Simultaneous Scheduling of Machines and a Single Moving Robot in a Job Shop Environment by Metaheuristics based Clustered Holonic Multiagent Model

Houssem Eddine Nouri, Olfa Belkahla Driss and Khaled Ghédira
Stratégies d'Optimisation et Informatique Intelligente, Institut Supérieur de Gestion de Tunis, Université de Tunis, 41, Avenue de la Liberté, Cité Bouchoucha, Bardo 2000, Tunis, Tunisie

Keywords: Scheduling, Transport, Robot, Genetic Algorithm, Tabu Search, Holonic Multiagent.

Abstract: In systems based robotic cells, the control of some elements such as transport robot has some difficulties when planning operations dynamically. The Job Shop scheduling Problem with Transportation times and a Single Robot (JSPT-SR) is a generalization of the classical Job Shop scheduling Problem (JSP) where a set of jobs additionally have to be transported between machines by a single transport robot. Hence, the JSPT-SR is more computationally difficult than the JSP presenting two NP-hard problems simultaneously: the job shop scheduling problem and the robot routing problem. This paper proposes a hybrid metaheuristic approach based on clustered holonic multiagent model for the JSPT-SR. Firstly, a scheduler agent applies a Neighborhood-based Genetic Algorithm (NGA) for a global exploration of the search space. Secondly, a set of cluster agents uses a tabu search technique to guide the research in promising regions. Computational results are presented using benchmark data instances from the literature of JSPT-SR. New upper bounds are found, showing the effectiveness of the presented approach.

1 INTRODUCTION

Scheduling is a field of investigation which has known a significant growth these last years. The scheduling problems appear in all the economic areas, from computer engineering to industrial production and manufacturing. The Job Shop scheduling Problem (JSP), which is among the hardest combinatorial optimization problems (Sonmez and Baykasoglu, 1998), is a branch of the industrial production scheduling problems. The JSP is known as one of the most popular research topics in the literature due to its potential to dramatically decrease costs and increase throughput (Jones and Rabelo, 1998). The Job Shop scheduling Problem with Transportation times and a Single Robot (JSPT-SR) is a generalization of the classical JSP where a set of jobs additionally have to be transported between machines by a single transport robot. Hence, the JSPT-SR is more computationally difficult than the JSP presenting an additional difficulty caused by a set of jobs to be transported by a single robot between a set of available machines. In the JSPT-SR, we have to consider two NP-hard problems simultaneously: the job shop scheduling problem (Lenstra and Kan, 1979) and the robot routing problem, which is similar to the pickup and delivery problem (Lenstra and Kan, 1981).

For the literature of the Job Shop scheduling Problem with Transportation times and a Single Robot, most of the researchers have considered the machine and robot scheduling as two independent problems. Therefore, only few researchers have emphasized the importance of simultaneous scheduling of jobs and the single robot.

To solve this problem, mathematical formulations are used to find optimal solutions for this problem, but the complexity of some large instances allowed to increase the processing time for some important solutions. (Raman et al., 1986) proposed a mixed integer programming formulation for this problem, and they assumed that the robot always returns to the load/unload station after transferring a load, which reduces the flexibility of the robot and influences the overall schedule length. An integer programming model is formulated by (Bilge and Ulusoy, 1995) for the machine and robot scheduling problems with a set of time window constraints. According to the authors, the resulting model is intractable in practice, because of its nonlinearity and its size. (Caumond et al., 2009) adapted a mathematical formula-

Nouri, H., Driss, O. and Ghédira, K.
Simultaneous Scheduling of Machines and a Single Moving Robot in a Job Shop Environment by Metaheuristics based Clustered Holonic Multiagent Model.
DOI: 10.5220/0005694300510062
In *Proceedings of the 8th International Conference on Agents and Artificial Intelligence (ICAART 2016) - Volume 2*, pages 51-62
ISBN: 978-989-758-172-4

tion for a shop scheduling problem with one transporter robot. This formulation differed from the published works because it considered the maximum number of jobs authorized in the system, the upstream and downstream storage capacities and the robot loaded/unloaded movements.

Moreover, heuristic and metaheuristic methods offered new opportunity to find solutions for this problem in a reasonable time but they did not guaranteed the optimality. (Pundit and Palekar, 1990) implemented a Branch and Bound procedure for a simultaneous scheduling of machines and resources handling in a Job shop environment. But, they did not take into consideration the violation of precedence relations between the different machines operations belonging to the same job. An iterative heuristic is used by (Bilge and Ulusoy, 1995) based on the decomposition of the master problem into two sub-problems, allowing a simultaneous resolution of this scheduling problem with time windows. (Ulusoy et al., 1997) adapted a genetic algorithm for this scheduling problem in a flexible manufacturing system, and where they used a chromosome representation composed by two parts, the operation task sequencing and the transport resource assignment. (Anwar and Nagi, 1998) treated the simultaneous machine and robot scheduling problem using a forward propagation heuristic, and where they supposed that the robot movements between cells are considered as additional machines. A local search algorithm is proposed in (Hurink and Knust, 2002) and (Hurink and Knust, 2005) for the job shop scheduling problem with a single robot, where they supposed that the robot movements can be considered as a generalization of the travelling salesman problem with time windows, and additional precedence constraints must be respected. The used local search is based on a neighborhood structure inspired from (Mastrolilli and Gambardella, 2000) to make the search process more effective. (Abdelmaguid et al., 2004) addressed the problem of simultaneous scheduling of machines and identical robots in flexible manufacturing systems, by developing a hybrid approach composed by a genetic algorithm and a heuristic. The genetic algorithm is used for the jobs scheduling problem and the robot assignment is made by the heuristic algorithm. A hybrid multi-objective genetic algorithm is proposed by (Reddy and Rao, 2006) to solve this combined problem, and considered three minimization objectives, which are the makespan, mean flow time and mean tardiness. (Lacomme et al., 2007) studied the job shop scheduling problem with several transport robots, where they used a local search algorithm based on a neighbourhood generated by permutation of two operations or by assigning another robot to a transport operation. (Deroussi et al., 2008) addressed the simultaneous scheduling problem of machines and robots in flexible manufacturing systems, by proposing new solution representation based on robots rather than machines. Each solution is evaluated using a discrete event approach. An efficient neighbouring system is then implemented into three different metaheuristics: iterated local search, simulated annealing and their hybridisation. A differential evolution algorithm is developed by (Babu et al., 2010) for the machines and two robots scheduling problem, this algorithm is inspired by (Storn and Price, 1995) which was proposed for the chebyshev polynomial fitting problem. (Deroussi and Norre, 2010) considered the flexible Job shop scheduling problem with transport robots, where each operation can be realized by a subset of machines and adding the transport movement after each machine operation. To solve this problem, an iterative local search algorithm is proposed based on classical exchange, insertion and perturbation moves. Then a simulated annealing schema is used for the acceptance criterion. A hybrid metaheuristic approach is proposed by (Zhang et al., 2012) for the flexible Job Shop problem with transport constraints and bounded processing times. This hybrid approach is composed by a genetic algorithm to solve the assignment problem of operations to machines, and then a tabu search procedure is used to find new improved scheduling solutions. (Lacomme et al., 2013) solved the machines and robots simultaneous scheduling problem in flexible manufacturing systems, by adapting a memetic algorithm using a genetic coding containing two parts: a resource selection part for machine operations and a sequencing part for transport operations. (Zhang et al., 2014) considered the job shop scheduling problem with transport robots and bounded processing times. A modified shifting bottleneck procedure is used coupled with a heuristic for assigning and sequencing transportation tasks iteratively.

