Flexible Job-shop Scheduling Problems Resolution Inspired from Particle Swarm Optimization

Hela Boukef [1, 2], Mohamed Benrejeb[1], Pierre Borne[2]

[1] LARA, École Nationale d'Ingénieurs de Tunis

BP 37, Le Belvédère 1002 Tunis, Tunisie

[2]LAGIS, École Centrale de Lille, Cité scientifique

BP 48, 59651 Villeneuve d'Ascq Cedex, France

Abstract: A new algorithm inspired from particle swarm optimization method is successfully implemented for flexible job-shop work-shop optimization problems. Its efficiency for solving combinatory problems is comparable to the genetic algorithms one taking into account the Makespan criterion.

Keywords: Particle Optimization Method (SPO), Flexible Job-Shop Problem (FJSP), Makespan.

Hela Boukef graduated from "Institut Supérieur de Gestion de Tunis" in 2003 and obtained the Master of automatic and signal treatment in 2006 at the "Ecole Nationale d'Ingénieur de Tunis". She is currently preparing the Ph.D. degree in automatic and computer science within the framework of LAGIS-EC-Lille and LARA-ENIT cooperation. Her research is related to optimization methods for discrete events systems, computer science and operational research.

Mohamed Benrejeb has obtained the Diploma of "Ingénieur IDN" (French "Grande Ecole") in 1973, the Master degree of Automatic Control in 1974, the PhD in Automatic Control of the University of Lille in 1976 and the DSc of the same University in 1980. He is currently a full Professor at the Ecole Nationale d'Ingénieurs de Tunis and an invited Professor at the Ecole Centrale de Lille. His research interests are in the area of analysis and synthesis of complex systems based on classical and non conventional approaches.

Pierre Borne received the Master degree of Physics in 1967, the Masters of Electronics, of Mechanics and of Applied Mathematics in 1968. The same year he obtained the Diploma of "Ingénieur IDN" (French "Grande Ecole"). He obtained the PhD in Automatic Control of the University of Lille in 1970 and the DSc of the same University in 1976. He became Doctor Honoris Causa of the Moscow Institute of Electronics and Mathematics (Russia) in 1999, of the University of Waterloo (Canada) in 2006 and of the Polytechnic University of Bucarest (Romania). He is author or co-author of about 200 Journal articles and book chapters, and of 34 plenary lectures and of more than 250 communications in international conferences. He has been the supervisor of 68 PhD thesis and is author of 20 books. He is Fellow of IEEE and has been President of the IEEE/SMC society in 2000 and 2001. He is presently Professor "de classe exceptionnelle" at the Ecole Centrale de Lille and director of the French pluriformations national group of research in Automatic Control.

1. Introduction

The importance of scheduling has increased in recent years due to the growing consumer demand for variety, reduced product life cycles, changing markets with global competition and rapid development of new processes and technologies [Hu and al, 06].

Scheduling problems are part of strong combinatory optimization problems. Many applications, varying from metallurgy, chemistry, agro-food [Tangour, 06], [Tangour, 06] or pharmaceutical industries [Boukef and al, 06], [Boukef and al, 07] can be treated by using heuristics and metaheuristics for their resolution.

Among the used metaheuristics, simulated annealing [Kirkpatrick, 83], tabu search [Glover, 89], genetic algorithms [Holland, 75] and ant colony [Colorni and al, 91] proved their performances.

But nowadays, a new optimization method is being used and is given satisfying results. This method proposed by Kennedy and Eberhart [Kennedy and Eberhart, 95] is the Particle Swarm Optimization method.

In fact, in 1995, J. Kennedy and R. Eberhart, motivated by bird flocking observation, proposed a new algorithm for representing social behaviour of artificial agents and, then created the Particle Swarm Optimization (PSO). Since 2000, PSO has been growing rapidly [Liao and al, 07] and

has been applied successfully to continuous nonlinear functions, neural networks [Van de Bergh and Engelbrecht, 00], etc.

The PSO functioning makes it classified among iterative methods (progressive approach of finding optimal solution) and stochastic ones. Its aim is to improve existing states by moving partially at random and partially according to some defined rules in order to reach the global solution [Clerc, 99].

Most of the research on PSO took into account continuous optimization problems but the studies on discrete ones and particularly on flow-shop [Lian and al, 06], [Lian and al, 06], [Lian and al, 07] and job-shop [Sha and Hsu, 06], [Xia and Wu, 05] scheduling problems are very few.

The principal scope took into account in these types of scheduling problems is the Makespan minimization, even if some authors are interested in total tardiness [Tasgetiren and al, 04].

The problem treated in this paper is dealing with flexible job-shop work-shop scheduling with Makespan optimization objective.

