Diversifying TS using GA in Multi-agent System for Solving Flexible Job Shop Problem

Ameni Azzouz, Meriem Ennigrou and Boutheina Jlifi
L. SOIE. Stratégie d'Optimisation et Informatique IntelligentE , ISG, Institut Supérieur de Gestion, Université de Tunis, Tunis, Tunisie

Keywords: Diversification, Tabu Search, Genetic Algorithm, Multi-agent System, Flexible Job Shop.

Abstract: No doubt, the flexible job shop problem (FJSP) has an important significance in both fields of production management and combinatorial optimization. For this reason, FJSP continues to attract the interests of researchers both in academia and industry. In this paper, we propose a new multi-agent model for FJSP. Our model is based on cooperation between genetic algorithm (GA) and tabu search (TS). We used GA operators as a diversification technique in order to enhance the searching ability of TS. The computational results confirm that our model MAS-GATS provides better solutions than other models.

1 INTRODUCTION

The Flexible Job Shop Problem is a generalization of the classical Job Shop Scheduling Problem, where each operation can be processed by more than one resource. Considering the flexibility constraints, FJSP presents additional difficulty than the classical JSP and requires more effective algorithms. In recent decades, many attempts have been made to find the near optimal solution of FJSP using a varied range of tools and techniques such as Branch and Bound (Fattahi et al., 2007; Zribi et al., 2007) at one hand and Heuristics (Wang and Yu, 2010; Ziaee, 2014) at the other hand. FJSP is known to be strongly NP-hard (Garey et al., 1976). Consequently, most of the literature related to the FJSP is based on meta-heuristic methods like genetic algorithms (GAs) (Pezzella et al., 2008; Zhang et al., 2011; Zhou et al., 2013; Zambrano rey et al., 2014), particle swarm optimization (PSO)(Zhang et al., 2009; Nouiri et al., 2015), simulated annealing (SA) (Najid et al., 2002; Yazdani et al., 2009), tabu search (TS) (Vilcot and Billaut, 2011; Brandimarte, 1993; Saidi-Mehrabad and Fattahi, 2006), beam search (BS) (Wang et al., 2008).

Nevertheless, most of the above-mentioned research considered only one method optimization to solve FJSP. However, the literature reviews show that none of these methods are sufficient on their own to solve this NP-hard problem. Hence, almost of research considered hybridization between several methods such as (Xia and Wu, 2005; Zhang et al., 2008; Li et al., 2010a; Henchiri and Ennigrou, 2013). The motivation behind such hybrid methods is usually to obtain a near optimal solution. In (Shao et al., 2013), the authors propose a hybrid algorithm based on discrete PSO used for global search and SA significant for local search. Differently, (Li et al., 2010b) use the PSO to produce a swarm of high quality candidate solutions, and TS algorithm to find the near optimal solution around the given particle.

Generally, there are two main categories for hybridization (Puchinger and Raidl, 2005): The first one is the integrative combination in which one method is used as a step in the second method such as (Azzouz et al., 2012) which combine the GA and the TS for solving FJSP. The second category is the collaborative combination in which no algorithm is contained in the other.Here, the algorithms are executed sequentially, or in an intertwined or even parallel way. Our proposed model, MAS-GATS, appears in this last class in which tabu search and genetic algorithm collaborate by exchanging its solutions (see section 3).

Furthermore, meta-heuristic approaches can be easily trapped in local minima hence they need diversification strategies to generate solutions that differ from each other in significant ways in order to examine unvisited regions of the search solution space, and that yield interesting alternatives in the context of the problem considered (Glover, 1998). These strategies are especially significant to the search process that, starting from a particular point, explore a solu-

DOI: 10.5220/0005511000940101
In *Proceedings of the 12th International Conference on Informatics in Control, Automation and Robotics* (ICINCO-2015), pages 94-101
ISBN: 978-989-758-122-9

tion path until new exploitable regions are inaccessible, and a new starting point becomes necessary(Kelly et al., 1994).

Several diversification strategies are used in literature. For example, tabu search use memory list containing the solutions or movements used frequently. These latter will be penalized in order to not be visited during the diversification steps. Diversification in GRASP (Greedy Randomized Adaptive Search Procedures) is achieved by means of controlled random sampling. Simulated annealing incorporates randomization to make diversification a function of temperature, whose gradual reduction correspondingly diminishes the directional variation in the objective function trajectory of solutions generated (Glover and Laguna, 1999). Genetic Algorithms use randomization in component processes such as combining population elements and applying selection, crossover and mutation operators, thus providing some diversifying power.