Furthermore, a newly maturing area of the distributed artificial intelligence are used, providing some effective mechanisms for the management of such dynamic operations in manufacturing environments, such as the multi agent systems. (Braga et al., 2008) treated the machines and robots scheduling problem in flexible manufacturing systems. They proposed a distributed model based on cooperative agents, composed by five agents: an order-agent, a store-agent, a set of machine-agents and a set of robot-agents, using negotiation between them in order to obtain a best scheduling solution for this problem. (Komma et al., 2011) formulated the machines and robots scheduling problem in flexible manufacturing systems as a multi-agent system, allowing to realize an Agent-Based Shop Floor Simulator (ABSFSim). This simulator is composed by eight agents classified into three categories: The first category containing agents with a single instance such as the part-generator-agent, the arrival-queue-agent and departure-agent. A second category taking agents with multiple instances and a long lifetime such as the machine-agent, the robot-agent, the node-agent and segment-agent. And for the third category, it contained agents with multiple instances and a short lifetime such as the part-agent. A multi-agent approach is proposed by (Erol et al., 2012) for robots and machines scheduling problem within a manufacturing system. The proposed multi-agent approach worked under a real-time environment and generated feasible schedules using negotiation/bidding mechanisms between agents. This approach is composed by four agents: a manager-agent, a robot-system-holon, an order-system-holon and a machine-system-holon.

In this paper, we propose a hybridization of two metaheuristics based on clustered holonic multiagent model for the job shop scheduling problem with a single transport robot. This new approach follows two principal hierarchical steps, where a genetic algorithm is applied by a scheduler agent for a global exploration of the search space. Then, a tabu search technique is used by a set of cluster agents to guide the research in promising regions. Numerical tests were made to evaluate the performance of our approach based on the instances of (Hurink and Knust, 2005), completed by comparisons with other approaches.

The rest of the paper is organized as follows. In section 2, we define the formulation of the JSPT-SR with its objective function and a simple problem instance. Then, in section 3, we detail the proposed hybrid approach with its holonic multiagent levels. The experimental and comparison results are provided in section 4. Finally, section 5 ends the paper with a conclusion.

2 PROBLEM FORMULATION

There is a set of n jobs $J = \{J_1, \dots, J_n\}$ to be processed without preemption on a set $M = \{M_0, M_1, \dots, M_m\}$ of $m+1$ machines (M_0 represents the load/unload or LU station from which jobs enter and leave the system). Each job J_i is formed by a sequence of ni operations $\{O_{i,1}, O_{i,2}, \dots, O_{i,ni}\}$ to be performed successively according to the given sequence. For each operation $O_{i,j}$, there is a machine $\mu_{ij} \in \{M_0, \dots, M_m\}$ and a processing time p_{ij} associated with it. In addition, each job J_i $(J_1, \dots, J_n)$ is composed by $ni-1$ transport operations $\{T_{i,1}, T_{i,2}, \dots, T_{i,ni-1}\}$ to be made by a robot R from one machine to another. In fact, for each transport operation $T_{i,j}$ there is two types of movements: travel transport operation and empty transport operation.

Firstly, travel transport operation $t_{\mu_{i,j},\mu_{i,j+1}}$ must be considered for the robot R when an operation $O_{i,j}$ is processed on machine $\mu_{i,j}$ and operation $O_{i,j+1}$ is processed on machine $\mu_{i,j+1}$. These transportation times are job-independent and robot-dependent. Each transportation operation is assumed to be processed by only one transport robot R which can handle at most one job at one time. For convenience, $t_{\mu_{i,j},\mu_{i,j+1}}$ is used to denote both a transportation operation and a transportation time.

Secondly, empty transport operation $t'_{i,j}$ have to be considered while the robot R moves from machine M_i to machine M_j without carrying a job. So, it is possible to assume that $t'_{i,i} = 0$ and $t_{i,j} \geq t'_{i,j}$.

As in job shop problems, we assume that sufficient buffer space exists between machines. This assumption is also stated as an "unlimited input/output buffer capacity". Jobs processed on one machine M_i are assumed to wait until the robot affected to this transport operation is available to do it. No additional time is required to transfer job from machine to the unlimited output buffer. In a similar way, each machine M_i has an unlimited input buffer to store jobs in waiting to be processed by it. All data p_{ij}, $t_{\mu_{i,j},\mu_{i,j+1}}$, $t'_{\mu_{i,j},\mu_{i,j+1}}$ are assumed to be non-negative integers.

The objective is to determine a feasible schedule which minimizes the makespan $Cmax = Max_{j=1,n}\{C_j\}$ where C_j denotes the completion time of the last operation $O_{i,ni}$ of job J_i including the processing times of machine operations and transport operations.

- Operation completion time

$$T_{i,j} = t_{\mu_{i,j},\mu_{i,j+1}} + t'_{\mu_{i,j},\mu_{i,j+1}} \tag{1}$$

$$O_{i,j} = p_{i,j} \tag{2}$$

- Job completion time

$$C_i = \sum_{i=1}^{n} \sum_{i,1}^{i,ni} O_{i,j} + T_{i,j} \tag{3}$$

for $T_{i,j}, j \in (1, \dots, ni-1)$

$$Makespan = max(C_1, C_2, C_3, \dots, C_n) \tag{4}$$

Where i = job, j = operation, $t_{\mu_{i,j},\mu_{i,j+1}}$ = traveling moving time, $t'_{\mu_{i,j},\mu_{i,j+1}}$ = empty moving time, $T_{i,j}$ = transport operation processing time, $p_{i,j}$ = machine operation processing time.

3 HYBRID METAHEURISTICS BASED CLUSTERED HOLONIC MULTIAGENT MODEL

(Glover et al., 1995) elaborated a study about the nature of connections between the genetic algorithm and tabu search metaheuristics, searching to show the existing opportunities for creating a hybrid approach with these two standard methods to take advantage of their complementary features and to solve difficult optimization problems. After this pertinent study, the combination of these two metaheuristics has become more well-known in the literature, which has motivated many researchers to try the hybridization of these two methods for the resolution of different complex problems in several areas.

(Ferber, 1999) defined a multiagent system as an artificial system composed of a population of autonomous agents, which cooperate with each other to reach common objectives, while simultaneously each agent pursues individual objectives. Furthermore, a multiagent system is a computational system where two or more agents interact (cooperate or compete, or a combination of them) to achieve some individual or collective goals. The achievement of these goals is beyond the individual capabilities and individual knowledge of each agent (Botti and Giret, 2008).

(Koestler, 1967) gave the first definition of the term “holon” in the literature, by combining the two Greek words “hol” meaning whole and “on” meaning particle or part. He said that almost everything is both a whole and a part at the same time. In fact, a holon is recursively decomposed at a lower granularity level into a community of other holons to produce a holarchy (Calabrese, 2011). Moreover, a holon may be viewed as a sort of recursive agent, which is a super-agent composed by a sub-agents set, where each sub-agent has its own behavior as a complementary part of the whole behaviour of the super-agent. Holons are agents able to show an architectural recursiveness (Giret and Botti, 2004).