First, flexible job-shop problems are introduced. Next, particle swarm optimization method is presented and a new algorithm inspired from it is proposed for flexible job-shop scheduling problems. Three examples are, then, treated. The two first ones deal with mono-operation flexible job-shop problem and the third with multi-operation flexible job-shop problem. In the last part, a comparison between the obtained results and those issued from genetic algorithm method is proposed, proving, thus, the performance of this new method.

2. Flexible Job-shop Problem Presentation

The flexible job-shop scheduling problem (FJSP) is known in the literature as one of the hardest optimization problems [Saad et al, 07]. Many studies have been done on this kind of problems, [Filip et al, 83], [Mesghouni et al, 96], [Mesghouni et al, 98], [Liouane et al, 07], [Saad et al, 08].

The difficulty of FJSP suggests the adoption of metaheuristic methods producing reasonably good schedules in a reasonable time, instead of looking for an optimal solution.

The FJSP may be formulated as follows [Saad et al, 07]:

- consider a set of n jobs which are carried out by m machines M_k, $k = 1, 2, \ldots, m$,
- each job J_j consists of a sequence of n_j operations $O_{i,j}$, $i = 1, 2, \ldots, n_j$,
- each routing has to be performed to achieve a job,
- the execution of each operation i of a job J_j requires one resource selected from a set of available machines,
- the assignment of the operation $O_{i,j}$ to the machine M_k entails the occupation of the latter one during a processing time, noted $p_{i,j,k}$.

The FJSP presents two difficulties. The first one is to assign each operation $O_{i,j}$ to a machine M_k. The second one is the computation of the starting times $t_{i,j}$ and the completion time $tf_{i,j}$ of operation $O_{i,j}$.

In this study, we considerate the minimization of makespan criteria for the following tables 2, 3 and 4 benchmark.

3. Particle Swarm Optimization

PSO as an optimization tool provides a population-based search procedure in which individuals called particles, change their position (state) with time. In a PSO system, particles fly around in a multidimensional search space. During flight, each particle adjusts its position according to its

own experience, and according to the experience of a neighbouring particle, making use of the best position encountered by itself and its neighbours.

This is similar to the human behaviour in making decisions where people consider their own best past experience and the best experience of how the other people around them have performed [Kennedy and Eberhart, 95]. Thus, the PSO system can combine Local Search (LS) methods with global search methods (metaheuristics), attempting to balance exploration and exploitation.

Many similarities exist between evolutionary type methods and PSO, the latter is different because it does not use the selection operation that choose the individuals that are kept into the next generation and the members of the entire population are maintained through the search procedure so that information is socially shared among individuals to direct the search towards the best found positions in the search space [Deroussi and al, 06].

3.1. Continuous particle swarm optimization

The basic principles in classical PSO are very simple. A swarm, which contains a set of particle, is initially moving into a search space. Each particle of the swarm has five characteristics [Kennedy and Eberhart, 95]:

- its position,
- its velocity,
- the objective function value for its position,
- its neighbours best position and the associated objective function value,
- its best previous position.

The relative notations to these characteristics are expressed as follow:

V_i^t : flying velocity of particle i at iteration t

X_i^t : current position of particle i at iteration t

P_i^t : best previous position of particle i at iteration t

G_i^t : neighbours best position of particle i at iteration t

At a specific time, each particle has to make a choice between:

- following its own way and keep its current position,
- taking into account its best previous position,
- taking into account its best neighbour's position.

The possibilities quoted above, can be formulated by the following expressions

$$V_i^{t+1} = c_1 V_i^t + c_2\left(P_i^t - X_i^t\right) + c_3\left(G_i^t - X_i^t\right) \quad (1)$$

$$X_i^{t+1} = X_i^t + V_i^{t+1} \quad (2)$$

with:

c_1, c_2, c_3 are coefficients that indicates the importance given to each expression and $c_1 + c_2 + c_3 = 1$.

3.2. Discrete particle swarm optimization

The PSO has proved its efficiency in solving continuous optimization problems [Clerc, 99], [Shelokar et al, 07],… but in the major part of production optimization, the problems treated are discrete ones. Some authors treated discrete optimization problems [Lian and al, 06], [Lian and al, 07], [Liao and al, 07], [Shi and al, 07],... relative to flow-shop scheduling problems. In this article, the PSO optimization for flexible job-shop scheduling problem is considered.