In this paper, we are interested for diversification techniques used in tabu search. Ennigrou M. and Ghdira K. (Ennigrou and Ghedira, 2008) show the effectiveness of three diversification techniques used in order to enhance the searching ability of TS via multi-agent system for solving flexible job shop problem: the first consists in starting from a solution selected randomly among its elite solution list (i.e. list containing the best solutions) and non belonging to its diversification tabu list (i.e. list containing the last solutions used during the last diversification phases). The second technique starts from a new solution created by re-sequencing the operations of one job selected randomly; the re-sequencing consists for each operation of the job selected in choosing a potential resource and fixing a start time equal to the finish time of its predecessor. The third technique is to start again TS for the local minima finding after a certain number of iterations. In (Laguna and Glover, 1993) frequency counts are used to bias the selection of moves in TS solution states where no improving moves are available. Applied to single machine scheduling, the frequency count is multiplied by a penalty parameter and added to the move value of every non-improving move. Then, the move with the least penalized move value is selected.

In the recent decades, few approaches have used GA operators in order to provide an approximate diversifying effect on local search algorithms. For example, the basic algorithm of (Sohn et al., 2005) referenced by RasID-GA (Adaptive Random Search with Intensification and Diversification combined with Genetic Algorithm) used Automatic switching: first, it executes a number of RasIDs in parallel; then the transition from RasID to GA aiming to escape local minima provided by RasID and finally the return to RasID again.

In this paper, we propose, a new approach MAS-GATS based on multi-agent system using two agent classes: the first named Resource agents responsible of local optimization process. The second named Supervisor agent responsible of global optimization process. This paper is organized as follows. Section 2 defines the flexible Job Shop. Section 3 describes our proposed approach. Section 4 presents an empirical evaluation of MAS-GATS. Finally, in section 5, we conclude by discussing the considerable promise of using our approach.

2 FLEXIBLE JOB SHOP PROBLEM

The FJSP consists in performing$n$ jobs on m machines. The set machines is noted M, $M = \{M_1, ..., M_m\}$. Each job i consists of a sequence of n_i operations (routing). Each routing has to be performed to complete a job. The execution of each operation j of a job i (noted O_{ij}) requires one machine out of a set of given machines M. The problem is to define a sequence of operations together with assignment of start times and machines for each operation.

Assumptions considered in this paper are the following:

- jobs are independent of each other;
- machines are independent of each other;
- one machine can process at most one operation at a time;
- no preemption is allowed.
- setup and transportation times are negligible
- all jobs are available at time zero.

The current FJSP based on these assumptions is aimed to minimize the makespan, i.e., the maximal completion time of machines or jobs. FJSP is classified as Total FJSP and Partial FJSP (Kacem et al., 2002). In Total FJSP (T-FJSP), each operation can be processes by all machines. However, in Partial FJSP (P-FJSP), at least one operation may not be processed on all machines. Several researches pointed out that the P-FJSP is more complex as compared to T-FJSP on the same scale. In this paper, we assume the P-FJSP. Table 1 illustrates an example of P-FJSP with four jobs and four machines. In the table, the symbol - means that the machine can't execute the corresponding operation, i.e., it does not belong to the subset of compatible machines for that operation.

Table 1: An instance of FJSP.

Job	operation	M1	M2	M3	M4
J_1	O_{11}	4	-	5	-
	O_{12}		3	4	5
	O_{13}	6	5	-	4
	O_{14}	5	-	7	2
J_2	O_{21}	3	-	4	9
	O_{22}	4	5	-	-
	O_{23}	-	4	7	3
	O_{24}	7	2	4	-
J_3	O_{31}	5	3	-	4
	O_{32}	-	4	-	5
	O_{33}	4	5	3	3
	O_{34}	-	-	5	6
J_4	O_{41}	5	-	5	6
	O_{42}	4	7	-	8
	O_{43}	9	-	5	7
	O_{44}	3	5	4	6

3 PROPOSED MODEL

The basic idea of our MAS-GATS model is to design: a set of Resource agents responsible of local optimization using Tabu search; a Supervisor agent which has two roles running simultaneously. The first is global optimization based on GA and the second is to improve the diversification in the TS. Each agent in this model has its own acquaintances (the agents that it knows and with which it can communicate), a local memory composed of its static and dynamic knowledge and a mailbox in which it stores the messages received from the other agents. In the following, we describe each agent class.

