International Scholarly Research Network
ISRN Applied Mathematics
Volume 2011, Article ID 165620, 19 pages
doi:10.5402/2011/165620

Research Article

Ant Colony Algorithm for Just-in-Time Job Shop Scheduling with Transportation Times and Multirobots

Fatima El Khoukhi,[1,2] Tarik Lamoudan,[1,2] Jaouad Boukachour,[1] and Ahmed El Hilali Alaoui[2]

[1] *CERENE, ISEL, Quai Frissard, BP 1137, 76063 Le Havre Cedex, France*
[2] *Modelling Laboratory and Scientific Computing, Faculty of Science and Technology, BP 2202, Route D'Imouzzer Fez, Morocco*

Correspondence should be addressed to Ahmed El Hilali Alaoui, elhilali_fstf2002@yahoo.fr

Received 27 March 2011; Accepted 27 April 2011

Academic Editor: S. G. Garcia

Copyright © 2011 Fatima El Khoukhi et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Handling rapidly evolving technology and almost daily changes in demand and customer satisfaction, while maintaining competitiveness in a highly competitive environment, requires good coordination and planning of both production and logistics activities on the shop floor, namely: machines and tools. The goal is to optimize costs and reduce delivery lead times in order to provide the customer just in time; we focus on the job shop scheduling problem (JSSP), which is one of the most complex problems encountered in real shop floor. In this paper, we study a generalized (JSSP) including transportation times and a set of additional constraints on the number of transporter vehicles and their multiple transfer capabilities and also on the limited capacity of input/output of machines. The objective is to minimize in one hand tardiness and earliness penalties on delays and advances compared to the lead-time delivery of finished jobs and on the other hand the number of empty moves of transporter vehicles.

1. Introduction

Scheduling may be defined as the allocation of resources to tasks overtime to optimize a criteria. From the view point of production scheduling, the resources and tasks are commonly referred to a machines and jobs and the commonly used criteria is the completion times of jobs (makespan). In this paper, we study a variant of JSSP where the jobs have to be transported between the machines by one or several transporter vehicles. At JSSP, each job has its own processing order through the machines (a sequence of operations). Each operation must be processed on a fixed machine on which it has to be processed without preemption for a given

duration. Each machine can process just one operation at each time and each job can be performed on one machine at each time. Additionally, transportation times are considered.

The JSSP is known to be strongly NP-hard and exists in several variants according to the additional constraints considered such as, the classical problem [1], the flexible one [2], the cyclic problem [3], the dynamic job shop scheduling [4], the stochastic case [5], the reentrant problem [6], the case with separable setup time [7], the blocking job shop [8].

The JSSP, classified NP-complete, has attracted many researchers, [1, 7, 9, 10]. Including the transportation times between different machines, gives rise to a number of variants of the classical JSSP, particularly those concerned with job shops and transportation times. According to [11], there are two kinds of problems (1) the first one occurs in production manufacturing, when we need to transport jobs between machines and (2) the second one generally occurs at the level of delivery of products to customers. In many industries, production and delivery systems are integrated, with finished products being transferred from a manufacturing execution service to a customer delivery service. In this case, the most common form of delivery to the customer is by vehicle, with the makespan being given by the last delivery date.

In both types, the commonly used objective is to find a scheduling solution which minimizes the makespan. Related works that examine the JSSP with the transportation times (either type 1 or 2), include the works of [12–17]. Additionally, Yuan et al. [18] studied the complexity of flow shop problem with transportation times, in order to minimize the makespan. Brucker et al. [19] considered the job shop scheduling with limited capacity buffers. Finally, Caumond et al. [20] give a linear formulation taking into account the maximum number of jobs allowed in the system, limited input/output buffer capacities, empty trips and no-move-ahead trips simultaneously.