3.2.1. Problem formulation

A new formulation is necessary to move from continuous PSO to discrete one. The notations related to this kind of problem can be expressed as follows:

O_{ij}^{f} : operation i of a job j for a particle f

$m_{ij}^{f}(t)$: selected machine for operation O_{ij}^{f} production at iteration t for present particle f position

$P^{f}(t)$: present position of particle f at iteration t

$P_{M}^{f}(t)$: best known position of particle f neighbours at iteration t

$P_{m}^{f}(t)$: best known position of particle f at iteration t

$[m_{ij}^{f}(t)]_{m}$: selected machine for operation O_{ij}^{f} production at iteration t for best known particle f position at iteration t

$[m_{ij}^{f}(t)]_{M}$: selected machine for operation O_{ij}^{f} production at iteration t for best known particle f position neighbours at iteration t

Δm_{ij}^{f} : changes applied to particle f for machines affectation

μm_{ij}^{f} : mutation vector applied to particle f to allow position changing,

α, β, γ : confidence coefficients

For each individual, the changes done to move from a position to another must respect the following formulations:

$$P^{f}(t+1) = P^{f}(t) + \Delta m_{ij}^{f} \tag{3}$$

with:

$$\Delta m_{ij}^{f} = f(\mu_{ij}^{f}, m_{ij}^{f}) \tag{4}$$

and :

$$\mu m_{ij}^{f} = \alpha[m_{ij}^{f}(t)] + \beta[(m_{ij}^{f}(t))_{m} - m_{ij}^{f}(t)] + \gamma[(m_{ij}^{f}(t))_{M}) - m_{ij}^{f}(t)] \tag{5}$$

Presenting a particle structure means presenting an affectation example of considered operations O_{ij}^{f} to a set of machines M_k by indicating begining times t_{ij} for each one of them, knowing that machines are classified by velocity order.

An example of a position structure $P^{f}(t)$ in a given iteration is illustrated in the following table 1.

Table 1. Example of a position structure $P^f(t)$

Operation	Machine number	Beginning time execution of O_{ij}
O_{11}	M_1	t_{11}
O_{21}	M_3	t_{21}
O_{31}	M_3	t_{31}
O_{12}	M_3	t_{12}
O_{22}	M_2	t_{22}
O_{32}	M_1	t_{32}
O_{13}	M_2	t_{13}
O_{23}	M_1	t_{23}

3.2.2. Proposed algorithm

The Particle Swarm Optimization algorithm steps, applied to flexible job-shop scheduling problems in discrete case, are given in figure1.

First, an Initial population, called swarm, is randomly generated by, affecting each operation to a machine while respecting their precedence order. After that, one particle is selected among the swarm and a neighbourhood containing this particle is chosen.

Then, two heuristics are used to improve machines allocations:

- the first one consists in comparing the machines affectation and their beginning time execution between the chosen particle and its best neighbour and changing the worst affectation ;
- the second one consists in verifying for each operation if the fact of waiting a release of another machine is better or not than to execute this operation on the current one.

To these two local heuristics, a global one is added which consists in changing machines affectation using Δm_{ij}^f and applying it to the current particle to improve optimization function.

For each element of Δm_{ij}^f :

- if this element value is between 0 and 0,5 then, keep the same machine,
- if this element value is between 0,5 and 1 then increment the machine number and if the machine is the last one then, take randomly another machine,
- if this element value is higher than 1 than decrease the machine number and if the machine is the first one then, take randomly another machine,
- in any other case, take the machine that needs the less time for the operation execution.

These steps are repeated until an iterations number fixed previously.