3.1 Resource Agent

This agent uses the tabu search method, which is a local search based meta-heuristic. The latter consists in exploring the search space composed of the set of solutions in order to find the optimal one. More precisely, tabu search begins from an initial solution and then chooses, at each iteration, the best solution in the current neighbourhood, even if it does not improve the quality of the current one. A neighbourhood is composed of all the solutions obtained by a simple move or transition on a solution. These solutions are named, then, neighbours.

In order to escape local optima in which the system can be easily trapped, tabu search uses a temporary me-morization structure in which it keeps track of the last visited solutions by memorizing the last moves performed: the tabu list. In fact, a tabu solution is forbidden during a number of iterations equal to the tabu list size. Then, the best solution among the ones not forbidden is selected for the next iteration. Tabu search method has many parameters that have to be defined such as the initial solution, neighbourhood function, evaluation technique, tabu list size and diversification techniques. The TS parameters used here is taken from (Ennigrou and Ghedira, 2008). Figure1 shows the global structure of tabu search used in our proposed model.

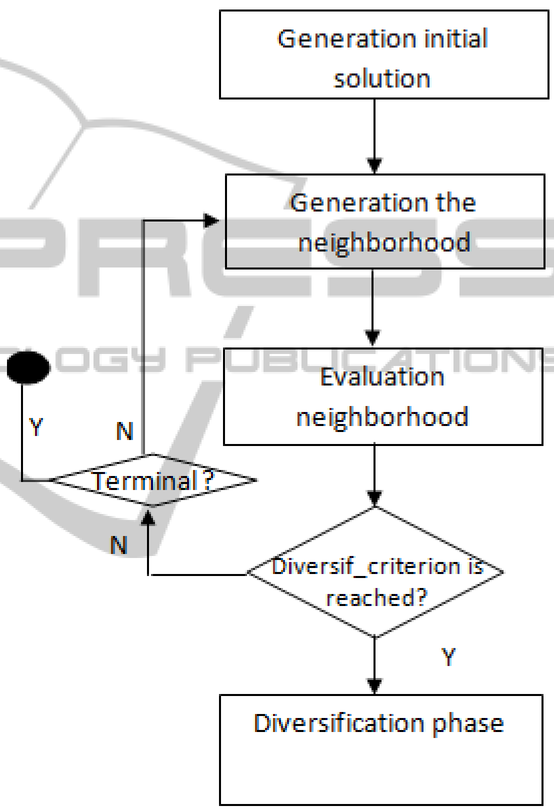
Generation initial solution
Generation the neighborhood
Evaluation neighborhood
Terminal ?
Y
N
N
Diversif_criterion is reached?
Y
Diversification phase

Figure 1: Tabu Search.

Once the initial solution has been received from the Supervisor agent, the local optimization phase begins. The Resource agent determines, then, the neighbourhood of the current solution and evaluates it in order to choose the best non-tabu neighbour or the best one which satisfies the aspiration criterion. Subsequently, the move will be stored in the tabu list and the new current solution is obtained after performing the move selected and after satisfying all problem constraints. In the case that the new current solution improves the best solution encountered so far, i.e. an elite solution, the Resource agent sends it to the other Resource agents in order to add it to their elite solution lists (it also stores it in its own elite solution list). Simultaneously, elite solutions are sent also to the Supervisor agent who keeps track of the best global solution and executes a global optimization described in the next subsection.

3.2 Supervisor Agent

Unlike the Resource agent, the Supervisor agent has a global optimization process based on genetic algorithm. Recently, genetic algorithms have become increasingly to solving combinatorial optimization problems. GA can find solutions using the mechanism of biological heredity and selection (Holland, 1975). Moreover, GA is highly evaluated for searching near optimal solutions. In the basic GA a population composed of many individuals is generated randomly. For a certain number of iteration a new population P(i+1) is generated by choosing individuals from Pi as parents for the next generation, selecting those parents or crossing them over, and possibly mutating them. Slightly different, we present the scheme of our proposed genetic algorithm in Figure2.