In this work, we propose a hybrid metaheuristic approach based on clustering processing two general steps: a first step of global exploration using a genetic algorithm to find promising areas in the search space and a clustering operator allowing to regroup them in a set of clusters. In the second step, a tabu search algorithm is applied to find the best individual solution for each cluster. The global process of the proposed approach is implemented in two hierarchical holonic levels adopted by a recursive multiagent model, named a hybrid Genetic Algorithm with Tabu Search based on clustered Holonic Multiagent model (GATS+HM), see Figure 1. The first holonic

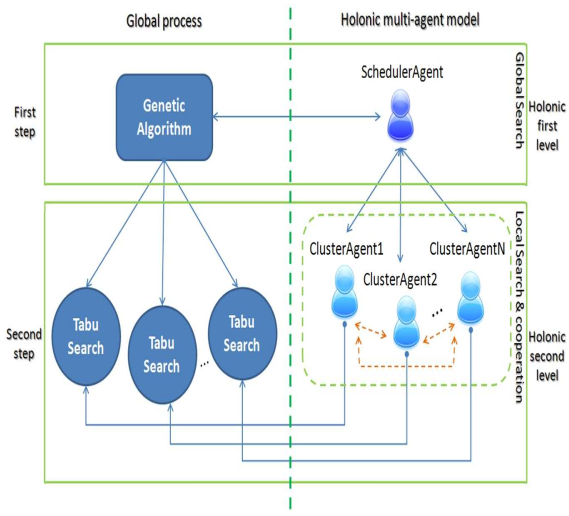


Figure 1: Hybrid metaheuristics based clustered holonic multiagent model.

level is composed by a Scheduler Agent which is the Master/Super-agent, preparing the best promising regions of the search space, and the second holonic level containing a set of Cluster Agents which are the Workers/Sub-agents, guiding the search to the global optimum solution of the problem. Each holonic level of this model is responsible to process a step of the hybrid metaheuristic approach and to cooperate between them to attain the global solution of the problem.

In fact, the choice of this new metaheuristic hybridization is justified by that the standard metaheuristic methods use generally the diversification techniques to generate and to improve many different solutions distributed in the search space, or by using local search techniques to generate a more improved set of neighbourhood solutions from an initial solution. But they did not guarantee to attain promising areas with good fitness converging to the global optimum despite the repetition of many iterations, that is why they need to be more optimized. So, the novelty of our approach is to launch a genetic algorithm based on a diversification technique to only explore the search space and to select the best promising regions by the clustering operator. Then, applying the intensification technique of the tabu search allowing to relaunch the search from an elite solution of each cluster autonomously to attain more dominant solutions of the search space. The use of a multiagent system gives the opportunity for distributed and parallel treatments which are very complimentary for the second step of the proposed approach. Indeed, our combined metaheuristic approach follows the paradigm of “Master” and “Workers” which are two recursive hierarchical levels adaptable for a holonic multiagent model, where the Scheduler Agent is the Master/Super-agent of its society and the Cluster Agents are its Workers/Sub-agents.

3.1 Non Oriented Disjunctive Graph

In this work, we chose to use the disjunctive graph of (Hurink and Knust, 2005) for the job shop problem with transportation times and one robot. To explain this graph, a sample problem of three jobs and five machines with their transportation times for a single robot R is presented in Table 1.

Table 1: One instance of job shop problem with a single robot.

Processing times for each job J_i

Job1 :	M5(1)	M4(4)	
Job2 :	M1(1)	M3(6)	M2(5)
Job3 :	M3(6)	M1(3)	

Transportation times for the robot R

	M1	M2	M3	M4	M5
M1	0	1	2	3	4
M2	1	0	1	2	3
M3	2	1	0	1	2
M4	3	2	1	0	1
M5	4	3	2	1	0

The disjunctive graph $G = (V_m \cup V_t, C \cup D_m \cup D_r)$, see Figure 2, is composed by : a set of vertices V_m containing all machine operations, a set of vertices V_t is the set of transport operations obtained by assigning the robot R to each transport operation, and two dummy nodes 0 and $*$. Also, this graph consists of: a set of conjunctions C representing precedence constraints $O_{i,k} \rightarrow t_{\mu_{i,k},\mu_{i,k+1}} \rightarrow O_{i,k+1}$, undirected disjunctions for machines D_m, and undirected disjunctions for the transport robot D_r. For each job J_i, $ni-1$ transport operations $t_{\mu_{i,k},\mu_{i,k+1}}$ are introduced including precedence $O_{i,k} \rightarrow t_{\mu_{i,k},\mu_{i,k+1}} \rightarrow O_{i,k+1}$. In fact, the robot R may be considered as an additional "machine" which has to process all these transport operations. The arcs from machine node to transport node are weighted with the machine operation durations. Edges between machine operations represent disjunctions for machine operations which have to be processed on the same machine and cannot use it simultaneously.

As for the classical job shop, the conjunctions C model the execution order of operations within ech job J_i. In addition to the classical set of undirected machine disjunctions D_m (all pairs of machine operations which have to be processed on the same machine and which are not linked by a directed path), it is necessary to consider the set of undirected robot disjunctions D_r (all pairs of transport operations which have to be transported by the robot R and which are not linked by a directed path). To solve the scheduling problem it is necessary to turn all undirected arcs in $D_m \cup D_r$ into directed ones, and to assign the robot R to each transport operation, where the final graph becomes an oriented disjunctive graph.

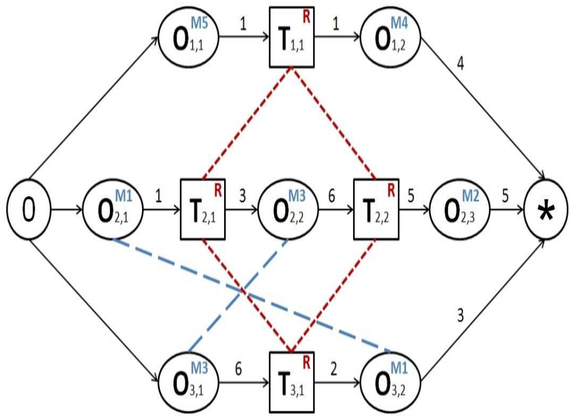


Figure 2: Non oriented disjunctive graph.

3.2 Scheduler Agent

The Scheduler Agent (SA) is responsible to process the first step of the hybrid algorithm by using a genetic algorithm called NGA (Neighborhood-based Genetic Algorithm) to identify areas with high average fitness in the search space during a fixed number of iterations *MaxIter*, see Figure 3. In fact, the goal of using the NGA is only to explore the search space, but not to find the global solution of the problem. Then, a clustering operator is integrated to divide the best identified areas by the NGA in the search space to different parts where each part is a cluster $CL_i \in CL$ the set of clusters, where $CL = \{CL_1, CL_2, \ldots, CL_N\}$. In addition, this agent plays the role of an interface between the user and the system (initial parameter inputs and final result outputs). According to the number of clusters N obtained after the integration of the clustering operator, the SA creates N Cluster Agents (CAs) preparing the passage to the next step of the global algorithm. After that, the SA remains in a waiting state until the reception of the best solutions found by the CAs for each cluster CL_i. Finally, it finishes the process by displaying the final solution of the problem.

