable resources to minimize makespan, they compare the algorithm with an exact method for small size problems and with a heuristic for large size problems. The obtained results show the effectiveness of the proposed algorithm.

Through literature review we can notice that there are little works on metaheuristics adaptation for solving parallel machines scheduling problem with consumable resources. Although research efforts were deployed to scheduling problems with consumable resources, they are essentially concentrated on classical situations which, generally, cannot be immediately applied to complex configurations.

Thus, the consideration of resource constrain in parallel machines, which can be modified and applied to several different cases, is common in many operations management and is of crucial importance for manufacturing systems. An efficient organization of resources may result in further efficiencies and provide many benefits, such as improved production rate and reduced idle time.

The aim of this paper is firstly to improve the behavior of the algorithm presented in [6] by adding a local search and secondly is to propose an integer linear programming model for parallel machines scheduling problem with critical resource consumption function modeling.

III. PROBLEM DESCRIPTION

In this paper, we focus on a parallel machines environment when each job depends on an amount of resource consumption to be carried out.

Each resource is allocated to activities at their start time. Every job can be carried out on each machine when all necessary resources are available. The arrival of each component is represented by a staircase curve-shaped. All machines are ready from time zero onwards and can process only one job at a time. The machines are identical so each job can be processed on any machine. Preemption is not permitted. In addition, storage buffers are located before each machine (see figure 1).

Several researchers have proposed various priority rules to provide flexibility in scheduling and to increase system performance. Among the most successful rules we can cite the FIFO rule which is used in this work. Several other rules have been proposed, for more details, the reader can refer to Liu and Wu [27].

The criterion to minimize is the makespan. This problem is NP-Hard and the resolution of this class of problem has not been extensively studied.

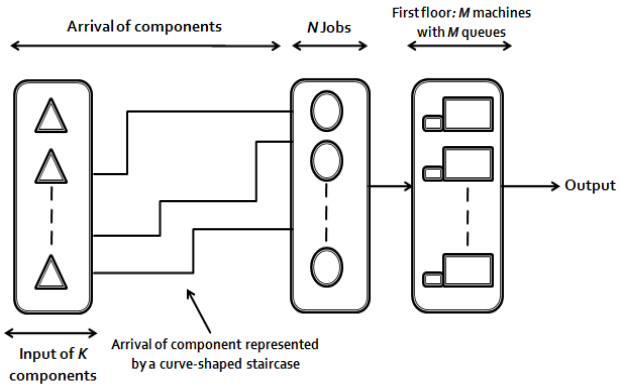

Figure 1. Problem representation

IV. MATHEMATICAL FORMULATION

A mathematical program represents an optimal way to solve makespan minimization problem with few jobs, machines and components. To formulate the considered problem we exploit a representation based position variables, indicating if a job is at position p in scheduling or not. This representation reflects the real functioning of the considered problem and avoids the large number of variables and constraints indexed by time. After studying mathematical models proposed in: [28], [29], [30], [31] and [12], we propose an integer linear model to minimize C_{max}. The model can be expressed as follows:

Parameters:

- n : number of tasks
- m : number of machines
- c : number of components
- T_1 : time of the first arrival
- T_{last} : time of the last arrival
- j : index of the task, j = 1, ..., n
- i : index of the machine, i = 1, ..., m
- k : index of the resource, i = 1, ..., c
- t : index of components arrival.
- n_i : number of tasks assigned to machine i
- p : position of job in a machine, p=1,…n_i
- p_j : operating time of job j
- d_{i0} : start date scheduling
- d_{ip}: start date of job processing on machine i in position p.
- p_{ip} : processing time of job i on the machine in position p.
- f_{ip} : completion date of job processing on machine i in position p.
- c_{jk} : quantity of component k that job j consumes
- c_{ipk} : number of component k that job which is on machine i in position p consumes.
- A_{tk} : total component k arrival until time t.
- C_{max} : makespan