The initial population is generated by the multi-agent dynamic described so far. After evaluating each solution in the population, if the stop criterion is not met, there are two choices. According to the probability PGen, the current individual executes crossover operator or the mutation one. The stop criterion is that the MaxIter is reached or the best makespan has not been improved for a certain number of iteration.

In this section, we present our adaptation of the different GA parameters to our problem and later the core of global optimization.

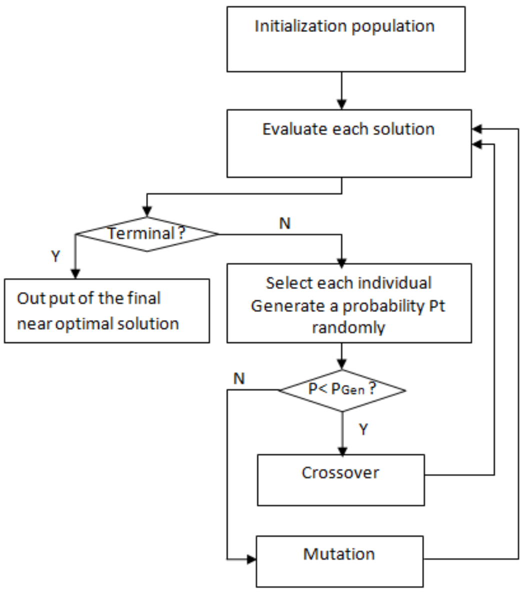
Initialization population
Evaluate each solution
Terminal ?
N
Y
Out put of the final near optimal solution
Select each individual Generate a probability Pt randomly
N
P< PGen ?
Y
Crossover
Mutation

Figure 2: Proposed Genetic Algorithm.

$$\begin{array}{c|cccc}
 & M1 & M2 & M3 & M4 \\
\hline
O_{11} & 1 & 0 & 0 & 0 \\
O_{21} & 0 & 0 & 1 & 0 \\
O_{31} & 0 & 1 & 0 & 0 \\
O_{32} & 0 & 0 & 0 & 1 \\
O_{12} & 0 & 1 & 0 & 0 \\
O_{22} & 1 & 0 & 0 & 0 \\
O_{41} & 0 & 0 & 1 & 0 \\
O_{13} & 0 & 1 & 0 & 0 \\
O_{33} & 0 & 0 & 1 & 0 \\
O_{42} & 1 & 0 & 0 & 0 \\
O_{23} & 0 & 1 & 0 & 0 \\
O_{34} & 0 & 0 & 0 & 1 \\
O_{14} & 1 & 0 & 0 & 0 \\
O_{43} & 0 & 0 & 1 & 0 \\
O_{24} & 1 & 0 & 0 & 0 \\
O_{44} & 0 & 1 & 0 & 0
\end{array}$$

Figure 3: Chromosome representation.

3.2.1 Representation of Individuals

For solving FJSP by GA, the first step is to represent a solution of a problem as a chromosome. In order to improve the number of feasible solutions produced after genetic recombination, we try to design an efficient coding of the individuals which respects the most important constraints of our problem. Then, our chromosome is designed as a binary matrix, where:

- The rows represent all operations of jobs. Furthermore, the order in which they appear in the chromosome describes the sequence of operations present in the solution. solution
- The columns represent all machines.

Moreover, in our representation, we present a constraint described as follows:

$$\sum_{k=1}^{m} X_{ijk} = 1 \qquad (1)$$

$X_{ijk} = 1$ when O_{ij} is assigned to resource m_k
$X_{ijk} = 0$ otherwise.
Otherwise, the sum of each row is equal to one, because an operation must be assigned to only one machine. The order, in which the operations appear in this representation, is found according to the start times of the operations. When we have more than one operation executed in the same time, we choose the one which has the less number of resources. Figure3 shows an example of our individuals which represent the solution figured in the Gantt chart in Figure4 and also the data from Table1.

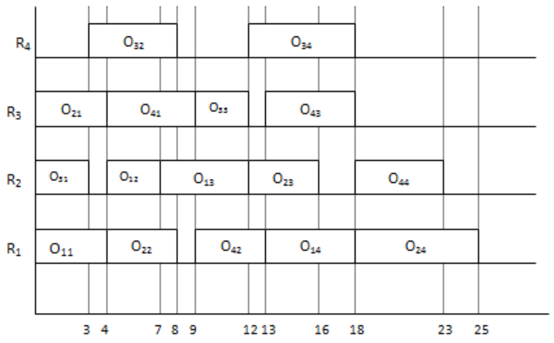

Figure 4: Gantt Chart of solution.