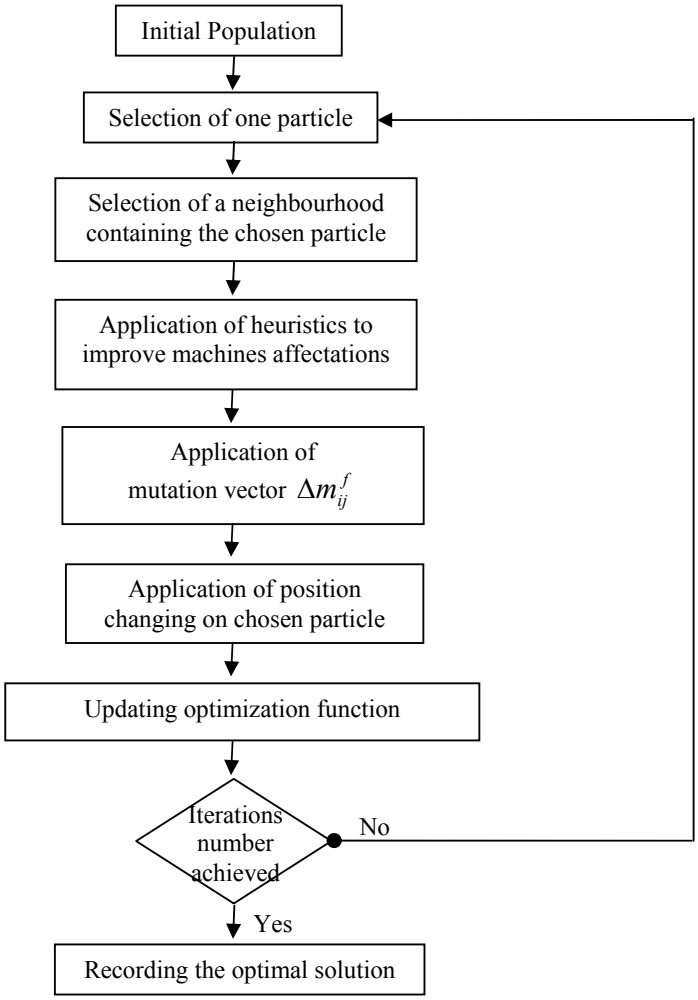

Figure 1. Particle swarm algorithm steps

4. Simulations and Results

In this paper, two flexible job-shop mono-operation problems are treated, the first one deals with a 20 products and 5 machines scheduling where all the machines can be used at a specific time, as shown in table 1. The second one deals with a 10 products and 6 machines scheduling where some of these products can not be executed in specific machines, as shown in table 2 by «--» symbol.

In these two cases, each product contains only one operation and it needs for its execution one of the disposed machines. Another problem, dealing with flexible job-shop multi-operation scheduling is considered. 3 products have to be executed on 5 machines. Each product contains more than one operation, as shown in table 3. Each operation is then, executed on a specific machine and precedence constraints have to be respected.

Table 2. Scheduling data for a 20x5 mono-operation flexible job-shop problem

	M1	M2	M3	M4	M5
J_1	16	79	58	66	54
J_2	89	03	56	58	83
J_3	49	11	20	31	15
J_4	15	99	85	68	71
J_5	89	56	53	78	77
J_6	45	70	35	91	36
J_7	60	99	53	13	53
J_8	23	60	41	59	38
J_9	57	05	69	49	27
J_{10}	64	56	13	85	87
J_{11}	07	03	86	85	76
J_{12}	01	61	72	09	91
J_{13}	63	73	08	39	14
J_{14}	41	75	49	41	29
J_{15}	63	47	47	56	12
J_{16}	47	12	87	40	77
J_{17}	26	21	58	54	32
J_{18}	75	86	18	77	87
J_{19}	77	05	68	51	68
J_{20}	40	77	28	31	94

This table expresses various manufacturing durations of jobs J_j on machines M_k. These durations are different from a machine to another making possible the choice of the most suitable combination to reach optimal solution.

Table 3. Scheduling data for a 10x6 mono-operation flexible job-shop problem

	M1	M2	M3	M4	M5	M6
J1	07	13	06	10	--	--
J2	05	12	08	02	07	11
J3	05	12	06	09	06	17
J4	08	10	--	--	15	--
J5	12	06	08	15	10	09
J6	05	13	07	09	--	--
J7	13	20	08	14	14	17
J8	16	11	05	07	17	09
J9	16	11	08	09	--	--
J10	14	18	06	08	21	14

This table above, also expresses the durations of jobs J_j on machines M_k. But the difference with the previous table is that some jobs can not be executed on some of the machines.

Table 4. Scheduling data for a 3x5 multi-operation flexible job-shop problem

	M1	M2	M3	M4	M5
O11	1.50	3.12	4.91	4.50	9.50
O21	3.00	1.75	4.70	4.50	4.50
O31	4.50	1.75	4.50	3.75	7.00
O12	1.50	4.50	3.25	6.37	4.50
O22	1.50	4.91	4.50	3.75	8.25
O32	4.50	1.75	2.00	4.50	4.50
O13	1.50	4.91	3.25	3.00	4.50
O23	4.50	1.75	4.50	3.75	9.50

This table shows the durations of the operation *i* of the job *j,* expressed by O_{ij}, on the machine M_k. In this example, precedence constraints have to be respected. So, operation O_{22} must be necessarily executed before the operation O_{32}.

For the first problem, a 50 particles population is randomly generated and a neighbourhood containing this particle with 5 others is chosen. The α, β and γ coefficients take respectively 0.2, 0.3 and 0.5 values. Figure 2 shows the algorithm evolution through generations and the stabilization at the 232nd one. In figure 3, Gantt diagram shows the best individual for this 20x5 FJSP problem.