Objective function:

$$Min\ C_{max} \tag{1}$$

Constraints:

$$\sum_{j=1}^{n} X_{jip} = 1 \quad \forall\, i = 1,2,\dots,m \quad \forall\, p = 1,2,\dots,n_i \tag{2}$$

$$\sum_{i=1}^{m}\sum_{p=1}^{n_i} X_{jip} = 1 \quad \forall\, j = 1,2,\dots,n \tag{3}$$

$$p_{ip} = \sum_{j=1}^{n} X_{jip} p_j \quad \forall\, i = 1,2,\dots,m \quad \forall\, p = 1,2,\dots,n_i \tag{4}$$

$$d_{i0} = 0 \quad \forall\, i = 1,2,\dots,m \tag{5}$$

$$d_{i(p-1)} + p_{ip} \le f_{ip} \quad \forall\, i = 1,2,\dots,m \quad \forall\, p = 1,2,\dots,n_i \tag{6}$$

$$d_{ip} = f_{ip} - p_{ip} \quad \forall\, i = 1,2,\dots,m \quad \forall\, p = 1,2,\dots,n_i \tag{7}$$

$$c_{ipk} = \sum_{j=1}^{n} X_{jip} c_{jk} \quad \forall\, i = 1,2,\dots,m \quad \forall\, p = 1,2,\dots,n_i \quad \forall\, k = 1,2,\dots,c \tag{8}$$

$$\sum_{i=1}^{v}\sum_{p=1}^{w} c_{ipk} \le \sum_{t=T_1}^{T_{last}} A_{tk} * Y_{iwt} \quad \forall\, v = 1,2,\dots,m \quad \forall\, w = 1,2,\dots,n_i \quad \forall\, k = 1,2,\dots,c \tag{9}$$

$$M * (Y_{ipt} - 1) \le d_{ip} - t \quad \forall\, v = 1,2,\dots,m \quad \forall\, t = t_1,\dots,T_{last} \tag{10}$$

$$C_{max} = \max_i f_{ip} \quad \forall\, i = 1,2,\dots,m \quad \forall\, p = 1,2,\dots,n_i \tag{11}$$

$$X_{jip} = \begin{cases} 1 \text{ if the job j is scheduled in position p on machine i} \\ 0\ otherwise \end{cases} \tag{12}$$

$$Y_{ipt} = \begin{cases} 1\ si\ d_{ip} \ge t \\ 0\ otherwise \end{cases} \tag{13}$$

Objective function minimizes the makespan (1). Constraint (2) confirms that each job is assigned to exactly one machine i at position p. Constraint (3) ensures that each job is assigned only once on these machines. Constraint (4) calculates the operation time of the job in position p on machine i. Constraint

(5) ensures that the beginning of the schedule is at time 0. Constraint (6) calculates the completion date of jobs processing at each position p. Constraint (7) calculates the start date of jobs on machine i in position p. Constraint (8) calculates the quantity of components consumed by the job in position p on machine i. Constraint (9) verifies that, for each component the quantity consumed by a job at position p on machine i is less than or equal to the available components. And it ensures that the start date of job processed on machine i in position p is greater than the arrival date of the component. Constraint (10) make the link between the variable Y_{ipt} and execution starting of job on position p. Constraint (11) describes the makespan, it is equal to the completion time of the last job. Constraint (12) indicates that the binary variable X_{jip} is equal to 1 if the job j is in position p on machine i and 0 otherwise. Constraint (13) is a binary variable equal to 1 if $d_{ip} \geq$ t and 0 otherwise.

Although, small instances of the problem can be solved optimally with mathematical model, it necessary to develop an efficient algorithm for large size problems.

V. GENETIC ALGORITHM

This method was formalized by Holland [32]. It is a metaheuristic based on the use of mechanisms inspired by natural system and evolution theory. It is an iterative algorithm; therefore it is very hard to obtain the optimal solution at the first iteration.

