A multi-agent model based on Tabu Search for the permutation flow shop problem minimizing total flowtime

Soumaya Ben Arfa* — Olfa Belkahla Driss**

* Stratégies d'Optimisation et Informatique intelligentE (SOIE) High Institute of Management University of Tunis 41, Street of Liberty Bouchoucha-City CP-2000-Bardo Tunis, Higher Business School of Tunis, University of Manouba
Tunisia
benarfa.soumaya@gmail.com

** Stratégies d'Optimisation et Informatique intelligentE (SOIE) High Institute of Management University of Tunis 41, Street of Liberty Bouchoucha-City CP-2000-Bardo Tunis, Higher Business School of Tunis, University of Manouba
Tunisia
olfa.belkahla@isg.rnu.tn

ABSTRACT. In this paper, we treat the permutation flowshop scheduling problem with total flowtime minimization. We have propose a multi-agent model using tabu search method for solving this type of problem. Our proposed model MA.TF.PFS « Multi-Agent model to minimize Total Flowtime in Permutation Flow Shop » is composed by two classes of agents which are the supervisor agent and n job agents. The supervisor agent generates an initial solution and each job agent has a key role, it is a scheduler looking for a neighbor solution to improve the current solution by tabu search metaheuristic. Computational results show that the MA.TF.PFS is performant and it is significantly better than the BES (LR) method and three of other metaheuristics.

RÉSUMÉ. Dans cet article, nous traitons le problème d'ordonnancement d'atelier de type flow shop de permutation avec la minimisation de temps d'écoulement total. Nous proposons un modèle Multi-Agents en utilisant la méthode de recherche tabou pour résoudre ce type de problème. Notre modèle proposé MA.TF.PFS « Multi-Agent model to minimize Total Flowtime in Permutation Flow Shop » est composé de deux classes d'agents : Un agent superviseur et n agents jobs. L'agent superviseur génère une solution initiale et l'agent job a un rôle primordial, c'est un ordonnanceur qui cherche une solution voisine pour améliorer la solution courante en utilisant la métaheuristique recherche tabou. Les résultats obtenus montrent que MA.TF.PFS est performant et il est nettement meilleur que la méthode BES (LR) et trois autres métaheuristiques.

KEYWORDS : Multi-agent systems, Scheduling, Permutation flow shop, Total flowtime, Tabu search.

MOTS-CLÉS : Système multi-agents, Ordonnancement, Flow shop de permutation, Temps d'écoulement total, Recherche tabou

1. Introduction

The Permutation Flow shop Scheduling Problem (PFSP) is an important manufacturing system widely existing in industrial environments. So it can be described as follows: n different jobs are processed on m machines, where jobs on each machine follows the same order. The makespan or the minimization of total completion time, is considered to be the traditional criterion. Nowadays, the minimization of total flowtime has become an interesting topic in the scheduling literature. The PFSP with total flowtime criterion has proved to be NP-complete [6], even with two machines. However, so far no method seems to be the best for total flowtime minimization, including mathematical methods [1] [7], many heuristics and metaheuristics have been proposed. Most researches [5] [11][9][10] have been devoted to developing heuristic algorithms to obtain good solutions. Liu and Reeves [8] proposed an effective method LR(x) to generate the initial solution for their composite heuristics, by which new best solutions were found for nearly all 120 benchmark instances [13]. At the same time, many heuristics have been [5] integrated NEH insertion method as well as the pairwise exchange strategy in their algorithm. Indeed, we are looking for faster solutions leading to the development of several metaheuristics. Rajendran and Ziegler proposed two ant colony algorithms called M-MMAS and PACO [13]. The Particle Swarm Optimization algorithm called PSOvns where a SPV (the smallest position value) rule and VNS (variable neighborhood search) local search were applied proposed by [14]. Some of the most recent are the artificial bee colony algorithm and a discrete differential evolution algorithm illustrated by [15]. Dong et al. [3] proposed a Multi-Restart Iterated Local Search algorithm called MRSILS. Nowadays, they showed that the works are clearly superior to the heuristics addressed by Liu and Reeves unless for 100 benchmark instances by Taillard. All works that have been done for solving this type of problem are centralized, but for the minimization of Makespan, [2] used multi-agent systems proving best results. Based on these results, we suggest a model based on multiagent paradigm. The remainder of this paper is structured as follows: in section 2, we briefly describe the formulation of PFSP with total flowtime minimisation. We describe in details the Multi-Agent model based on Tabu Search in section 3. Section 4 contains the adaptation of the different elements of the Tabu Search. In section 5, we propose the dynamic of MA.TF.PFS. In section 6, experimental results are proposed. Finally, section 7 concludes the paper and suggests some future studies.