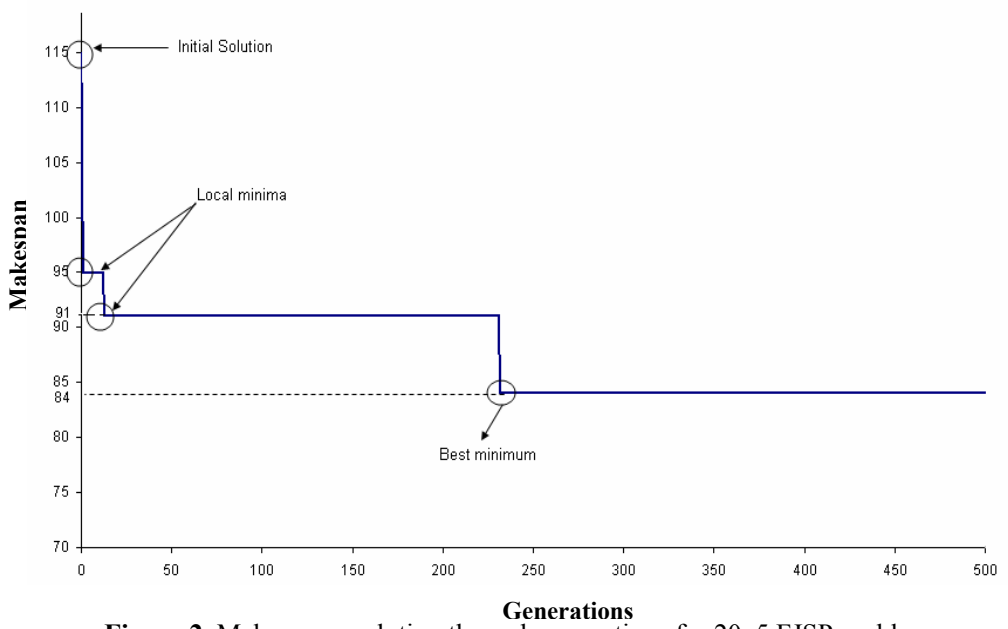

Figure 2. Makespan evolution through generations for 20x5 FJSP problem

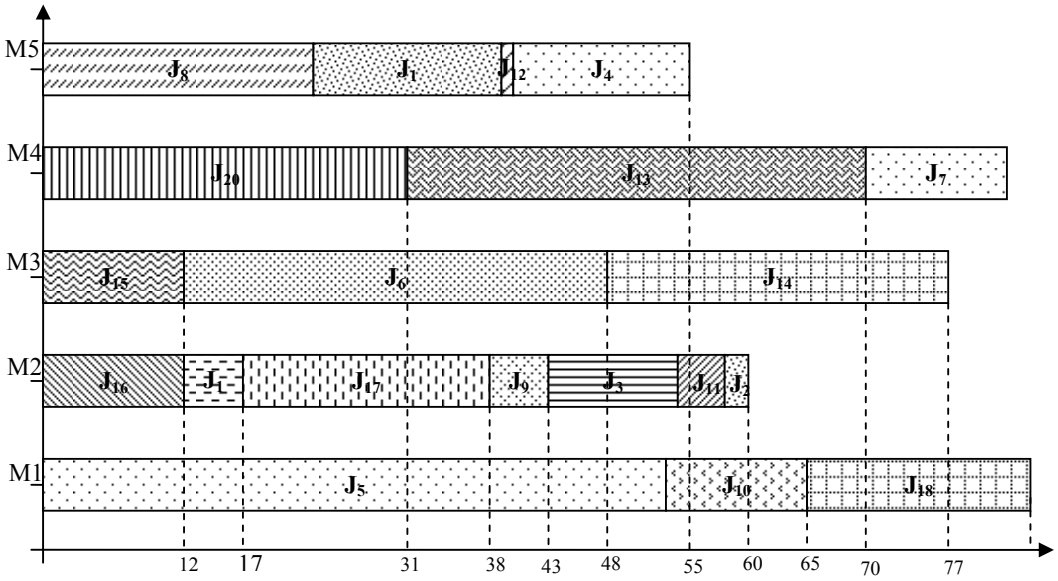

Figure 3. Gantt Diagram of best individual for 20x5 FJSP problem

For the second problem, a 50 particles population is randomly generated and a neighbourhood containing this particle with 5 others is chosen. The α, β and γ coefficients take respectively 0.2, 0.3 and 0.5 values. Figure 4 shows the algorithm evolution through generations and the stabilization at the 63rd one. In figure 5, Gantt diagram shows the best individual for this 10x6 FJSP problem.