3.2.1 Individual's Solution Presentation based Oriented Disjunctive Graph

The Job Shop scheduling Problem with Transportation times and a Single Robot is composed by two

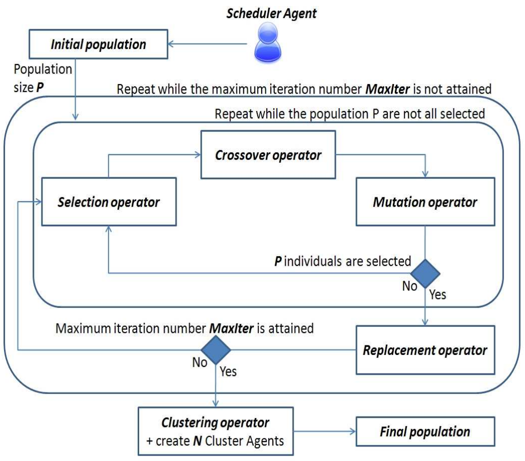
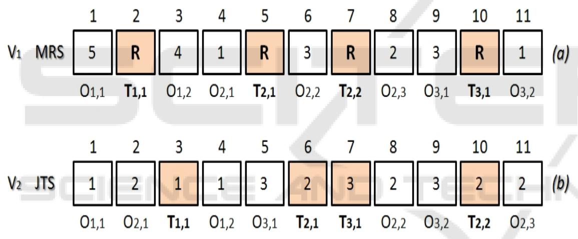

Scheduler Agent
Initial population
Population size *P*
Repeat while the maximum iteration number *MaxIter* is not attained
Repeat while the population P are not all selected
Crossover operator
Selection operator
Mutation operator
P individuals are selected
No
Yes
Maximum iteration number *MaxIter* is attained
Replacement operator
No
Yes
Clustering operator
+ create *N* Cluster Agents
Final population

Figure 3: First step of the global process by the Scheduler Agent.

sub-problems: firstly the machines and robot selection, secondly the operations scheduling problem, that is why the chromosome representation is encoded in two parts: Machines and Robot Selection part (MRS), and Job and Transport operation Sequence part (JTS), see Figure 4.



	1	2	3	4	5	6	7	8	9	10	11	
V₁ MRS	5	R	4	1	R	3	R	2	3	R	1	(a)
	$O_{1,1}$	$T_{1,1}$	$O_{1,2}$	$O_{2,1}$	$T_{2,1}$	$O_{2,2}$	$T_{2,2}$	$O_{2,3}$	$O_{3,1}$	$T_{3,1}$	$O_{3,2}$	

	1	2	3	4	5	6	7	8	9	10	11	
V₂ JTS	1	2	1	1	3	2	3	2	3	2	2	(b)
	$O_{1,1}$	$O_{2,1}$	$T_{1,1}$	$O_{1,2}$	$O_{3,1}$	$T_{2,1}$	$T_{3,1}$	$O_{2,2}$	$O_{3,2}$	$T_{2,2}$	$O_{2,3}$	

Figure 4: The chromosome representation of a scheduling solution.

The first part MRS is a vector V_1 with a length L equal to the total number of operations and where each index represents the selected machine or robot to process an operation indicated at position p, see Figure 4 (a). For example $p = 3$ and $p = 7$, $V_1(3)$ is the selected machine M_4 for the operation $O_{1,2}$ and $V_1(7)$ is the selected robot R_1 for the operation $T_{2,2}$. The second part JTS is a vector V_2 having the same length of V_1 and where each index represents a machine operation $O_{i,j}$ or a transport operation $T_{i,j}$ according to the predefined operations for each job, see Figure 4 (b). For example this operation sequence 1-2-1-1-3-2-3-2-3-2-2 can be translated to: $(O_{1,1}, M_5) \rightarrow (O_{2,1}, M_1) \rightarrow (T_{1,1}, R) \rightarrow (O_{1,2}, M_4) \rightarrow (O_{3,1}, M_3) \rightarrow (T_{2,1}, R) \rightarrow (T_{3,1}, R) \rightarrow (O_{2,2}, M_3) \rightarrow (O_{3,2}, M_1) \rightarrow (T_{2,2}, R) \rightarrow (O_{2,3}, M_2)$. In addition, for each job J_i $(J_1, \dots, J_i)$ $ni - 1$ transport operations are generated $T_{1,1}$, $T_{2,1}$, $T_{2,2}$ and $T_{3,1}$, and scheduled following the presented solution in vector JTS, allowing to fix the final path to be considered by the robot R during the shop process, see Figure 5.

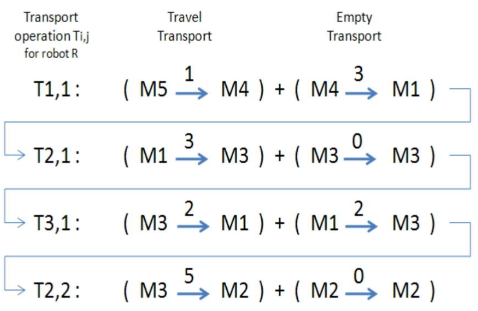

Transport operation Ti,j for robot R
Travel Transport
Empty Transport
T1,1 : (M5 →1 M4) + (M4 →3 M1)
T2,1 : (M1 →3 M3) + (M3 →0 M3)
T3,1 : (M3 →2 M1) + (M1 →2 M3)
T2,2 : (M3 →5 M2) + (M2 →0 M2)

Figure 5: Final path of the robot R.

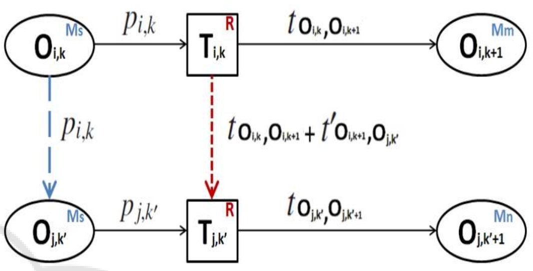


Figure 6: Exemple of oriented machine and robot disjunctions.

To model an oriented disjunctive graph we should consider some rules. Let the example in Figure 6, if the edge is oriented in the direction $O_{i,k} \rightarrow O_{j,k'}$ it gets the weight $p_{i,k}$, else it takes $p_{j,k'}$ in the inverse case. If an arc is added from $T_{i,k}$ to $T_{j,k'}$, its gets the weight $t_{O_{i,k},O_{i,k+1}} + t'_{O_{i,k+1},O_{j,k'}}$ and $t_{O_{j,k'},O_{j,k'+1}} + t'_{O_{j,k'+1},O_{i,k}}$ if it is oriented in the other direction. Thus, basing on (Hurink and Knust, 2005) we can define a fixed machine selection S_m called directed Machine Disjunctions and a fixed transport selection S_r called directed Transport Disjunctions, with their precedence relations C called operation Conjunctions. So, a fully oriented disjunctive graph can be obtained using $\hat{S} = C \cup S_m \cup S_r$, which is called a complete selection. In fact, the selections of the two sets of disjunctions S_m and S_r with their set of conjunctions C are based on the two proposed vectors MRS and JTS, where MRS allows to present the selected machines to process job operations and the selected robot to process transport operations. JTS presents the execution order of the job and transport operations in their selected machines and robot allowing to fix the final Machine and Transport Disjunctions $S_m \cup S_r$ with their set of Conjunctions C representing the precedence relations between the different operations. The union $C \cup S_m \cup S_r = \hat{S}$ fully describes a solution if the resulting oriented disjunctive graph $G = (V_m, V_t, \hat{S})$ is acyclic. A feasible schedule can be constructed by longest path calculation which permits to obtain the earliest starting time of both machine and transport operations and fully defines a semi-active schedule with the Cmax given by the length of the longest path from node 0 to *, see Figure 7.