3.2.2 Selection

The selection phase aims to choose the chromosomes for reproduction to create the next generation. In this study, we adopt our selection operator. Four individuals are randomly chosen from parent population and the four objective values are compared to select the best and the worst solution for reproduction.

3.2.3 Crossover operator

The goal of the crossover is to obtain better chromosomes to improve the result by exchanging information contained in the current good ones. In our work, we have adopted the crossover operator order1 (Davis, 1985). The idea of this operator is as follows: we randomly select two positions XP1 and XP2 in Parent1. The middle part is copied to the offspring1. The rest is filled from the parent2 starting with position XP2 + 1 and jumping elements that are already present in the offspring 1. The same steps are repeated for the second offspring by starting with the Parent2. We adapted this crossover to our own coding described above. To more explain the crossover operator, we present an example in Figure5. Note that in this example, we take XP1 = 4 and XP2 =9.

3.2.4 Mutation

Mutation operator is used also to get a new individual having only one value different from an already exist-ing one. In our work, we adopt intelligent mutation proposed by (Pezzella et al., 2008)in which we select an operation on the machine with the maximum workload (i.e. the amount of work that a machine produces in a specified time period), and assign it to the machine with the minimum workload if possible.

3.3 Global Dynamic Multi-agent

The global dynamic begins with sending the initial solution by the Supervisor agent to all Resource agents. At this moment, the local optimization processes based on TS are launched by each Resource agent. Every time the Resource agent encounters an elite solution, it sends it to the other Resource agents and to the Supervisor agent.

After a certain number of iterations of local TSs, the global optimization process begins by generating the initial population of GA. The initial population contains in addition to the elite solutions a number of random solutions generated by the Supervisor agent. After that, the crossover and the mutation operators above described are executed as presented in Figure2. Both global and local optimization processes are executed simultaneously. In the case that the Supervisor agent encounters a solution better than the best solution encountered so far, it sends it to the other Resource agents which add it to their elite solution lists in order to be used eventually during their diversification phase. The Resource agents still send their elite solutions in order to add it to the current population of GA. The Supervisor agent and the Resource agents maintain cooperation until reaching stopping criteria. This latter can be no improvement in certain number of iterations or IterMax is reached.

4 COMPUTATIONAL RESULTS

In order to evaluate the performance of the proposed model, this section describes the computational experiments. Our proposed algorithm has been implemented using the multi-agent platform JADE (Java Agent DEvelopment framework) based on JAVA and run on core2Duo, 2,6GHZ and 2GB RAM.

The benchmark problems used were the set instances taken from:

- Brandimarte (Brandimarte, 1993) (BRdata). It consists of 10 problems mk1mk10, where the jobs range from 10 to 20 operations, machines from 6 to 15, operations for each job from 5 to 15.
- HUdata (Hurink et al., 1994): it composed of 3 problems (mt06, mt10, mt20) and 10 problems named rdata (la01-la10) where the jobs range from 6 to 15 ,machines from 5 to 10.

These well-known instance problems have been used by many papers in the literature to benchmark the proposed methods.

The performance of our algorithm is evaluated by comparing with some recent algorithms in the liter-