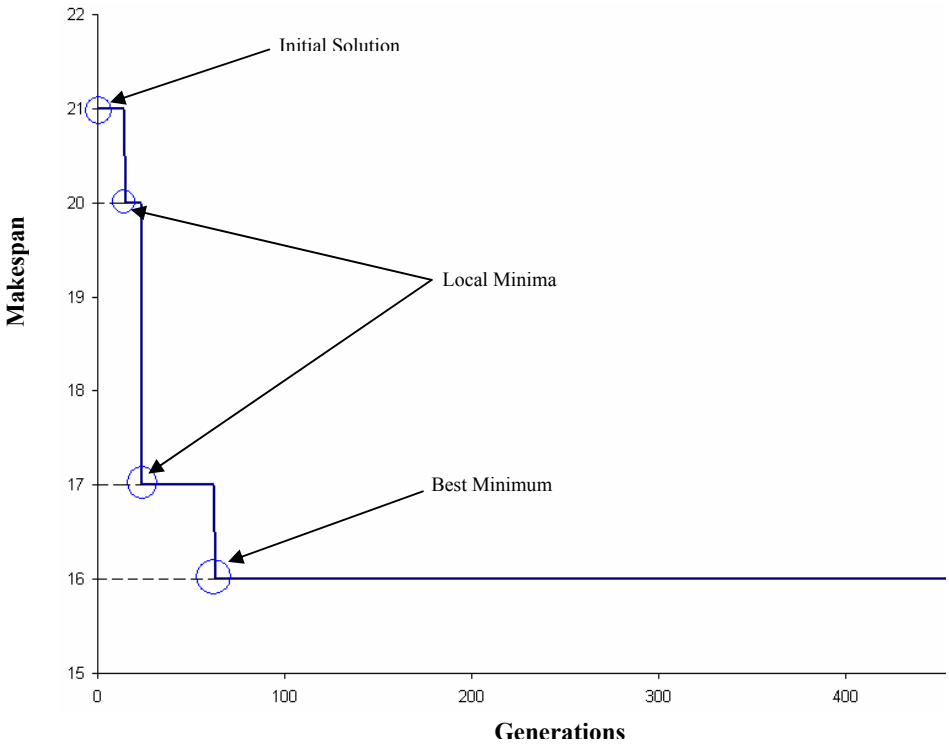

Figure 4. Makespan evolution through generations for 10x6 FJSP problem

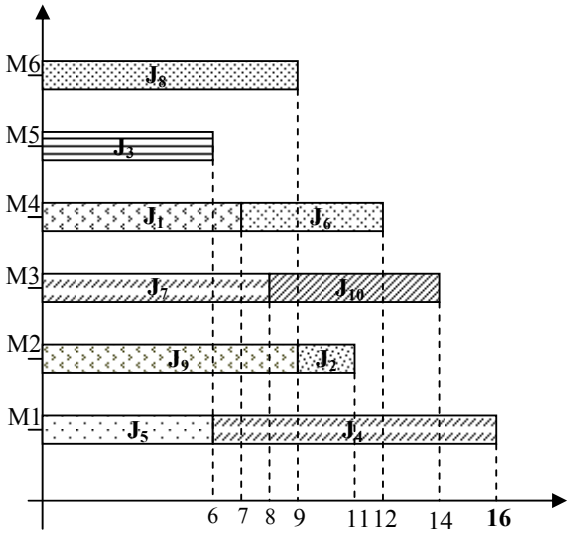

Figure 5. Gantt Diagram of best individual for 10x6 FJSP problem

For the third problem, a 50 particles population is randomly generated and a neighbourhood containing this particle with 5 others is chosen. The α, β and γ coefficients take respectively 0.2, 0.3 and 0.5 values. Figure 6 shows the algorithm evolution through generations and the stabilization at the 12th one. In figure 7, Gantt diagram shows the best individual for this 3x5 FJSP problem.