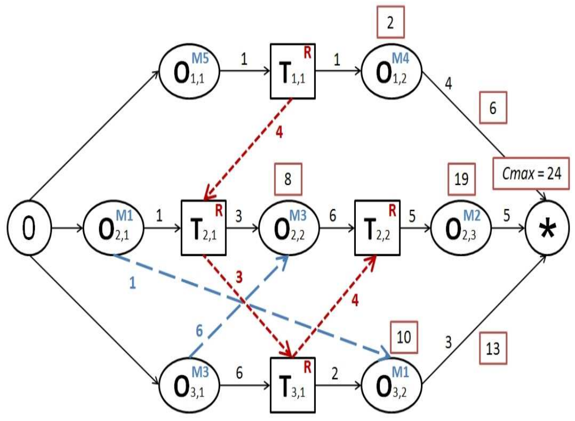


Figure 7: Oriented disjunctive graph.

Noting that the chromosome fitness is calculated by $Fitness(i)$ which is the fitness function of each chromosome i and $Cmax(i)$ is its makespan value, where $i \in \{1,\dots,P\}$ and P is the total population size, see Equation (5).

$$Fitness(i) = \frac{1}{Cmax(i)} \quad (5)$$

3.2.2 Population Initialization

The initial population is generated randomly following a uniform law and based on a neighborhood parameter to make the individual solutions more diversified and distributed in the search space. In fact, each new solution should have a predefined distance with all the other solutions to be considered as a new member of the initial solution. The used method to determinate the neighborhood parameter is inspired from (Bozejko et al., 2010), which is based on the permutation level of operations to obtain the distance between two solutions. In fact, the dissimilarity distance is calculated by verifying the difference between two chromosomes in terms of the execution order of all the shop operations $O_{i,j}$ and $T_{i,j}$ in the Job and Transport operation Sequence V_2 (JTS). So, if there is a difference in the vector V_2, the distance is incremented by 1 because it is in the order of O(1). Let $Chrom_1(JTS_1)$ and $Chrom_2(JTS_2)$ two chromosomes of two different scheduling solutions, L is the total number of operations of all jobs and $Dist$ is the dissimilarity distance. The distance is calculated by verifying the execution order difference of the Job and Transport operation Sequence vectors JTS_1 and JTS_2 which is in order of O(1), we give here how to proceed in Algorithm 1 :

Algorithm 1: How to calculate the dissimilarity distance between two solutions.

```
1: procedure
2:     $Dist \leftarrow 0, k \leftarrow 1$
3:     for k from 1 to L do
4:         if Chrom1($JTS_1$(k)) $\neq$ Chrom2($JTS_2$(k)) then
5:             $Dist \leftarrow Dist + 1$
6:         end if
7:     end for
8:     return $Dist$
9: end procedure
```

Noting that $Distmax$ is the maximal dissimilarity distance and it is calculated by Equation (6), representing 100% of difference between two chromosomes.

$$Distmax = \sum_{i=1}^{n} \sum_{i,1}^{i,ni} 1 \quad (6)$$

3.2.3 Selection Operator

The selection operator is used to select the best parent individuals to prepare them to the crossover step. This operator is based on the fitness function allowing to analyze the quality of each selected solution. But progressively the fitness values will be similar for the most individuals. That is why, we integrate the neighborhood parameter, where we propose a new combined parent selection operator named Fitness-Neighborhood Selection Operator (FNSO) allowing to add the dissimilarity distance parameter to the fitness function to select the best parents for the crossover step. The FNSO chooses in each iteration two parent individuals until engaging all the population to create the next generation. The first parent takes successively in each case a solution i, where $i \in \{1,\dots,P\}$ and P is the total population size. The second parent obtains its solution j randomly by the roulette wheel selection method based on the two Fitness and Neighborhood parameters relative to the selected first parent, where $j \in \{1,\dots,P\} \setminus \{i\}$ in the P population and where $j \neq i$. In fact, to use this random method, we should calculate the Fitness-Neighborhood total FN for the population, see Equation (7), the selection probability sp_k for each individual I_k, see Equation (8), and the cumulative probability cp_k, see Equation (9). After that, a random number r will be generated from the uniform range [0,1]. If $r \leq cp_1$ then the second parent takes the first individual I_1, else it gets the k^{th} individual $I_k \in$

$\{I_2, \ldots, I_P\} \setminus \{I_i\}$ and where $cp_{k-1} < r \leq cp_k$. For *Equations (7), (8)* and *(9)*, $k = \{1, 2, \ldots, P\} \setminus \{i\}$.

- The Fitness-Neighborhood total for the population:

$$FN = \sum_{k=1}^{P} [1/(Cmax[k] \times Neighborhood[i][k])] \tag{7}$$

- The selection probability sp_k for each individual I_k:

$$sp_k = \frac{1/(Cmax[k] \times Neighborhood[i][k])}{FN} \tag{8}$$

- The cumulative probability cp_k for each individual I_k:

$$cp_k = \sum_{h=1}^{k} sp_h \tag{9}$$

3.2.4 Crossover Operator

The crossover operator has an important role in the global process, allowing to combine in each case the chromosomes of two parents in order to obtain new individuals and to attain new better parts in the search space. In this work, this operator is applied only for the parent chromosome vector V_2 (JTS).

JTS Crossover. An improved precedence preserving order-based on crossover (iPOX), inspired from (Lee et al., 1998), is adapted for the parent operation vector JTS. This iPOX operator is applied following four steps, a first step is selecting two parent operation vectors (JTS_1 and JTS_2) and generating randomly two job sub-sets Js_1/Js_2 from all jobs. A second step is allowing to copy any element in JTS_1/JTS_2 that belong to Js_1/Js_2 into child individual JTS'_1/JTS'_2 and retain them in the same position. Then the third step deletes the elements that are already in the sub-set Js_1/Js_2 from JTS_1/JTS_2. Finally, fill orderly the empty positions in JTS'_1/JTS'_2 with the reminder elements of JTS_2/JTS_1 in the fourth step, see the example in the Figure 8.

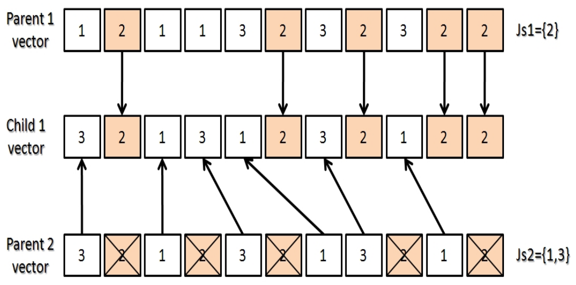


Figure 8: JTS crossover exemple.

3.2.5 Mutation Operator

The mutation operator is integrated to promote the children generation diversity. In fact, this operator is applied on the chromosomes of the new generated children by the JTS crossover operator.

JTS Mutation. This operator selects randomly two indexes $index_1$ and $index_2$ from the vector JTS. Next, it changes the position of the job number in the $index_1$ to the second $index_2$ and inversely, see Figure 9.