To solve our problem we develop a genetic algorithm which has the following structure:

- Coding: The chromosome is encoded in a table consisting of two lines. The first line indicates index of job and the second line represents the index of machine i, which the job j is assigned. This allows for chromosomes to illustrate the machines selected for each job in the system.
- Initial population: Initial chromosomes are obtained randomly. Each chromosome is represented by a random assignment of jobs to machines. In this work, an individual represents the sequence of the tasks affected to the same machine.
- Fitness function: The fitness function is represented by the makespan and it is calculated for each chromosome in the current generation.
- Selection: In this step, we choose chromosomes for reproduction through evaluation of their makespan. The technique used for selection in this work is elitist technique.
- Crossover: In this step, we choose 2 chromosomes randomly and a two point crossover is applied to produce two new chromosomes. The crossover probability is equal to 1.
- Mutation: Following a probability, a gene of the new chromosomes is randomly changed to maintain the diversity of population through the perturbations caused in the solution. In our problem, the mutation is done by changing assignment of jobs to machines. The mutation probability is equal to 0.2.
- Constitute the new generation, we select the next generation composed of the best chromosomes based on their fitness function.
- Stopping criterion: A stopping criterion is applied. It is represented by a desired number of iterations and the best chromosome which characterizes the most excellent solution is given as output.

To improve the performance of this algorithm, a local search method is proposed. This method is based on the exploration of the neighborhood of a solution.

The pseudo code of the local search procedure is given as follows:

Step 1: For each individual (initial condition)
Step 2: While stop criterion is not satisfied
Step 3: Changing assignment of some jobs to machines (find another nearby solution)
Step 4: If this solution is smaller than the best solution **then** actualize the best solution.
Step 5: End while
Step 6: End for

The functioning of the proposed genetic algorithm can be summarized as follows:

Step 1: If there are a list of jobs not scheduled then
Step 2: Generate a random population (initial population).
Step 3: For each chromosome
Step 4: Calculate the fitness of chromosome which is represented by the makespan.
Step 5: Evaluate the fitness of this chromosome.
Step 6: If this solution is smaller than the best solution then actualize the best solution.
Step 7: End for
Step 8: While stop criterion is not satisfied
Step 9: Apply the selection operator.
Step 10: Apply the crossover operator.
Step 11: Apply the local search.
Step 12: Apply the mutation operator.
Step 13: For each chromosome
Step 14: Calculate the fitness of chromosome which is represented by the makespan.
Step 15: Evaluate the fitness of this chromosome.
Step 16: If this solution is smaller than the best solution then actualize the best solution.
Step 17: End for
Step 18: Constitute the next generation.
Step 19: End while
Step 20: End if

VI. COMPUTATIONAL RESULTS

The objective of the computational experiments is to analyze the impact of consumable resource on system performances and to evaluate the effectiveness of the proposed Hybrid Genetic Algorithm (HGA).

The performance of the results is measured in terms of both solution quality and computational time.

All experimental tests are performed on Core (TM) i3 CPU with 2.13 GHz and 4.00 Go of Ram. The genetic algorithm and heuristic were coded in Java language and the integer linear program is solved on linear programming solver CPLEX.

The Genetic Algorithm (GA) is compared with an Exact Method (EM) in which a complete enumeration is applied, an Integer Linear Programming (ILP) and a Heuristic (He).

- **Heuristic description**

The heuristic applied in this work has been used by Carrera et *al.*, [12] in single machine problem with consumable resources which consists in: Ordering jobs in decreasing order of their relative processing time and the sum of components that consume. We generalize this heuristic for parallel machines scheduling problem. This technique is applied for each component independently.

Experiments imply three tests for makespan minimization problem. These tests are listed in table 1-3 for each experimental combination. Each test inquires three variables which are the number of jobs (n), the number of machines (m) and the number of components (k). The processing time of each job *j* is generated by a uniform distribution *U* [1, 50]. The total resource required is always less than or equal to the total amount of resources coming to the system.