2. Problem Formulation

The PFSP with minimizing total flow time can be formally defined as follows: A set of jobs N= 1, 2 . . ., n available at time zero must be processed on m machines, where n $\geq$ 1 and m$\geq$1. The processing time for job i on the machine j is noted by $p_{i,j}$. C_i denotes the completion time of job i, where the completion time for job i on the machine j noted by $C(i,j)$ whether π (π_1,π_2,...π_n) a permutation, which represents the completion time of job π_i on the machines j. It Can be calculated as follows:

$C(\pi_1, 1) = p_{\pi_1,1}$,

$C(\pi_i, 1) = C(\pi_1, 1) + p_{\pi_1,1}$ for i=2, ..., n,

$C(\pi_1, j) = C(\pi_1, j-1) + p_{\pi_1,j}$ for j=2, ..., n,

$C(\pi_i, j) = \max C(\pi_{i-1}, j), C(\pi_1, j-1) + p_{\pi_1,1}$,

For $i = 2, ..., n; j = 2, ...m$.

Since ready times are zero, the flow time $C(\pi_i)$ is equivalent to the completion time $C(\pi_i, m)$. As a result, The PFSP with the total flow time is to find a π^* permutation throughout all Π permutations so that:

$$\sum_{i=1}^{n} C(\pi_i^*, m) \leq \sum_{i=1}^{n} C(\pi_i, m), \qquad \forall \pi \in \Pi \qquad [1]$$

3. The Multi-Agent model based on Tabu Search for PFSP

The different solving approaches that exist in the literature are all centralized architectures. They are sometimes ineffective given the difficulty of the problem. That's why we turned to the solving distribution by the use of Multi-Agent Systems [4]. So this type of system offers some parallel architectures that save computation time when solving difficult or large problems. We present in this section our multi-agent model named Multi-Agent model to minimize Total Flow in Permutation Flow Shop (MA.TF.PFS) as illustrated in Figure 1. The model consists of two types of agents: one Supervisor agent and n Job Agents where n is the number of jobs. Each agent in our model has its own static and dynamic knowledge and its own behavior. This behavior depends on its state, it can be: satisfied, unsatisfied or gives priority to the processing of messages. In addition, each agent has some acquaintances, the agents knows with which to communicate. In the remainder of this section, we show the different types of agents.

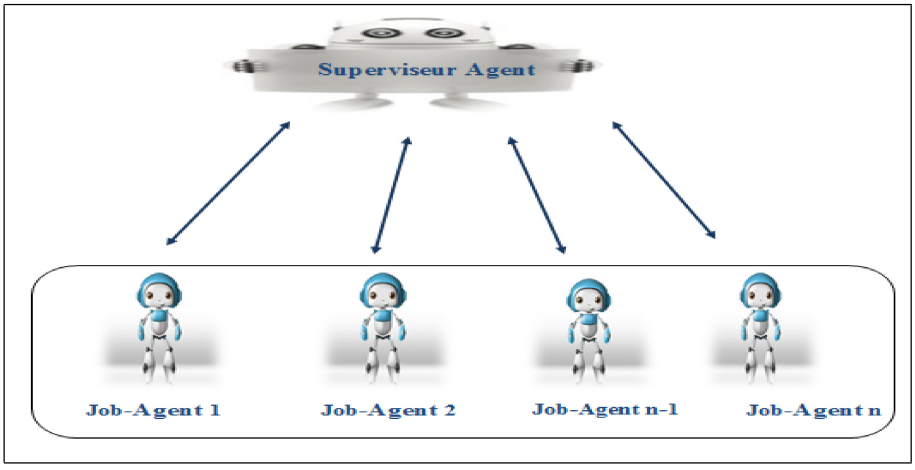

Figure 1. *MA.TF.PFS model*

3.1. The Supervisor Agent

The Supervisor Agent contains the core of the tabu search algorithm. It aims to launch the program, generating an initial solution and create as many agents as jobs named Job-Agent. In our approach, the Supervisor Agent is a regulator, it can communicate with all the job agents with the overall goal of finding an optimal schedule minimizing the total

flow time. As the number of iterations has reached the maximum number, the Supervisor Agent is not satisfied. Otherwise, it provides the best solution to the user. The Supervisor Agent has as acquaintances all agents in the system. Its static knowledge includes:

– The system operations to be performed and their respective durations on different machines;

– The size of the tabu list ;

– The maximal number of iterations allowed;

– The initial solution S0 from which the optimization process begin.