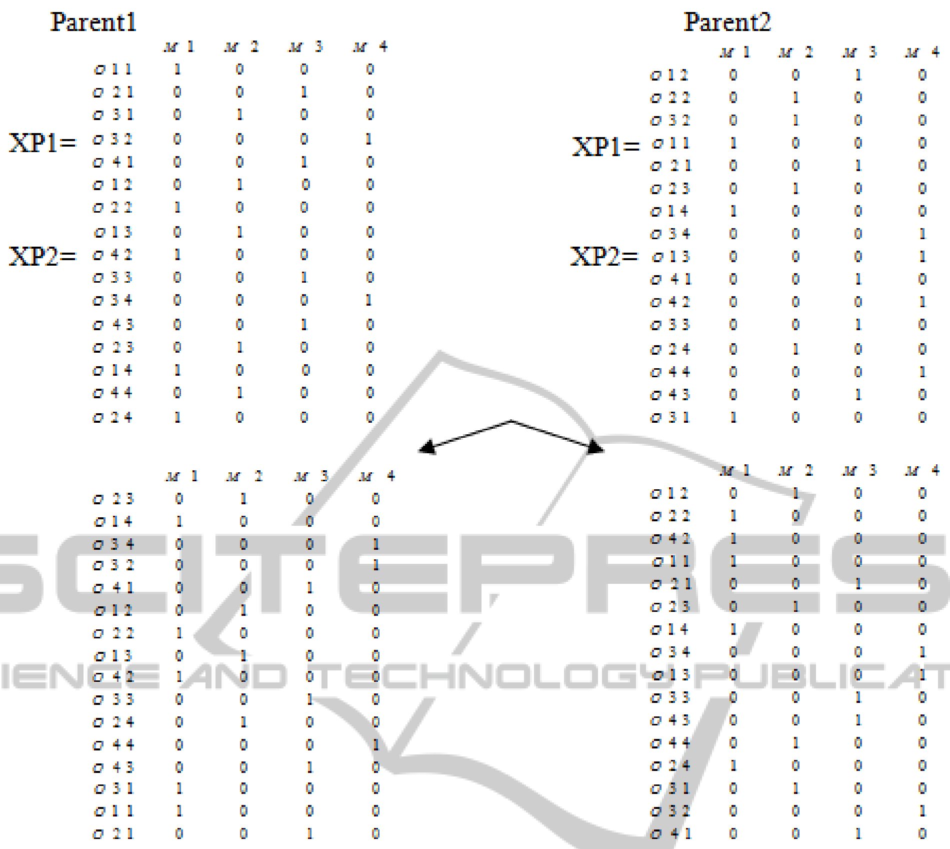

Figure 5: Crossover operator.

ature. The non-deterministic nature of our approach makes it necessary to carry out multiple runs on the same problem instance in order to obtain meaningful results.

In our experiment, we tested different values for a list of algorithm parameters, and computational experience proves that the following values are more effective:

- Tabu list size: 12;
- Number of generation of TS: 1000;
- Number iteration for diversification: 80;
- Population size: 800;
- P_{Gen}: 0.7;
- Number of generation of GA: 800.

In Table2, P denotes the level of flexibility of the instance problem. $n \times m$ denotes the problem scale (i.e. n is the number of jobs, m is the number of machines. (LB, UB) denotes the optimal makespan if known, otherwise, the best lower and upper bound found to date. We compare our proposed model MAS-GATS with:

- FJSMATSLO+ proposed by (Ennigrou and Ghedira, 2008) described in section 1,
- MAPSO proposed by (Nouiri et al., 2015) which developed a PSO based multi-agents approach.,
- Heuristics based on a constructive procedure proposed by (Ziaee, 2014),

It can be seen from Table2 that our MAS-GATS outperforms FJS MATSLO+ in 8 out of 10 problems, Heuristic approach in 6 out of 10 problems and MAPSO in 4 instance of problems. We also note that all our results are located between the lower and upper bounds in all instances of (Brandimarte, 1993). Our model has a better result for high level of flexibility and high number of jobs such as MK03, MK07. We confirm this remarks by the results presented in table3. It is evident from this table that MAS-GATS performed very well. Our results obtained are clearly better than those obtained by FJS MATSLO+ model in 85% of instances. Then, we can conclude that GA operators are more efficient than other diversification strategies. Likewise, it can be show that the cooperation between GA and TS is more efficient than simply TS. Therefore, the superior results indicate the successful hybridization of GA and TS. This is due to

Table 2: Benchmarks of Brandimarte (Brandimarte, 1993).

Instance	P	$n \times m$	(LB, UB)	FJSMATSLO+	MAPSO	Heuristic	MAS-GATS
MK01	2	10×6	36 ,42	40	41	42	39
MK02	3.5	10×6	24,32	32	26	28	28
MK03	3	15×8	204 ,211	207	207	204	204
MK04	2	15×8	48,81	67	65	75	67
MK05	1.5	15×4	168,186	188	171	179	176
MK06	3	10×15	33,86	85	61	69	75
MK07	3	20×5	133,157	154	173	149	148
MK08	1.5	20×10	523	523	523	555	523
MK09	3	20×10	299,369	437	307	342	341
MK10	1.5	20×15	165,296	380	312	242	264

Table 3: Benchmarks of HUdata de (Hurink et al., 1994) rdata.