Tables 1-3 synthesize tests for small, medium and large instances problems. The first column of tables contains the studied instance, the second shows the CPU_{time} obtained by all used methods. GAP1 represents the relative average variation between the best solution and the solution obtained by heuristic. GAP2 is the relative average variation between the optimal solution and GA and finally, GAP3 shows the relative average variation between the best solution and the HGA.

TABLE I. EXPERIMENTAL RESULTS FOR SMALL SIZE PROBLEMS

Problem (n, m, k)	**CPU_{time} (s)**					**GAP**		
	EM	***ILP***	***He***	***GA***	***HGA***	***GAP1***	***GAP2***	***GAP3***
(4, 2, 2)	0.06	0.06	0.03	0.007	0.09	0	0	0
(4, 2, 3)	0.06	0.05	0.03	0.03	0.05	0	0	0
(4, 3, 2)	0.03	0.04	0.02	0.01	0.04	0	0	0
(4, 3, 3)	0.01	0.01	0.02	0.02	0.04	0	0	0
(6, 2, 2)	15.8	0.2	0.05	0.04	0.06	0.09	0	0
(6, 2, 3)	22	0.4	0.04	0.04	0.08	0	0	0
(6, 3, 2)	36.4	1 s	0.05	0.03	0.07	0.12	0	0
(6, 3, 3)	/	1 s	0.03	0.04	0.06	0	0	0

As observed in table 1, the CPU_{time} for exact method starts to increase exponentially when the number of jobs, machines and components increase. For instance with 6 jobs, 3 machines and 3 components, complete enumeration does not give results but integer linear programming provide exact solution.

According to these results, CPU_{time} remains small for heuristic, GA and HGA and it starts to increase slowly for the integer linear programming when the number of machines is greater than 2.

Furthermore, GAP1 is different from 0 for instance 2, which implies that other methods give better results compared to heuristic. Note that the makespan obtained by GA and HGA is the same.

TABLE II. EXPERIMENTAL RESULTS FOR MEDIUM SIZE PROBLEMS

Problem (n, m, k)	**CPU_{time} (s)**			**GAP**		
	He	***GA***	***HGA***	***GAP1***	***GAP2***	***GAP3***
(10, 6, 2)	0.05	0.18	0.21	0	0	0
(10, 6, 3)	0.02	0.09	0.12	0.06	0	0
(10, 8, 2)	0.02	0.09	0.11	0	0	0
(10, 8, 3)	0.01	0.09	0.11	0	0	0
(20, 6, 2)	0.05	0.20	0.21	0	0.1	0
(20, 6, 3)	0.04	0.20	0.22	0	0.04	0
(20, 8, 2)	0.04	0.22	0.24	0	0.11	0
(20, 8, 3)	0.03	0.22	0.24	0	0.06	0

In table 2, we notice that the CPU_{time} is practically the same for all techniques.

According to these results, the GAP2 start to increase when the number of jobs is greater than 20, but GAP3 and GAP1 are equal to 0, which imply that HGA improve the results obtained by GA.

Furthermore, heuristic give the same results in comparison with the HGA for instances with 20 jobs.

TABLE III. EXPERIMENTAL RESULTS FOR LARGE SIZE PROBLEMS

Problem (n, m, k)	**CPU_{time} (s)**			**GAP**		
	He	***GA***	***HGA***	***GAP1***	***GAP2***	***GAP3***
(50, 10, 2)	0.54	0.92	0.95	0	0.18	0
(50, 10, 3)	0.54	0.64	0.68	0	0.02	0
(50, 15, 2)	0.55	0.67	0.71	0	0.07	0
(50, 15, 3)	0.56	0.65	0.69	0	0.18	0
(100, 10, 2)	0.84	1.22	1.25	0	0.2	0
(100, 10, 3)	1.10	1.29	1.32	0	0.06	0
(100, 15, 2)	1.35	1.25	1.28	0	0.14	0
(100, 15, 3)	1.37	1.23	1.28	0	0.11	0

The results concerning large instances are given in table 3. As seen in this table, the CPU_{time} is practically the same for all methods.