Its dynamic knowledge consists of:

– The tabu list elements;

– The current solution and the total flow time ;

– The neighbours and their solution ;

– The number of iterations ;

– The best solution found by the tabu search until the current iteration and its total flowtime.

3.2. The Job Agent

In our model, we have n Job Agents (n is the number of jobs of the flow shop problem). Each Job-Agent has an important role, it is a scheduler that is looking for a neighbor solution in order to improve the current solution. Each agent has its own dynamic and static knowledge and his own behavior. This behavior depends on its state: satisfied or unsatisfied and gives priority to the processing of messages. Furthermore, each agent has acquaintances. Its static knowledge includes:

– The diversification requirement ;

– The execution times of each job on all machines;

Its dynamic knowledge consists of:

– The common solution sent by the Supervisor Agent;

– The tabu list elements;

– The best solution met for each Job-Agent.

4. Global dynamic

In our model MA.TF.PFS, the global optimization process is carried out by collaboration between the Supervisor-Agent and the Job-Agents. The Supervisor-Agent knows the problem to solve. So, it generates an initial solution and then tries to improve it with applying the Tabu search method. Once the initial solution is determined, it is considered as a common solution. The Supervisor-Agent sends a message to the n Job-Agents each of which contains the current solution and the total flow time. In a parallel manner, the Job-Agents look for other neighbor solutions using a smart search in the diversification phase to get rid the local optimum. After the total flow time calculations of all neighbor solutions, they will be sent to the Supervisor-Agent. It chooses the best non-tabu neighbor with a minimum Total Flow Time to start a new iteration and inserts its total flow time in the tabu list. The above process continues until the stopping rule is satisfied. At this point, the Supervisor-Agent kills all the Job-Agents and displays to the user the best scheduling

found and its total flow time. We can see in Algorithm 1 the used tabu search algorithm.

Algorithm 1 The used tabu search algorithm

List-tabu$\leftarrow \emptyset$
Nbr-iter $\leftarrow 0$
Current-schedule$\leftarrow$ Receive-initial-schedule(initial-schedule, supervisor-agent)
Best-schedule $\leftarrow$ Current-schedule
Best-TFT $\leftarrow$ Courrent-Total-Flowtime
while (Nbr-iteration$\leq$ Nbr-iter-max) **do** */*the Nbr-iter-max can vary between 10-100*/*
    Diversification
    List-tabu$\leftarrow$ add-in-List-Tabu(Best-TFT) */*the list tabu = 50 */*
    Nbr-iter ++
end while

Despite the effectiveness of tabu search method in solving permutation flow shop scheduling problems, certain limitations have been detected. In fact, the main inconvenience is summed up in the absence of an effective diversification technique that encourages the search process to examine unvisited regions, as the best solutions at the local level are not necessarily good solutions globally. In our model, the Job-Agent is responsible for the diversification phase. Indeed, we implement a research method at Job-Agent level to get better neighbor solutions. Hence the research method is proposed in Algo 2. At each iteration diversification called iter-div, the Job-Agent moves its job to another position in the current solution and choose the best among them. Once the variable nbr-iter-div exceeds a predetermined number of iterations, called Threshold-Div, then the Job-agent sends the best solution 'Best-Sol' found and the Best Total Flow Time 'Best-TFT' to Supervisor-Agent.

Algorithm 2 The research method of Diversification

Nbr-iter-div = 0
List-tabu = $\emptyset$
while (Nbr-iter-div$\leq$ Threshold-Div) and (current-TFT $\leq$ Best-TFT) **do**
    Current-position$\leftarrow$ Insert-moves (another-position);
    List-tabu $\leftarrow$ Add(current-position);
    Nbr-iter-div ++
end while
Send (Best-Sol, Best-TFT, Supervisor-Agent)

5. Experimental results

In this section, MA.TF.PFS is compared with the best method (BES (LR) refers to the best performing heuristic as investigated by [8]); the two ant colony algorithms (M-MMAS and PACO) by [12]; the Particle Swarm Optimization algorithm with local search (PSOvns) by [15]; the a Discrete Artificial Bee Colony algorithm (DABC) by [14]; and the