In this paper, our contribution presents one generalization version of the JSSP with transportation times, integrating the different additional constraints mentioned previously, including the existence of several transporter vehicles with multiple transfer capacity in the shop. Moreover, our model incorporates other constraints related to storage buffers associated with each machine taking into account the case of limited buffer spaces. In our case, we study specially the transportation activities, the empty or not. To achieve a just in time (JIT) production [21], the goal is to minimize the earliness and tardiness penalties with regard to the delivery deadline of finished products, as well as penalties on empty activities.

The remainder of the paper is organized as follows. In Section 2, we give a formal definition of the considered scheduling problem and state some additional assumptions. Section 3 deals with the mathematical modeling of the problem and Section 4 presents our method of resolution. The computational results can be found in Section 5. Finally, a conclusion is given in Section 6.

2. Problem Context

In this work, we study the JSSP with transportation times taking into account the following constraints:

(i) JIT scheduling in order to respect the delivery times (latest completion time) imposed by customers,

(ii) a set of homogenous transporter vehicles,

(iii) a transporter vehicle has a finite capacity of transfer more than one (in terms of a number of tasks they can carry),

(iv) a station = {input + machine + output} (see Figure 1),

(v) a station input/output buffers has a limited capacity, not necessary the same for both storage spaces,

(vi) two deposits, the initial one for arrival jobs and the final one for finished jobs.

We show below a mathematical formulation of the problem, with all basics constraints of the classical JSSP, and additional transport constraints concerning the transporter vehicles, their transfer capacity and the limited capacity of storage buffers associated with each machine.

3. Mathematical Formulation

3.1. Classical Data

(i) m: number of machines,

(ii) $M = \{M_1, M_2, \ldots, M_m\}$: set of machines,

(iii) n: number of jobs,

(iv) $J = \{J_1, J_2, \ldots, J_n\}$: set of jobs,

(v) $O_{i\sigma_{ik}}$: the kth task of the job i,

(vi) σ_{ik}: the machine required by the task $O_{i\sigma_{ik}}$,

(vii) $J_i = \{O_{i\sigma_{i\,1}}, O_{i\sigma_{i\,2}}, \ldots, O_{i\sigma_{in_i}}\}$: technological Sequence associated with the job i, with n_i, the number of operations of the job i,

(viii) $P_i^{\sigma_{ik}}$: processing time of the operation $O_{i\sigma_{ik}}$,

(ix) $r_i^{\sigma_{ik}}$: earliest starting time of the operation $O_{i\sigma_{ik}}$,

(x) $d_i^{\sigma_{ik}}$: due date of the operation $O_{i\sigma_{ik}}$,

(xi) α_i^k: earliness Penalty of the task with regard to its delivery deadline,

(xii) β_i^k: tardiness Penalty of the task with regard to its delivery deadline.

3.2. Classical Decision Variables

(i) C_i^U: actual finish date of job i on machine U, with $C_i^{\sigma_{ik}}$ the date of last operation $O_{i\sigma_{ik}}$ on the required machine σ_{ik},

(ii)

$$\lambda_i^k = \begin{cases} 1, & \text{if } C_i^{\sigma_{ik}} \leq d_i^{\sigma_{ik}}, \\ 0, & \text{otherwise}, \end{cases} \tag{3.1}$$

The previous equation is the binary variable to record the earliness or lateness associated with delivery delay of operation $O_{i\sigma_{ik}}$.

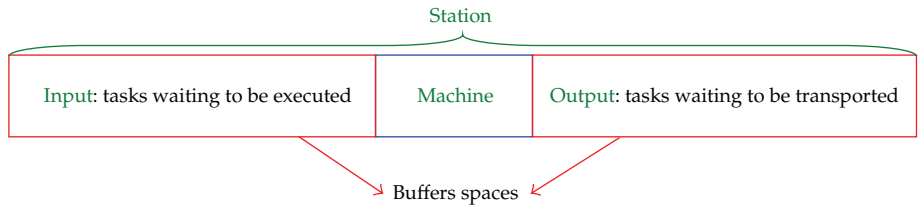

Figure 1: Station model used in this paper.