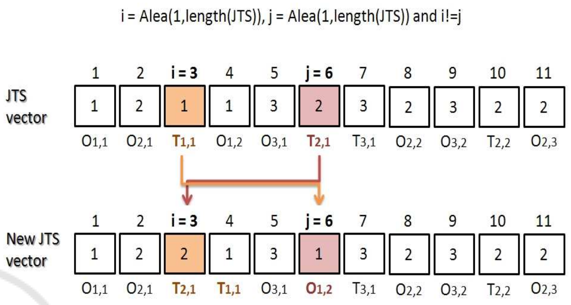

i = Alea(1,length(JTS)), j = Alea(1,length(JTS)) and i!=j
JTS vector
1 2 i=3 4 5 j=6 7 8 9 10 11
1 2 1 1 3 2 3 2 3 2 2
O1,1 O2,1 T1,1 O1,2 O3,1 T2,1 T3,1 O2,2 O3,2 T2,2 O2,3
New JTS vector
1 2 i=3 4 5 j=6 7 8 9 10 11
1 2 2 1 3 1 3 2 3 2 2
O1,1 O2,1 T2,1 T1,1 O3,1 O1,2 T3,1 O2,2 O3,2 T2,2 O2,3

Figure 9: JTS mutation exemple.

3.2.6 Replacement Operator

The replacement operator has an important role to prepare the remaining surviving population to be considered for the next iterations. This operator replaces in each case a parent by one of its children which has the best fitness in its current family.

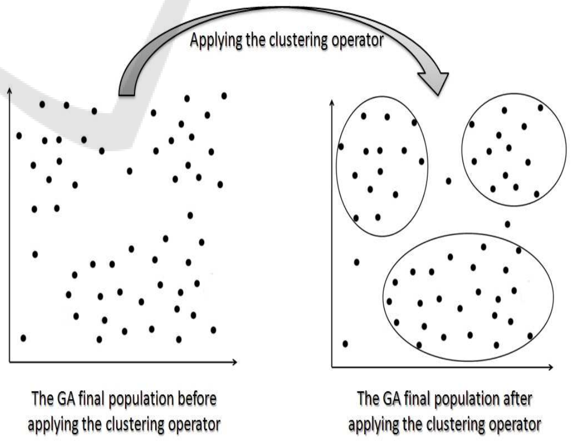

Applying the clustering operator
The GA final population before applying the clustering operator
The GA final population after applying the clustering operator

Figure 10: The final population transformation by applying the clustering operator.

3.2.7 Clustering Operator

By finishing the maximum iteration number *Max-Iter* of the genetic algorithm, the Scheduler Agent applies a clustering operator using the hierarchical clustering algorithm of (Johnson, 1967) to divide the final population into N Clusters, see Figure 10, to be treated by the Cluster Agents in the second step of the global process. The clustering operator is based on the neighbourhood parameter which is the dissimilarity distance between individuals. The clustering operator starts by assigning each individual $Indiv(i)$ to a cluster CL_i, so if we have P individuals, we have now P clusters containing just one individual in each of them. For each case, we fixe an individual $Indiv(i)$ and we verify successively for each next individual $Indiv(j)$ from the remaining population (where i and $j \in \{1,\ldots,P\}, i \neq j$) if the dissimilarity distance $Dist$ between $Indiv(i)$ and $Indiv(j)$ is less than or equal to a fixed threshold $Distfix$ (representing a percentage of difference X% relatively to $Distmax$, see Equation (10)) and where $Cluster(Indiv(i)) \neq Cluster(Indiv(j))$. If it is the case, $Merge(Cluster(Indiv(i)), Cluster(Indiv(j)))$, else continue the search for new combination with the remaining individuals. The stopping condition is by browsing all the population individuals, where we obtained at the end N Clusters.

$$Distfix = Distmax \times X\% \qquad (10)$$

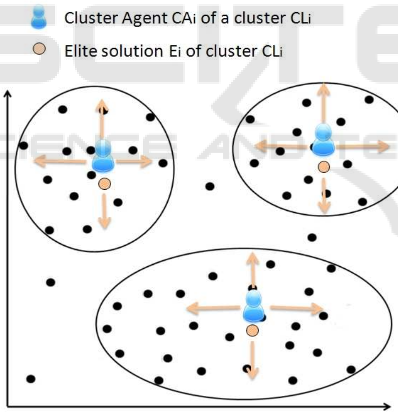

Cluster Agent CAi of a cluster CLi
Elite solution Ei of cluster CLi

Figure 11: Distribution of the Cluster Agents in the different clusters of the search space.

3.3 Cluster Agents

Each Cluster Agent CA_i is responsible to apply successively to each cluster CL_i a local search technique which is the Tabu Search algorithm to guide the research in promising regions of the search space and to improve the quality of the final population of the genetic algorithm. In fact, this local search is executed simultaneously by the set of the CAs agents, where each CA starts the research from an elite solution of its cluster searching to attain new more dominant individual solutions separately in its assigned cluster CL_i, see Figure 11. The used Tabu Search algorithm is based on an intensification technique allowing to start the research from an elite solution in a cluster CL_i (a promising part in the search space) in order to collect new scheduling sequence minimizing the makespan. Let E the elite solution of a cluster CL_i, $E' \in N(E)$ is a neighbor of the elite solution E, GL_i is the Global List of each CA_i to receive new found elite solutions by the remaining CAs, each CL_i plays the role of the tabu list with a dynamic length and Cmax is the makespan of the obtained solution. So, the search process of this local search starts from an elite solution E using the move and insert method of (Mastrolilli and Gambardella, 2000), where each Cluster Agent CA_i changes the execution order of an operation ($O_{i,j}$ if machine operation or $T_{i,j}$ if transport operation) from an index i to another index k in the vector JTS, searching to generate new scheduling combination $E' \in N(E)$. After that, verifying if the makespan value of this new generated solution $Cmax(E')$ dom-

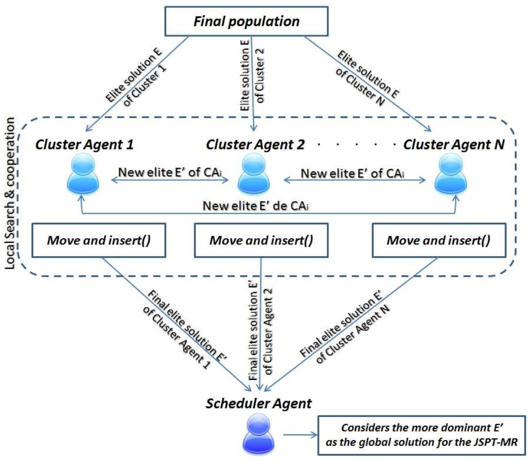

Final population
Elite solution E of Cluster 1
Elite solution E of Cluster 2
Elite solution E of Cluster N
Local Search & cooperation
Cluster Agent 1
Cluster Agent 2
Cluster Agent N
New elite E' of CAi
New elite E' of CAi
New elite E' de CAi
Move and insert()
Move and insert()
Move and insert()
Final elite solution E' of Cluster Agent 1
Final elite solution E' of Cluster Agent 2
Final elite solution E' of Cluster Agent N
Scheduler Agent
Considers the more dominant E' as the global solution for the JSPT-MR

Figure 12: Second step of the global process by the Cluster Agents.