Multi-Restart Iterated Local Search algorithm (MRSILS) by [3]. The proposed approach is implemented in the JADE platform and tested on a core i3 2.5 Mhz with 4GB RAM and we use the Taillard's instances [13]. We solve 110 problems in which the number of jobs between 20 and 200 and the number of resources varies between 5 and 20. Experimental results are presented in Table 1 by calculating the Relative Percentage Deviation (RPD) of the obtained results. RPD is calculated by the following equation:

$$RPD = \frac{Obt_{sol} - Best_{sol}}{Best_{sol}} * 100 \quad [2]$$

Hence Obt_{sol} is the solution yielded by a combination of factors for a given instance and the $Best_{sol}$ given by all combinations of factors for an instance. From table1, it can be concluded that the average performance of MA.TF.PFS is better than BES (LR), M-MMAS, PACO, and PSOvns. With respect to the rest of the methods, BES (LR) is outperformed by other algorithms. However, BES(LR) is rather simple and easily implemented compared to other algorithms. Therefore regarding the average performance, it seems that our model is effective in solving flow shop problems with the total flowtime criterion compared with the existing algorithms. According to the results presented in Table 1, we notice that MMAS, PACO, PSOvns, DABC and MRSILS approaches have not solved the problems of large size such as n=200. On the other hand, our approach is effective if the problem size increases. In table 2, we present only instances to that the best solution (bold values) are given by MA.TF.PFS. We also remark that the results obtained by MA.TF.PFS are better performing on 37.4 % of instances. So with n=200, we can see that the proposed model MA.TF.PFS provided the optimal solution for 11 instances out of 20.

Table 1. *Average relative percentage deviation over the best solutions*

instances	BES(LR)	M-MMAS	PACO	PSOvns	DABC	MRSILS	MA.TF.PFS
20x5	1.361	0.197	0.454	0.000	0.006	0.006	0.088
20x10	1.433	0.049	0.323	0.002	0.000	0.000	0.329
20x20	1.019	0.118	0.732	2.260	0.000	0.000	0.284
50x5	1.835	1.413	1.227	0.526	0.162	0.031	1.025
50x10	2.906	1.908	1.644	0.666	0.050	0.083	1.572
50x20	2.709	1.600	1.289	2.155	0.019	0.158	0.863
100x5	1.067	0.918	1.136	0.310	0.198	0.005	0.369
100x10	2.156	1.746	1.402	0.689	0.245	0.005	1.742
100x20	3.263	1.991	1.733	1.612	0.156	0.046	1.725
average	1.972	1.104	1.104	0.913	0.142	0.029	0.887

Table 2. *Best solutions obtained by MA.TF.PFS on Taillard's benchmarks*

Problem	N/M	BES(LR)	M-MMAS	PACO	PSOvns	DABC	MRSILS	MA.TF.PFS
Ta002		15446	15151	15214	15151	15151	15151	**15151**
Ta003		13676	13416	13403	13301	13301	13301	**13301**
Ta004		15750	15486	15505	15447	15447	15447	**15447**
Ta005	20x5	13633	13529	13529	13529	13529	13529	**13529**
Ta008		13968	13968	14042	13948	13948	13948	**13948**
Ta009		14456	14317	14383	14295	14295	14295	**14295**
Ta010		13036	12968	13021	12943	12943	12943	**12943**
Ta011		21207	20980	20958	20911	20911	20911	**20911**
Ta013	20x10	20072	19833	19968	19833	19833	19833	**19833**
Ta017		18723	18376	18377	18363	18363	18363	**18363**
Ta019		20561	20330	20330	20330	20330	20330	**20330**
Ta022		31918	31604	31597	32659	31587	31587	**31587**
Ta027	20x20	33449	33038	32922	33733	32922	32922	**32922**
Ta028		32611	32444	32533	33008	32412	32412	**32412**
Ta029		33625	33623	34446	33600	33600	33600	**33600**
Ta033	50x5	64378	64166	64149	63577	63162	63241	**63162**
Ta038		65582	64863	65123	64638	64381	64578	**64381**
Ta056	50x20	124061	122369	122262	123217	120850	121083	**120850**
Ta057		126363	125609	125351	125586	123043	123084	**123043**
Ta059		125318	126582	123646	124932	121872	122111	**121872**
Ta066		235793	236225	236409	234082	234017	233651	**233651**
Ta068	100x5	235171	234813	234579	232755	232238	232167	**232167**
Ta069		251291	252384	253325	249959	249884	248999	**248999**
Ta079	100x10	312175	309664	305376	305605	304457	304026	**304026**
Ta091		1063976	-	-	-	-	-	**1062859**
Ta092		1049076	-	-	-	-	-	**1040604**
Ta094	200x10	1051335	-	-	-	-	-	**1048682**
Ta095		1055823	-	-	-	-	-	**1052832**
Ta097		1071471	-	-	-	-	-	**1052832**
Ta099		1045183	-	-	-	-	-	**1043902**
Ta100		1044888	-	-	-	-	-	**1038016**
Ta103		1297768	-	-	-	-	-	**1254529**
Ta105	200x20	1255708	-	-	-	-	-	**1236246**
Ta109		1259311	-	-	-	-	-	**1237428**
Ta110		1273354	-	-	-	-	-	**1253075**