3.3. Classical Constraints

The first constraint ensures that the first operations of the jobs must be processed respecting their corresponding earliest starting times

$$C_i^{\sigma_{i1}} - P_i^{\sigma_{i1}} \geq r_i^{\sigma_{i1}}, \quad \forall i = 1, \ldots, n. \tag{3.2}$$

The second constraint guarantees the respects of the precedence sequences constraints predefined between the tasks of the same job

$$C_i^{\sigma_{ik}} \leq C_i^{\sigma_{ik+1}} - P_i^{\sigma_{ik+1}}, \quad \forall i = 1, \ldots, n, \ \forall k = 1, \ldots, (n_i - 1). \tag{3.3}$$

The disjunction at the level of machines is specified in the third constraint

$$C_{E_{ij}}^{U} - P_{E_{ij}}^{U} \geq C_{i+j-E_{ij}}^{U}, \quad \forall i < j \ \in O_U, \ \forall U \in M, \tag{3.4}$$

with

$$E_{ij} = \begin{cases} j, & \text{if job } i \text{ precedes } j, \\ i, & \text{otherwise,} \end{cases} \tag{3.5}$$

$$O_u = \{\text{set of jobs requiring the machine } U\}.$$

Remark 3.1. Our constraint is an improvement of the constraint of disjunction appearing in the literature (see [22]) taking the following general form:

$$C_j^U - C_i^U + H(1 - a_{ijU}) \geq P_j^U, \quad \forall i, j \ = 1, \ldots, n, \ \forall U \in M, \tag{3bis}$$

with

$$a_{ijU} = \begin{cases} 1, & \text{if job } i \text{ precedes } j \text{ on the machine } U, \\ 0, & \text{otherwise,} \end{cases} \tag{3.6}$$

$$H : \text{a big value.}$$

3.4. Additional Data, Variables, and Constraints

In manufacturing production, a transporter vehicle undertaken to perform any or all of the following activities: loading the input of a station, unloading the output of a station and/or removing a task from one a station to another.

In general, where the transporter vehicle can carry only one task at a time, two cases are presented:

(i) if the transporter vehicle moves from a source machine to a destination one in order to load it, then a transportation time is considered, it is a transport activity,

(ii) if the transporter vehicle moves from a source machine to a destination one without loading it, an empty travel time is considered; it is an empty movement activity.

Here, we consider the problems that the transporter vehicles can carry more than one task at a time and for each machine there are two associated storage buffers input/output, where tasks can wait for their execution or transportation. In the remainder of this paper, we introduce the following definitions:

(1) a transporter vehicle performs a "transport activity" if it moves in a direct way from a station to another one to load it,

(2) a transporter vehicle performs an "empty movement activity" (even it is charged), if it moves in a direct way from a source station to a destination one without loading it.

3.4.1. Additional Data

(1) R: number of transporter vehicles in the workshop (assuming more than one),

(2) $S = \{S_1, S_2, \ldots, S_m\}$: set of stations,

(3) C_r: capacity of the transporter vehicle r (in terms of numbers of tasks to carry),

(4) ε_c: loading time of one or several tasks on the input of a station,

(5) ε_d: unloading time of one or several tasks from the output of a station,

(6) Cout (U): output Capacity of the machine U,

(7) Cin (U): input Capacity of the machine U,

(8) T: big value representing an upper bound for the production horizon,

(9) δ_{sd}: transportation time from a station s to a station d,

(10) δ'_{sd}: empty movement activity from a station s to a station d.