Algorithm 2: The local search process.

```
1:  procedure
2:      $E \leftarrow$ Elite($CL_i$)
3:      while N($E$) $\neq \emptyset$ do
4:          $E' \leftarrow$ Move-and-insert($E$) | $E' \in$ N($E$) | $E' \notin CL_i$
5:          if Cmax($E'$) < Cmax($E$) and $E' \notin GL_i$ then
6:              $E \leftarrow E'$
7:              $CL_i \leftarrow E'$
8:              Send-to-all($E'$, $CA_i$)
9:          end if
10:     end while
11:     return $E$
12: end procedure
```

inates $Cmax(E)$ ($Cmax(E') < Cmax(E)$), and if it is the case CA_i saves E' in its tabu list (which is CL_i) and sends it to all the other CAs agents to be placed in their Global Lists $GLs(E', CA_i)$, to ensure that it will not be used again by them as a search point. Else continues the neighborhood search from the current solution E. The stopping condition is by attaining the maximum allowed number of neighbors for a solution E without improvement, see Figure 12. We give here how to proceed in Algorithm 2 :

By finishing this local search step, the CAs agents terminate the process by sending their last best solutions to the SA agent, which considers the best one of them as the global solution for the JSPT-SR.

4 EXPERIMENTAL RESULTS

4.1 Experimental Setup

The proposed GATS+HM is implemented in java language on a 2.10 GHz Intel Core 2 Duo processor and 3 Gb of RAM memory, where we use the Integrated Development Environment (IDE) *Eclipse* to code the algorithm and the multiagent platform *Jade* (Bellifemine et al., 1999) to create the different agents of our holonic model. To evaluate its efficiency, numerical tests are made based on the benchmark instances of (Hurink and Knust, 2005) from the literature of the JSPT-SR, which consists of two sets P1(6×6) and P2(10×10) inspired from (Muth and Thompson, 1963). In both instances the number of operations per job is equal to the number of machines and each job is processed on each machine exactly once with a fixed processing time. This shop problem considers a single moving robot for all transport operations, where various test instances were obtained by adding transportation and empty moving times with different characteristics.

Due to the non-deterministic nature of the proposed approach, we run it ten independent times for each case of the (Hurink and Knust, 2005) benchmark instances in order to obtain significant results. The used parameter settings for our algorithm are adjusted experimentally and presented as follow: the crossover probability = 1.0, the mutation probability = 0.5, the maximum number of iterations = 1000 and the population size = 200. The computational results are presented by three metrics in Table 2, such as the best makespan, the CPU time of our GATS+HM in minutes and the gap between our approach and the best results in the literature of the JSPT-SR, which is calculated by Equation (11). The *Mk***o** is the makespan obtained by **O**ur approach and *Mk***c** is the makespan of one of the chosen algorithms for **C**omparisons.

$$Gap = [(Mko - Mkc)/Mkc] \times 100\% \qquad (11)$$

4.2 Experimental Comparisons

To show the efficiency of our GATS+HM approach, we compare its obtained results from the (Hurink and Knust, 2005) benchmark instances with other algorithms from the literature of the JSPT-SR, which have obtained the best upper bounds for this problem. The chosen algorithms are : the one-stage approach (UB_{one}) of (Hurink and Knust, 2005) (with their known lower bound LB) which obtained the first results in the literature for their proposed instances, the genetic algorithm-tabu search procedure (GATS) of (Zhang et al., 2012) and the hybrid memetic algorithm (BFS) of (Lacomme et al., 2013) which are two recent hybrid metaheuristic approaches.

From Table 2, the comparison results show that the GATS+HM obtains twelve out of fifteen best results for the (Hurink and Knust, 2005) instances, where we attain ten new upper bounds and two similar results. Indeed, our algorithm outperforms the UB_{one} in eleven out of fifteen instances with a maximum gap of -7,80% for the P02-T5-t2 instance, and it gets slightly worse result for three instances with a maximum gap of 1,21% for the P02-D3-d1 instance. Moreover, our GATS+HM outperforms the BFS in twelve out of fifteen instances with a maximum gap of -4,13% for the P02-T2-t1 instance, and it gets one bad result for the P01-D2-d1 instance with a gap of 0,68%. For the comparison with the GATS, the GATS+HM obtains fourteen out of fifteen best results with a maximum gap of -30,52% for the P02-D2-d1 instance.

By analyzing the computational time in few minutes and the comparison results of our approach in terms of makespan, we can distinguish the efficiency of the new proposed GATS+HM relatively to the literature of the JSPT-SR. This efficiency is explained by the flexible selection of the promising parts of the search space by the clustering operator after the genetic algorithm process and by applying the intensification technique of the tabu search allowing to start from a set of elite solutions to attain new more dominant solutions.

5 CONCLUSION

In this paper, we present a new metaheuristic hybridization approach based on clustered holonic multiagent model, called GATS+HM, for the job

Table 2: Results of (Hurink and Knust, 2005) data instances.

Instance	LB	Gap_{LB}	UB_{one}	$Gap_{UB_{one}}$	BFS	Gap_{BFS}	GATS	Gap_{GATS}	GATS+HM	CPU Time
P01.D1-d1	**82**	2,44	87	-3,45	87	-3,45	96	-12,50	**84**	0,30
P01.D1-t1	**77**	2,60	81	-2,47	81	-2,47	83	-4,82	**79**	0,27
P01.D2-d1	**147**	1,36	**148**	0,68	**148**	0,68	155	-3,87	149	0,29
P01.D3-d1	**213**	0,00	217	-1,84	**213**	0,00	220	-3,18	**213**	0,28
P01.T2-t1	**71**	1,41	74	-2,70	74	-2,70	79	-8,86	**72**	0,26
P01.T3-t0	**92**	0,00	**92**	0,00	**92**	0,00	**92**	0,00	**92**	0,19
P02.D1-d1	**880**	10,57	1044	-6,80	1012	-3,85	1339	-27,33	**973**	6,15
P02.D1-t0	**880**	12,50	1042	-4,99	1017	-2,65	1352	-26,78	**990**	4,26
P02.D1-t1	**880**	11,36	1016	-3,54	983	-0,31	1337	-26,70	**980**	4,56
P02.D2-d1	**892**	12,56	1070	-6,17	1045	-3,92	1445	-30,52	**1004**	8,53
P02.D3-d1	**906**	19,54	**1070**	1,21	1100	-1,55	1516	-28,56	1083	11,45
P02.D5-t2	**1167**	13,71	**1325**	0,15	1361	-2,50	1689	-21,43	1327	13,15
P02.T1-t1	**874**	8,35	1006	-5,86	978	-3,17	1322	-28,37	**947**	7,14
P02.T2-t1	**880**	8,18	1015	-6,21	993	-4,13	1279	-25,57	**952**	7,31
P02.T5-t2	**898**	13,14	1102	-7,80	1022	-0,59	1339	-24,12	**1016**	9,25

shop scheduling problem with tansportation times and a single robot (JSPT-SR). In this approach, a neighborhood-based genetic algorithm is adapted by a scheduler agent for a global exploration of the search space. Then, a local search technique is applied by a set of cluster agents to guide the research in promising regions of the search space and to improve the quality of the final population. To measure its performance, numerical tests are made using benchmark data instances from the literature of JSPT-SR, and where new upper bounds are found showing the effectiveness of the presented approach. In the future work, we will search to treat other extensions of the JSPT-SR, such as the general case of this problem where a set of robots can be used for the transport oprations, and by considering the machine assignment problem for each operation in the shop process. So, the problem becomes a Flexible Job Shop scheduling Problem with Tansportation times and Many Robots (FJSPT-MR). Thus, we will make improvements in the chromosome first part MRS to adapt it to this new transformation and study its effects on the makespan.