6. Conclusion and future works

In this paper, we have proposed a multi-agent approach by using tabu search method to solve the permutation Flow Shop scheduling problem with minimizing total flow time. The model MA.TF.PFS (Multi-Agent model to minimize Total Flowtime in Permutation Flow Shop) provides good results and allows to solve large size problems. It is competitive with other successful methods. In the future works, we are planning to make some modifications in order to enhance the performance of our model. We can reinvest our work to study Flow shop using other optimization criteria. Another interesting work field would be to adapt our model for multi-objective scheduling problems.

7. References

[1] BANSAL.S. P, "Minimizing the sum of completion times of n jobs over m machines in a flowshop - A branch and bound approach ", *AIIE Transactions* , vol. 9, num. 306-311, 1977.

[2] BELKAHLA.DRISS.O, BARGAOUI.H, " Multi-Agent Model based on Tabu Search for the Permutation Flow Shop Scheduling Problem", *Advances in Distributed Computing And Artificial Intelligence Journal*, 2014.

[3] DONG. X, CHEN .P, HUANG.H , NOWAK.M, " A multi-restart iterated local search algorithm for the permutation flow shop problem minimizing total flow time", *Computers Operations Research*, 2013.

[4] FERBER.J, "Les systèmes multi-agents vers une intelligence collective", *InterEditions*, 1995.

[5] FRAMINAN.JM, LEISTEN.R, "An efficient constructive heuristic for flowtime minimization in permutation flow shops", *Omega*, 2003.

[6] GAREY.MR, JOHNSON.DS, SETHI.R "The complexity of flow shop and job shop scheduling", *Mathematics of Operations Research*, vol. 1 num. 117-29, 1976.

[7] IGNALL.E, SCHRAGE.L, "Application of the branch and bound technique to some flowshop scheduling problem", *Operations Research*, vol. 13 num. 400-412, 1965.

[8] LIU.JY, REEVES.CR, "Constructive and composite heuristic solutions to the $P//\sum Ci$ scheduling problem", *European Journal of Operational Research*, 2001.

[9] LI.XP, WU.C, "An efficient constructive heuristic for permutation flow shops to minimize total flow time ", *Chinese Journal of Electronics*, 2005.

[10] LAHA.D, SARIN. SC, "A heuristic to minimize total flowtime in permutation flowshop", *Omega*,2009.

[11] RAJENDRAN.C, ZIEGLER.H, "An efficient heuristic for scheduling in a flow shop to minimize total weighted flow time of jobs", *European Journal of Operational Research*, vol. 38 num. 103-129 ,1997.

[12] RAJENDRAN.C, ZIEGLER.C, "Ant-colony algorithms for permutation flowshop scheduling to minimize makespan/total flowtime of jobs", *European Journal of Operational Research*, 2004.

[13] TAILLARD.E, "Benchmarks for basic scheduling problems", *European Journal of Operational Research*, vol. 64 num. 78−285 , 1993.

[14] TASGETIREN.M, PAN.Q.K,, SUGANTHAN.P,, CHEN.AH.L "A discrete artificial bee colony algorithm for the total flowtime minimization in permutation flow shops", *Information Sciences*, 2011.

[15] TASGETIREN.M.F, LIANG.Y.C, SEVKLI.M, GENCYILMAZ.G "A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem", *European Journal of Operational Research*, vol. 177 num. 1930-1947,2007.