According to these results, the difference between GA and HGA starts to increase when the number of jobs, machines and components increase. We notice that heuristic has the same behavior as HGA for large size problems.

Furthermore, hybrid genetic algorithm improves the results obtained by genetic algorithm. Therefore, we can state that hybrid genetic algorithm performs well when the problem size increases this is due to the local search that provides diversification. Finally, we can conclude that the change introduced has led to an improvement of genetic algorithm.

VII. CONCLUSION

This paper examines the problem of minimizing the makespan of scheduling in a parallel machine environment with consumable resources. Since our problem is NP-Hard, we have proposed a genetic algorithm combined to a local search method to improve its performance. In addition, an integer linear programming formulation is developed.

To validate the results of the proposed hybrid genetic algorithm, we have compared it with the mathematical model for small instance and with heuristic for large instance. The obtained results confirm that HGA is better for all problem sizes and it can be adapted for a large variety of combinatorial optimization problems.

Further perspective can be to propose lower bounds and to develop a branch and bound algorithm to solve problems of moderate sizes. Other research works may be to apply other priority rules to manage the buffer queues and to quantify their impact on the behavior of the system.

REFERENCES

[1] J. Erschler, G. Fontan and F. Roubellat, "*Encyclopédie du management – Ordonnancement en ateliers spécialisés*," Helfer et Orsoni,Vuibert, Paris, 2 : 208-229, 1992.

[2] B. Chen, C. N. Potts and G. J. Woeginger, "A review of machine scheduling: Complexity, algorithms and approximability," In: D. Z. Du, P. M. Pardalos (Eds.), Handbook of Combinatorial Optimization, Kluwer Academic Publishers, Dordrecht, pp. 21–169, 1998.

[3] E. Mokotoff, "Parallel machine scheduling problems: A survey," Asia-Pacific Journal of Operational Research, vol. 18, pp. 193-242, 2001.

[4] A. Allahverdi, C. T. Ng, T. C. E. Cheng, and M. Y. Kovalyov, "A survey of scheduling problems with setup times or costs," European Journal of Operational Research, 187(3):985–1032, 2008.

[5] M. R. Garey and D. S. Johson, "Computers and intractability a guide of the theory of NP- completeness," W.H. Freeman and company, San Fransisco, 1979.

[6] F. Belkaid, F. Yalaoui and Z. Sari, "A genetic algorithm for parallel machine scheduling with consumable resources to minimize makespan. The 4th International Conference on Metaheuristics and Nature Inspired Computing, META'2012. Sousse, Tunisia – October, 2012.

[7] A. Toker, S. Kondakci and N. Erkip, "Scheduling under a non-renewable resource constraint," Journal of the Operational Research Society, 42(9):811–814, 1991.

[8] J. Xie, "Polynomial algorithms for a single machine scheduling problems with financial constraints. Operations Research Letters, 21:39–42, 1997.

[9] M. Kaspi and D. Shabtay, "Convex resource allocation for minimizing the makespan in a single machine with job release dates," Computers and Operations Research, 31:1481-1489, 2004.

[10] A. A. Lazarev and F. Werner, "Algorithms for Special Cases of the Single Machine Total Tardiness Problem and an Application to the Even-Odd Partition Problem," Mathematical and Computer Modelling, Vol. 49, No. 9-10, 2061 – 2072, 2009.

[11] A. Janiak, C.N. Potts, T. Tautenhahn, "Single Machine Scheduling with Nonlinear Resource Dependencies of Release Times," Abstract, 14th Workshop on Discrete Optimization, Holzhau/Germany, May 2000.

[12] S. Carrera, M. C. Portmann and W. Ramdane Cherif, "Scheduling supply chain node with fixed components arrivals and two partially flexible deliveries," 5th International Conference on Management and Control of Production and Logistics. France, 2010.

[13] E. R. Gafarov and A. A. Lazarev, "Single machine scheduling with a non-renewable financial resource," Working paper. 2010.