REFERENCES

Abdelmaguid, T. F., Nassef, A. O., Kamal, B. A., and Hassan, M. F. (2004). A hybrid ga/heuristic approach to the simultaneous scheduling of machines and automated guided vehicles. *International Journal of Production Research*, 42(2):267–281.

Anwar, M. F. and Nagi, R. (1998). Integrated scheduling of material handling and manufacturing activities for just-in-time production of complex assemblies. *International Journal of Production Research*, 36(3):653–681.

Babu, A. G., Jerald, J., Haq, A. N., Luxmi, V. M., and Vigneswaralu, T. P. (2010). Scheduling of machines and automated guided vehicles in fms using differential evolution. *International Journal of Production Research*, 48(16):4683–4699.

Bellifemine, F., Poggi, A., and Rimassa, G. (1999). Jade - a fipa-compliant agent framework. In *In Proceedings of the fourth International Conference and Exhibition on The Practical Application of Intelligent Agents and Multi-Agent Technology*, pages 97–108.

Bilge, U. and Ulusoy, G. (1995). A time window approach to simultaneous scheduling of machines and material handling system in an fms. *Operations Research*, 43(6):1058–1070.

Botti, V. and Giret, A. (2008). *ANEMONA: A Multi-agent Methodology for Holonic Manufacturing Systems*. Springer Series in Advanced Manufacturing. Springer-Verlag.

Bozejko, W., Uchronski, M., and Wodecki, M. (2010). The new golf neighborhood for the flexible job shop problem. In *In Proceedings of the International Conference on Computational Science*, pages 289–296.

Braga, R. A. M., Rossetti, R. J. F., Reis, L. P., and Oliveira, E. C. (2008). Applying multi-agent systems to simulate dynamic control in flexible manufacturing scenarios. In *European Meeting on Cybernetics and Systems Research*, volume 2, pages 488–493. Austrian Society for Cybernetic Studies.

Calabrese, M. (2011). *Hierarchical-granularity holonic modelling*. Doctoral thesis, Universita degli Studi di Milano, Milano, Italy.

Caumond, A., Lacomme, P., Moukrim, A., and Tchernev, N. (2009). An milp for scheduling problems in an fms with one vehicle. *European Journal of Operational Research*, 199(3):706–722.

Deroussi, L., Gourgand, M., and Tchernev, N. (2008). A simple metaheuristic approach to the simultaneous scheduling of machines and automated guided vehicles. *International Journal of Production Research*, 46(8):2143–2164.

Deroussi, L. and Norre, S. (2010). Simultaneous scheduling of machines and vehicles for the flexible job shop problem. In *International Conference on Metaheuristics and Nature Inspired Computing*, pages 1–2.

Erol, R., Sahin, C., Baykasoglu, A., and Kaplanoglu, V. (2012). A multi-agent based approach to dynamic scheduling of machines and automated guided vehicles in manufacturing systems. *Applied Soft Computing*, 12(6):1720–1732.

Ferber, J. (1999). *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition.

Giret, A. and Botti, V. (2004). Holons and agents. *Journal of Intelligent Manufacturing*, 15(5):645–659.

Glover, F., Kelly, J. P., and Laguna, M. (1995). Genetic algorithms and tabu search: Hybrids for optimization. *Computers and Operations Research*, 22(1):111–134.

Hurink, J. and Knust, S. (2002). A tabu search algorithm for scheduling a single robot in a job-shop environment. *Discrete Applied Mathematics*, 119(1-2):181–203.

Hurink, J. and Knust, S. (2005). Tabu search algorithms for job-shop problems with a single transport robot. *European Journal of Operational Research*, 162(1):99–111.

Johnson, S. C. (1967). Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254.

Jones, A. and Rabelo, L. C. (1998). Survey of job shop scheduling techniques. Tech. rep., National Institute of Standards and Technology, Gaithersburg, USA.

Koestler, A. (1967). *The Ghost in the Machine*. Hutchinson, London, United Kingdom, 1st edition.

Komma, V. R., Jain, P. K., and Mehta, N. K. (2011). An approach for agent modeling in manufacturing on jade reactive architecture. *The International Journal of Advanced Manufacturing Technology*, 52(9-12):1079–1090.

Lacomme, P., Larabi, M., and Tchernev, N. (2007). A disjunctive graph for the job-shop with several robots. In *Multidisciplinary International Conference on Scheduling : Theory and Applications*, MISTA, pages 285–292, Paris, France.

Lacomme, P., Larabi, M., and Tchernev, N. (2013). Job-shop based framework for simultaneous scheduling of machines and automated guided vehicles. *International Journal of Production Economics*, 143(1):24–34.

Lee, K., Yamakawa, T., and Lee, K. M. (1998). A genetic algorithm for general machine scheduling problems. In *In Proceedings of the second IEEE international Conference on Knowledge-Based Intelligent Electronic Systems*, pages 60–66.

Lenstra, J. K. and Kan, A. H. G. R. (1979). Computational complexity of scheduling under precedence constraints. *Annals of Discrete Mathematics*, 4:121–140.

Lenstra, J. K. and Kan, A. H. G. R. (1981). Complexity of vehicle routing and scheduling problems. *Networks*, 11(2):221–227.

Mastrolilli, M. and Gambardella, L. (2000). Effective neighbourhood functions for the flexible job shop problem. *Journal of Scheduling*, 3(1):3–20.

Muth, J. F. and Thompson, G. L. (1963). *Industrial Scheduling*. International series in management. Prentice-Hall.

Pundit, R. and Palekar, U. (1990). Job shop scheduling with explicit material handling considerations. Technical report, Univ. of Illinois at Urbana-Champaign, Dept. of M. and I.E.

Raman, N., Talbot, F. B., and Rachamadgu, R. V. (1986). Simultaneous scheduling of machines and material handling devices in automated manufacturing. In *In Proceedings of the 2nd ORSA/TIMS Conference on Flexible Manufacturing Systems*, pages 455–466.

Reddy, B. S. P. and Rao, C. S. P. (2006). A hybrid multi-objective ga for simultaneous scheduling of machines and agvs in fms. *The International Journal of Advanced Manufacturing Technology*, 31(5-6):602–613.

Sonmez, A. I. and Baykasoglu, A. (1998). A new dynamic programming formulation of (nm) flow shop sequencing problems with due dates. *International Journal of Production Research*, 36(8):2269–2283.

Storn, R. and Price, K. (1995). Differential evolution: A simple and efficient adaptive scheme for global optimization over continuous spaces. Technical report, International Computer Science Institute, Berkeley.

Ulusoy, G., Erifolu, F. S., and Bilge, U. (1997). A genetic algorithm approach to the simultaneous scheduling of machines and automated guided vehicles. *Computers and Operations Research*, 24(3):335–351.

Zhang, Q., Manier, H., and Manier, M. A. (2012). A genetic algorithm with tabu search procedure for flexible job shop scheduling with transportation constraints and bounded processing times. *Computers and Operations Research*, 39(7):1713–1723.

Zhang, Q., Manier, H., and Manier, M. A. (2014). A modified shifting bottleneck heuristic and disjunctive graph for job shop scheduling problems with transportation constraints. *International Journal of Production Research*, 52(4):985–1002.