Benchmarks	P	$n \times m$	(LB, UB)	FJS MATSLO+	MAS-GATS
mt06	2	6×6	47	47	**47**
mt10	2	10×10	(679, 686)	724	729
mt20	2	20×5	(1022, 1024)	1036	**1036**
la01	2	10×5	(570, 574)	590	**570**
la02	2	10×5	(529, 532)	543	**540**
la03	2	10×5	(477,479)	480	485
la04	2	10×5	(502, 504)	524	**514**
la05	2	10×5	(457, 458)	464	**463**
la06	2	15×5	(799,800)	806	**805**
la07	2	15×5	(746, 750)	765	**754**
la08	2	15×5	(765, 767)	775	**770**
la09	2	15×5	(853, 854)	867	**858**
la10	2	15×5	(804, 805)	811	**807**

the fact that TS has facilitated a better exploitation of the search space whereas GA has facilitated a better exploration of it.

5 CONCLUSIONS

This work combines two optimization approaches in order to get an optimal solution to FJSP: Local optimization approach based on tabu search cooperates with global optimization approach based on genetic algorithm in order to improve the searching ability in terms of obtaining a solution as near as possible from the global optimum. A multi-agent approach is proposed in this context to ensure the relation between the two different approaches. In future work, it will be interesting to investigate the dynamic scheduling problem to closely reflect the real job shop scheduling environment.

REFERENCES

Azzouz, A., Ennigrou, M., Jlifi, B., and Ghedira, K. (2012). Combining tabu search and genetic algorithm in a multi-agent system for solving flexible job shop problem. In *11th Mexican International Conference on Artificial Intelligence (MICAI)pp. 8388.*

Brandimarte, P. (1993). Routing and scheduling in a flexible job shop by tabu search. In *Journal Annals of Operations Research 22: 158-183.*

Davis, L. (1985). Applying adaptive algorithms to epistatic domains. In *In Proc. International Joint Conference on Artificial Intelligence.*

Ennigrou, M. and Ghedira, K. (2008). New local diversification techniques for flexible job shop scheduling problem with a multi-agent approach. In *In Autonomous Agents and Multi-Agent Systems, vol17(2), 270-287.*

Fattahi, P., Mohamed, s.-M., and Fariborz, J. (2007). Mathematical modeling and heuristic approaches to flexible job shop scheduling problems. In *Journal of intelligent manufacturing18(3) 331-342.*

Garey, M., Johnson, D., and Sethi, R. (1976). The complexity of flow shop and job-shop scheduling. In *Math Oper Res 1(2): 117129.*

Glover, F. (1998). A template for scatter search and path relinking. In *In Artificial Evolution. Lecture Notes in Computer Science Volume 1363,1-51.*

Glover, F. and Laguna, M. (1999). *Tabu search.* Handbook of Combinatorial Optimization 2093-2229.

Henchiri, A. and Ennigrou, M. (2013). Particle swarm optimization combined with tabu search in a multi-agent model for flexible job shop problem. In *In Advances in Swarm Intelligence (pp. 385-394). Springer Berlin Heidelberg.*

Holland, J. H. (1975). *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence.* U Michigan Press.

Hurink, J., Jurisch, B., and Thole, M. (1994). Tabu search for the job shop scheduling problem with multi-purpose machines. In *Operations-Research-Spektrum,15(4):205-215.*

Kacem, I., Hammadi, S., and Borne, P. (2002). Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems. In *Syst IEEE Syst Man Cybern 32(1) 113.*

Kelly, J. P., Lagunat, T. M., and Glover, F. (1994). A study of diversification strategies for the quadratic assignment problem. In *Computers and Operations Research, 21(8), 885-893.*

Laguna, M. and Glover, F. (1993). Integrating target analysis and tabu search for improved scheduling systems. In *In Experts Systems Applic. 6 287-297.*

Li, J.-Q., Pan, Q.-K., Suganthan, P., and Chua, T. (2010a). A hybrid tabu search algorithm with an efficient neighborhood structure for the flexible job shop scheduling problem. In *The international journal of advanced manufacturing technology 52(58) 683697.*