3.4.2. Additional Decision Variables

We use additional decision variables to express the objective function and additional constraints on the transport of tasks between machines, and the activities of transporter vehicles

$$t_{i\sigma_{ik}}\text{: starting transportation time of task } O_{i\sigma_{ik}},$$

$$x^r_{i\sigma_{ik}} = \begin{cases} 1, & \text{if the task } O_{i\sigma_{ik}} \text{ is transported by the transporter vehicle } r, \\ 0, & \text{otherwise,} \end{cases}$$

$$S^t_{i\sigma_{ik}} = \begin{cases} 1, & \text{if the task } O_{i\sigma_{ik}} \text{ is transported in time } t, \\ 0, & \text{otherwise,} \end{cases} \tag{3.7}$$

$$V^a_r = \begin{cases} 1, & \text{if the transporter vehicle is in "empty movement activity,"} \\ 0, & \text{if the transporter vehicle is in "transport activity."} \end{cases}$$

3.4.3. Additional Constraints

A task can be handled by only one transporter vehicle at a time

$$\sum_{r=1}^{R} x^r_{i\sigma_{ik}} = 1, \quad \forall i = 1, \ldots, n, \ \forall k = 1, \ldots, n_i. \tag{3.8}$$

Each job requires $n_i + 1$ tasks of transport

$$\sum_{r=1}^{R} \sum_{k=0}^{n_i+1} x^r_{i\sigma_{ik}} = n_i + 1, \quad \forall i = 1, \ldots, n, \tag{3.9}$$

σ_{i0} and $\sigma_{i(n_i+1)}$ are two fictitious machines representing the initial and the final deposit of the jobs in the workshop.

A task can be performed only after its transport

$$C_i^{\sigma_{ik+1}} - P_i^{\sigma_{ik+1}} \geq t_{i\sigma_{ik}} + \delta_{S(\sigma_{ik})\, S(\sigma_{ik+1})} + \varepsilon_d, \quad \forall i = 1, \ldots, n, \ \forall k = 1, \ldots, n_i, \tag{3.10}$$

where $S(\sigma_{ik})$ is the station associated with the machine σ_{ik}

Each task can be transported only after its completion

$$t_{i\sigma_{ik}} \geq C_i^{\sigma_{ik}} + \varepsilon_c, \quad \forall i = 1, \ldots, n, \ \forall k = 1, \ldots, n_i. \tag{3.11}$$

The current capacity of the transporter vehicle r at time t (noted C_r^t) must not exceed its capacity

$$C_r^t = \sum_{i=1}^{n} \sum_{k=1}^{n_i} S_{\sigma_{ik}}^t x_{i\sigma_{ik}}^r \leq C_r, \ \forall r = 1, \ldots, R, \ \forall t = 0, \ldots, T. \tag{3.12}$$

The load of a transporter vehicle at each time t must not exceed its current available capacity $(C_r - C_r^t)$,

$$\sum_{i=1}^{n} \sum_{k=1}^{n_i} \text{out}_{\sigma_{ik}}^t x^r{}_{i\sigma_{ik}} \leq C_r - C_r^t, \quad \forall r = 1, .., R, \ \forall t = 0, \ldots, T \tag{3.13}$$

such as

$$\text{out}_{\sigma_{ik}}^t = \begin{cases} 1, & \text{if } C_i^{\sigma_{ik}} \leq t < t_{i\sigma_{i\,k+1}}, \\ 0, & \text{otherwise.} \end{cases} \tag{3.14}$$

The limited capacity of the outputs of machines is respected at every time t

$$\sum_{i=1}^{n} \sum_{k=\,1/\sigma_{ik}=\,u}^{n_i} \text{out}_{\sigma_{ik}}^t \leq \ C_{\text{out}}(u), \quad \forall u \in M, \ \forall t = 0, \ldots, T. \tag{3.15}$$

The limited capacity of the inputs of machines is respected at every time t

$$C_{in}^t(u) = \sum_{i=1}^{n} \sum_{k=\frac{1}{\sigma_{ik}}=u}^{n_i} \text{int}_{\sigma_{ik}}^t \leq \ C_{in}(u) \quad \forall u \in M \ \forall t = 0, \ldots, T, \tag{3.16}$$

with

$$\text{int}_{\sigma_{ik}}^t = \begin{cases} 1, & \text{if } t_{i\sigma_{ik}} + \delta_{S(\sigma_{i(k-1)})S(\sigma_{ik})} + \varepsilon_d \leq t < C_i^{\sigma_{ik}} - P_i^{\sigma ik}, \\ 0, & \text{otherwise,} \end{cases} \tag{3.17}$$

where $C_{in}^t(u)$ is the current capacity of the input associated with the machine u.