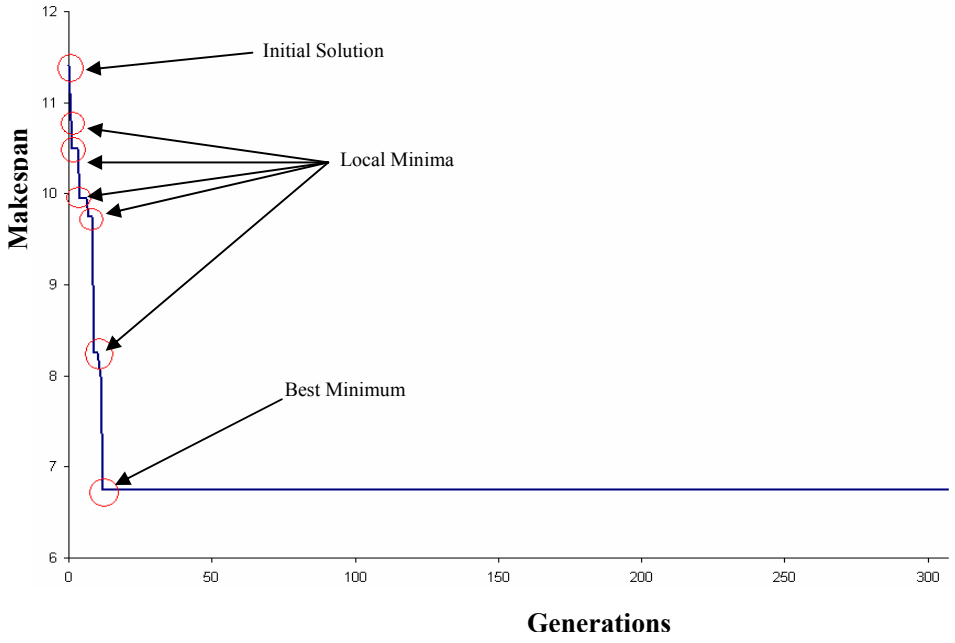

Figure 6. Makespan evolution through generations for 3x5 FJSP problem

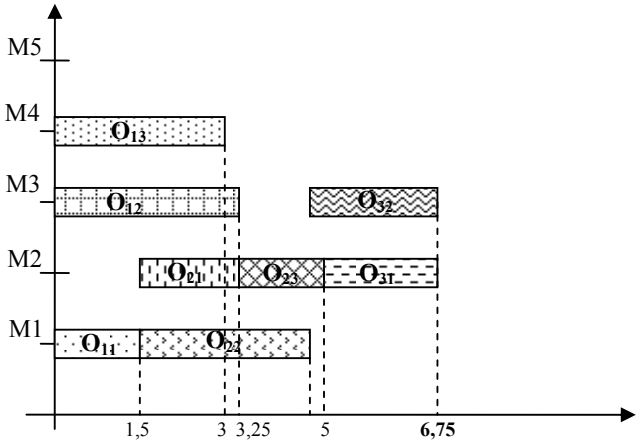

Figure 7. Gantt Diagram of best individual for 3x5 FJSP problem

5. Comparison Between Particle Swarm Optimization and Genetic Algorithm for FJSP Problems

The simulations results in Table 5 below, obtained by particle swarm optimization method are compared with those obtained by genetic algorithm application [Saad, 07] in order to minimize the Makespan.

	20x5 problem		10x6 problem		3x5 problem	
Optimization Method	GA	PSO	GA	PSO	GA	PSO
Makespan	84	84	16	16	6,75	6,75
Convergence	905	232	20	63	14	12

This table shows that for the three problems considered, the particle swarm optimization method and the genetic algorithm method reached the same Makespan, 84 for the first example, 16 for the second one and 6.75 for the third one with best convergences according to the PSO algorithm for first and last examples and a little slower for the second one.

6. Conclusion

The results obtained by applying our algorithm inspired from particle swarm optimization method on flexible job-shop scheduling problems and illustrated with Gantt diagrams, show the effectiveness of this method which leads to a charge balance of operations on selected machines and a minimization of Makespan.

Comparing this method with genetic algorithms one allows us to validate the use of the PSO in the discrete case.

REFERENCES

1. BOUKEF, H., TANGOUR F. AND BENREJEB M., **Sur la formulation d'un problème d'ordonnancement de type flow-shop d'ateliers de production en industries pharmaceutiques**, Journées Tunisiennes d'Electrotechnique et d'Automatique, JTEA'06, Hammamet, 2006.
2. BOUKEF, H., BENREJEB M. AND BORNE P., **A proposed genetic algorithm coding for flow-shop scheduling problems**, International Journal of Computers, Communications and Control, IJCCC, vol. 2, no. 3, 2007, pp. 229-240.
3. CLERC, M., **The swarm and the queen: towards a deterministic and adaptive Particle swarm optimization**, IEEE Congress on Evolutionary Computation, vol. 3, 1999, pp. 1951-1957.
4. COLORNI, A., DORIGO M., MANIEZZO V. AND TRUBIAN M., **Distributed optimization by ant colonies** , First European Conference on Artificial Life, Paris, 1991, pp. 134-142.
5. DEROUSSI, L., GOURGAND M., KEMMOE S. AND QUILLIOT A., **Discrete particle swarm optimization for the permutation flow-shop problem**, 2006.
6. FILIP, F.G., NEAGU G. AND DONCIULESCU D.A., **Job-Shop scheduling optimization in real- time production control,** Elsevier Science Publishers, North Holland, 1983, pp. 395-403.
7. GLOVER, F., **Tabu search, part II**, ORSA, Journal of Computing, vol. 2, 1989, pp. 24-32.
8. HO N. B., TAY J. C. and LAI E. M., **An effective architecture for learning and evolving flexible job-shop schedules,** European Journal of Operational Research, vol. 179, 2006, pp. 316–333.
9. HOLLAND, J.H., **Adaptation in natural and artificial systems** , University of Michigan Press, Michigan, 1975.
10. KENNEDY, J. and EBERHART R.C., **Particle swarm optimization**, IEEE International Conference on Neural Networks, Piscataway, 1995, pp. 1942-1948.
11. KIRKPATRICK, S., and VECCHI M.P., **Optimization by simulated annealing**, Science, vol. 220, 1983, pp. 671-680.
12. LIAN, Z., JIAO B. and GU X., **A similar particle swarm optimization algorithm for job-shop scheduling to minimize Makespan**, Applied Mathematics and Computation, vol. 183, 2006, pp. 1008–1017.
13. LIAN, Z., GU X. and JIAO B., **A similar particle swarm optimization algorithm for permutation flow-shop scheduling to minimize makespan**, Applied Mathematics and Computation, no. 175, 2006, pp. 773–785.