Li, J.-Q., Pan, Q.-K., Xie, S.-X., Jia, B.-X., and Wang, Y.-T. (2010b). A hybrid particle swarm optimization and tabu search algorithm for flexible job-shop scheduling problem. In *International Journal of Computer Theory and Engineering 2(2) 17938201.*

Najid, N., Dauzere-Peres, S., and Zaidat, A. (2002). A modified simulated annealing method for flexible job shop scheduling problem. In *In proceedings of the IEEE International Conference on Systems, Man and Cybernetics, Vol.5, 6-9.*

Nouiri, M., Bekrar, A., Jemai, A., Niar, S., and Ammari, A. C. (2015). An effective and distributed particle swarm optimization algorithm for flexible job-shop scheduling problem. In *Journal of Intelligent Manufacturing, 1-13.*

Pezzella, F., Morganti, G., and Ciaschetti, G. (2008). A genetic algorithm for the flexible job-shop scheduling problem. In *In Proc. International Joint Conference on Artificial Intelligence. Computers and Operations Research, 35(10) 3202-3212.*

Puchinger, J. and Raidl, G. (2005). Combining metaheuristics and exact algorithms in combinatorial optimization: A survey and classication. In *In Proceedings of the First International Work-Conference on the Interplay Between Natural and Articial Computation, Part II. Volume 3562 of LNCS., Springer 4153.*

Saidi-Mehrabad, M. and Fattahi, P. (2006). Flexible job shop scheduling with tabu search algorithms. In *The international journal of Advanced Manufacturing technology , (32) 563-570.*

Shao, X. Y., Liu, W. Q., Liu, Q., and Y., Z. C. (2013). Hybrid discrete particle swarm optimization for multi-objective flexible job-shop scheduling problem. In *The International Journal of Advanced Manufacturing Technology, vol. 67, no. 912, 28852901.*

Sohn, D., Hirasawa, K., and Hu, J. (2005). Adaptive random search with intensification and diversification combined with genetic algorithm. In *In Proc. of the congress on evolutionary computation (CEC05), 1462-1469.*

Vilcot, G. and Billaut, J. C. (2011). A tabu search algorithm for solving a multicriteria flexible job shop scheduling problem. In *International Journal of Production Research. Vol. 49, Iss. 23: 6963-6980.*

Wang, S. and Yu, J. (2010). An effective heuristic for flexible job-shop scheduling problem with maintenance activities. In *Computers and Industrial Engineering (59) 436447.*

Wang, S., Zhou, B., and Xi, L. (2008). A filtered-beam-search-based algorithm for flexible job-shop scheduling problem. In *International Journal of Production Research, 46 (11) 30273058.*

Xia, W. and Wu, Z. (2005). An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems. In *Computers and industrial Engineering 48(2) 409-425.*

Yazdani, M., Gholami, M., Zandieh, M., and M., M. (2009). A simulated annealing algorithm for flexible job-shop scheduling problem. In *Journal of Applied Sciences, 9. 662-670.*

Zambrano rey, G., Bekrar, A., Prabhu, V., and Trentesaux, D. (2014). Coupling a genetic algorithm with the distributed arrival-time control for the jit dynamic scheduling of flexible job-shops. In *International Journal of Production Research, 52(12), 36883709.*

Zhang, G., Liang, G., and Yang, S. (2011). An effective genetic algorithm for the flexible job-shop scheduling problem. In *Expert Syst. Appl. 38(4) 3563-3573.*

Zhang, G., Shao, X., Li, P., and Gao., L. (2009). An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem. In *Computers and Industrial Engineering. Volume 56, Issue 4. 13091318.*

Zhang, G., Shi, Y., and Gao, L. (2008). A genetic algorithm and tabu search for solving flexible job shop schedules. In *International Symposium on Computational Intelligence and Design.*

Zhou, W., Yan-ping, B., and Ye-qing, Z. (2013). An improved genetic algorithm for solving flexible job shop scheduling problem. In *25th Chinese on Control and Decision Conference (CCDC), page(s): 4553 4558.*

Ziaee, M. (2014). A heuristic algorithm for solving flexible job shop scheduling problem. In *Int Adv Manuf Technol 71: 519528.*

Zribi, N., Kacem, I., El Kamel, A., and Borne, P. (2007). Assignment and scheduling in flexible job-shops by hierarchical optimization. In *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions, Issue: (4) 652 661.*