At every moment, the transporter vehicles should serve only the available inputs of the machines

$$\sum_{r=1}^{R} \sum_{i=1}^{n} \sum_{k=1/\sigma_{ik}=u}^{n_i} S_{i\sigma_{ik}}^t x_{i\sigma_{ik}}^r \leq C_{in}(u) - C_{in}^t(u), \quad \forall u \in M, \ \forall t = 0, \ldots, T. \tag{3.18}$$

The following recurrent formula computes the completion time of service of transporter vehicles on every machine which they serve, noted by F_r^s,

$$F_r^s = F_r^{s-1} + v_a^r\left(\delta'_{(s-1)s} + \varepsilon_c\right) + (1 - v_a^r)\left(\delta_{(s-1)s} + \varepsilon_d + f_r^s\varepsilon_c\right), \quad \forall r = 1, \ldots, R,\ \forall s \in S, \tag{3.19}$$

with

$$f_r^s = \begin{cases} 1, & \text{if the transporter vehicle unloads the station } s, \\ 0, & \text{otherwise.} \end{cases} \tag{3.20}$$

Finally, we express the separation time restrictions due to the type of transporter vehicle activities, "transport activity" (transportation time δ_{sd}) or "empty movement activity" (time of empty travel time δ'_{sd})

$$\begin{aligned} v_a^r\left(\ F_r^s + \delta'_{sd} - F_r^d\right) + (\ 1 - v_a^r)\left(\ F_r^s + \delta_{sd} + \varepsilon_d - F_r^d\right) \le 0, \\ \forall r = 1, \ldots, R,\ \forall s, d \in S,\ \forall a \in \text{Activities}. \end{aligned} \tag{3.21}$$

3.5. Objective Function

The objective function seeks to minimize the sum of earliness and tardiness penalties with regard to the delivery deadlines of the finished jobs, as well as the sum of the penalties on the empty movement activities of the transporter vehicles

$$\text{Min} \sum_{i=1}^{n} \sum_{k=1}^{n_i} \left(-\alpha_i^k\right)^{\lambda_i^k} \left(\beta_i^k\right)^{(1-\lambda_i^k)} \left(c_i^{\sigma_{ik}} - d_i^{\sigma_{ik}}\right) + \sum_{r=1}^{R} \sum_{a \in \text{Activities}} m_a^r v_a^r \tag{3.22}$$

such as

α_i^k: earliness penalty on the task $O_{i\sigma_{ik}}$,

β_i^k: tardiness penalty on the task $O_{i\sigma_{ik}}$,

m_a^r: penalty on an "empty movement activity".

4. Ant Colony Algorithm

According to our previous work [23–25], we found that ant colony algorithm is better than genetic algorithm in terms of solutions even sometimes with more execution time.

For the resolution of the generalized version of the job shop with transportation times studied in this paper, we propose a resolution algorithm based on ant colony optimization that we denote by "ACOJST" (ant colony optimization for job shop with transportation times), in order to generate the starting times of operations, as well as their starting transportation times.

Ants move on the problem graph (see Figure 2). Each node is associated with a station S_k which is composed of the machine M_k and its input I_k and output O_k. A dotted arc

represents a connection between the deposits, respectively, initial and final towards the various stations. A continuous arc connecting two stations represents the transfer path for the transporter vehicles. Finally, a quantity of pheromone $\tau_{S_k S_h}$ is associated with each arc of the proposed graph; this quantity simulates the density of transfer between the stations.

Our algorithm contains two main stages, the initial deposit stage and the production one. The first stage initializes transporter vehicles by the first operations of jobs according to a heuristic that we developed. The heuristic takes into account the number of transporter vehicles, their capacity and the total capacity of the initial deposit (in terms of the number of operations to be transported). Then, the loaded transporter vehicles are directed to stations, according to a priority rule, priority is giving to the destination station for which each transporter vehicle will carry a maximum block of tasks in the minimum time. Moreover, to manage the transporter vehicles movements inside of the shop, we use a heuristic to avoid conflict that may be caused by the arrival of more than one transporter vehicle to the same station at the same time.