14. LIAN, Z., GU X. and JIAO B., **A novel particle swarm optimization algorithm for permutation flow-shop scheduling to minimize makespan** , Chaos, Solitons and Fractals, 2006, pp. 1-11.

15. LIAO, C. J., TSENG C. T. and LUARN P., **A discrete version of particle swarm optimization for flow-shop scheduling problems,** Computers & Operations Research, vol. 34, 2007, pp. 3099–3111.

16. LIOUANE, N., SAAD I. and BORNE P., **Ant system and fuzzy controller for multi-objective optimization of the flexible job-shop scheduling problems**, Studies in Informatics and Control, vol. 16, no. 2, 2007, pp. 217-226.

17. MESGHOUNI, K., HAMMADI S. and BORNE P., **Production job-shop scheduling using genetic algorithms**, Systems, Man and Cybernetics, IEEE International Conference, 1996.

18. MESGHOUNI, K., HAMMADI S. and BORNE P., **On modelling genetic algorithm for flexible job-shop scheduling problems**, Studies in Informatics and Control, vol. 7, no. 1, 1998, pp. 37-47.

19. SAAD, I., BOUKEF H. ET BORNE P., **The comparative of criteria aggregative approach for the multi-objective optimization flexible job-shop scheduling problems**, Fourth Conference on Management and Control of Production and Logistics, MCPL, Sibiu, 2007.

20. SAAD, I., HAMMADI S., BENREJEB M.and BORNE P., **Choquet integral for criteria aggregation in the flexible job-shop scheduling problems**, IMACS Intenational Journal, Mathematics and Computers in Simulation, MATCOM, Elsiver, vol. 76, 2008, pp. 447-462.

21. SHA, D. Y. and HSU C. Y., **A hybrid particle swarm optimization for job shop scheduling problem**, Computers and Industrial Engineering, vol. 51, 2006, pp. 791–808.

22. SHELOKAR, P.S., SIARRY P., JAYARAMAN V. K. and KULKARNI B. D., **A Particle swarm and ant colony algorithms hybridized for improved continuous optimization**, Applied Mathematics and Computation, vol. 188, 2007, pp.129-142.

23. SHI, X.H., LIANGA Y.C., LEEB H.P., LUB C. and WANG Q.X., **Particle swarm optimization-based algorithms for TSP and generalized TSP**, Information Processing Letters, vol. 103, 2007, pp. 169-176.

24. TANGOUR, F. and SAAD I., **Multi-objective optimization scheduling problems by pareto-optimality in agro-alimentary workshop**, International Journal of Computers Communication & Control, IJCCC, vol. 1, no. 3, 2006, pp. 71-83.

25. TANGOUR, F. and BORNE P., **Ordonnancement des opérations dans un atelier de production agroalimentaire en minimisant le coût**, Revue e-Sta, vol. 3, no. 1, 2006.

26. TASGETIREN, M., SEVKLI M., LIANG Y. C. and GENCYILMAZ G., **Particle Swarm Optimization Algorithm for Single Machine Total Weighted Tardiness Problem**, IEEE, Digital Object Identifier, vol. 2, 2004, pp. 1412-1419.

27. VAN DEN BERGH, F. and ENGELBRECHT A.P., **Cooperative learning in neural network using particle swarm optimizers**, South African Computer Journal, vol. 26, 2000, pp. 84-90.

28. XIA, W. and WU Z., **An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems**, Computers and Industrial Engineering, vol. 48, 2005, pp. 409-425.