[14] J. K. Lenstra, A. H. G. Rinnooy Kan and P. Brucker, "Complexity of Machine Scheduling Problems," Annals of Discrete Mathematics, Vol. 1, pp.342-362, 1977.

[15] R. Sethi, "On the complexity of mean flow time scheduling," Mathematics of Operations Research, 2(4), 320-330, 1977.

[16] R. Slowinski, "Preemptive scheduling of independent tâches on parallel machines subject to financial constraints," European Journal of Operational Research;15:366-373, 1984.

[17] H. S. Ling and Y. L. Chun, "Scheduling parallel machines with resource-dependent processing times," International Journal of Production Economics, 2009; Volume 117, Pages 256-266, 2009.

[18] L. Min and W. Cheng, "A genetic algorithm for minimizing makespan in the case of scheduling identical parallel machines," Artificial Intelligence Engineering, 13, 399-403, 1999.

[19] S. F. Serifoglu and G. Ulusoy, "Parallel machine scheduling with earliness and tardiness penalties," Computers and Operations Research 26 (8), 773–787, 1999.

[20] J. N. D. Gupta and J. Ruiz-Torres, "A listfit heuristic for minimizing makespan on identical parallel machines," Production Planning and Control, 12(1), 28-36, 2001.

[21] A. S. Mendes, F.M. Muller, P.M. França and P. Moscato, "Comparing meta-heuristic approaches for parallel machine scheduling problems," Production Planning and Control 13, 143-154, 2002.

[22] J. W. Fowler, S. M. Horng and J. K. Cochran, "A hybridized genetic algorithm to solve parallel machine scheduling problems with sequence dependent setups," International Journal of Industrial Engineering: Theory Applications and Practice 10, 232-243, 2003.

[23] A. D. Wilson, R. E. King and T.J. Hodgson, "Scheduling non-similar groups on a flow line: Multiple group setups," Robotics and Computer-Integrated Manufacturing 20, 505-515, 2004.

[24] M. Liu, C. Wu, "A genetic algorithm for minimizing the makespan in the case of scheduling identical parallel machines," Artificial Intelligence in Engineering; 13, 399–403, 1999.

[25] I. A. Chaudhry, "Minimizing flow time for the worker assignment problem in identical parallel machine models using GA," International Journal of Advanced Manufacturing Technology; DOI 10.1007/s00170-009-2323-1, 2009.

[26] Kai Li, Ye Shi, Y. Shan-lin and C. Ba-yi, "Parallel machine scheduling problem to minimize the makespan with resource dependent processing times, " Applied Soft Computing: pp, 5551-5557, 2011.

[27] M. Liu and C. Wu, "Genetic algorithm using sequence rule chain for multiobjective optimization in reentrant micro electronic production line, "Robotics and Computer Integrated Manufacturing, pp. 225 – 236, 2004.

[28] Marc Sevaux and Philippe Thomin, "Heuristics and metaheuristics for a parallel machine scheduling problem: a computational evaluation," Research Report 01-2-SP (extended version of report 01-1-SP), 2001.

[29] A. H. Gharehgozli, R. Tavakkoli-Moghaddam and N. Zaerpour, "A fuzzy-mixed-integer goal programming model for a parallel-machine scheduling problem with sequence-dependent setup times and release dates," Journal of Industrial Engineering International, Vol. 4, No. 7, 38-47, July 2008.

[30] J. Carlier, A. Moukrim, and H. Xu, "The project scheduling problem with production and consumption of resources: A list-scheduling based algorithm, " Discrete Applied Mathematics, 157(17): 3631-3642, 2009.

[31] Xiaohui Li, Lionel Amodeo, Farouk Yalaoui and Hicham Chehade, "Metaheuristiques multi objectif pour un problème d'ordonnancement de machines parallèles," 8e Conférence Internationale de MOdélisation et SIMulation - MOSIM'10 - Hammamet – Tunisie, 2010.

[32] J. H. Holland, "Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control and artificial intelligence," University of Michigan Press, Ann Arbor, MI., 1975.