During the production stage, we choose a task from the inputs buffer to be executed whenever the station is free. Managing the communication with every transporter vehicle with an input to load or an output to unload, and the need to continually choose the next station to be served are all the decisions that we take by appropriate rules. During the cycle of production, there are two steps.

(1) Associate with the input of every station, a fictitious immovable ant called "ant machine", is free to choose the task $O_{i\sigma_{ik}}$ to be executed whenever its required machine σ_{ik} is available. The choice of this task is based on a priority rule favoring the task with the maximum ratio $\mathrm{rap}_{ik}(t)$ in σ_{ik}:

$$O_{i\sigma_{ik}} = \underset{\substack{O_{j\sigma_{jh}} \in \mathrm{Input}^t_{\sigma_{ik}} \\ (j=1,\ldots,n, h=1,\ldots,n_j)}}{\arg\max} \left\{ \mathrm{rap}_{jh}(t) \right\}, \tag{4.1}$$

where

$$\mathrm{rap}_{ik}(t) = \frac{p_i^{\sigma_{ik}}}{\sum_{h=1}^{n_i} p_i^{\sigma_{ih}}}, \quad \forall O_{i\sigma_{ik}} \in \mathrm{Input}^t_r, \tag{4.2}$$

$\mathrm{Input}^t_{\sigma_{ik}} = \{\text{tasks belonging in input of machine } \sigma_{ik} \text{ at the moment } t\}$.

(2) In addition, create a family of mobile ants called "ants' robots", which help transporter vehicles to choose the next station to be served by distinguishing two scenarios.

(i) If the transporter vehicle is carrying one or several finished jobs, then facilitating the movement towards the final deposit.

(ii) If it does not carry any finished job then, the choice of the next station is made according to a transition rule making a compromise between the fact of facilitating the least loaded input and the most loaded output (to guarantee that the transporter vehicle serves at most, the chosen station). The rule, that we introduced, distinguishes between two scenarios, the quantity transported by the transporter vehicle overtake or not a threshold.

(a) If the capacity of a transporter vehicle exceeds this threshold, we select stations corresponding to the tasks transported by the chariot of this transporter

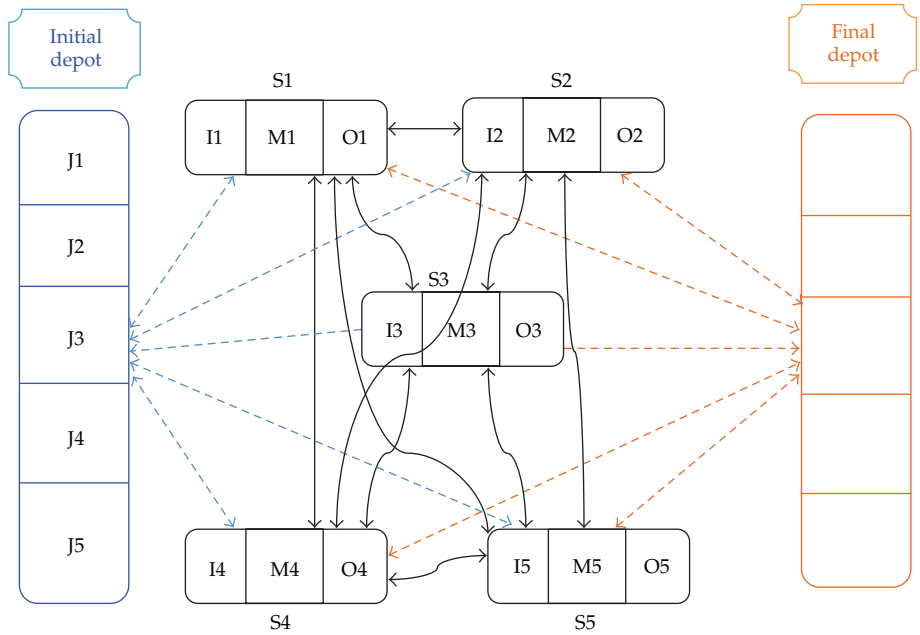

Figure 2: Problem graph.

vehicle, in which case the station to be served is chosen according to the following transition rule:

$$d(t) = \underset{s \in \text{Chario}(r)}{\arg\max} \, (\tau_{hs})^{\alpha} q_1(\beta,\gamma) q_2(\theta,\psi) q_3. \tag{4.3}$$

(b) If the capacity of the studied transporter vehicle does not exceed this threshold, all the stations of the workshop are considered, in this case the station to be served is chosen according to the following transition rules:

$$d(t) = \underset{s \in S}{\arg\max} (\tau_{hs})^{\alpha} q_1(\beta,\gamma) q_2(\theta,\psi) q_3, \tag{4.4}$$

with:

$$\begin{aligned} q_1(\beta,\gamma) &= \left(1 - \frac{1}{\left(\text{Cin}(s) - \text{Cin}^t(s)\right) + 1}\right)^{\beta} \left(\frac{1}{\left(\text{Cou}(s) - \text{Cou}^t(s)\right) + 1}\right)^{\gamma}, \\ q_2(\theta,\psi) &= \left(\frac{1}{\delta_{cs} + 1}\right)^{\theta} \left(\frac{1}{\delta'_{cs} + 1}\right)^{\psi}, \\ q_3 &= (card\_bloc(r,s))(tmachine(s)) \left(\frac{1}{Fs(s)}\right). \end{aligned} \tag{4.5}$$

$card\_bloc(r,s)$: number of tasks transported by the transporter vehicle r towards the station s $tmachine(s)$: Total duration of activities of the station s during the horizon of production α, β, γ, θ, and ψ: These parameters express the compromise between the rate of pheromone, load of the input (less loaded) the load of the output (more loaded, see saturated), the transportation times and the empty movements (minimal).

The goal is to reduce the empty travel and to minimize the transportation times while taking into account the quantity of pheromone, in order to minimize the makespan.

(3) Pheromone can be update locally or globally:

(i) The local update is performed after every transfer of an "ant robot" from a source station c to a destination one d, according to the following formula:

$$\tau_{cd}(k+1) = \tau_{cd}(k) + \tau_0 \left(1 - \rho_0\right). \tag{4.6}$$

(ii) The global update is performed at the level of the best solution obtained after each cycle of the algorithm, according to the following formula:

$$\begin{aligned} &\tau_{m(i)m(i+1)}(k+1) = \tau_{m(i)m(i+1)}(k) + \tau_0 \left(1 - \rho_0\right) \\ &\left(\text{for each pair of successive operation of the best solution}\right), \end{aligned} \tag{4.7}$$

with:

(a) $m(i)$: request machine of the task number i,

(b) $\tau_{s_1 s_2}(k)$: quantity of pheromone in the arc connecting the two stations $s1$ and $s2$, in the cycle k of the algorithm,

(c) $\tau_{s_1 s_2}(k+1)$: quantity of pheromone in the arc connecting the two stations $s1$ and $s2$, in the current iteration, after update,

(d) τ_0: initial quantity of pheromone,

(e) ρ_0: rate of evaporation.

4.1. Algorithm Description

In Algorithm 1, we summarize the different steps of our ant colony algorithm with:

(i) $m(i)$: the request machine of the task i,

(ii) $initial\_deposit$: initial deposit of the workshop,

(iii) $Chario(r)$: set of tasks transported by the transporter vehicle number r,

(iv) $c(r,t)$: current position,

(v) $d(r,t)$: destination position of the transporter vehicle number r at the moment t,

(vi) $N1\,\max$ and $N2\,\max$: stop tests,

(vii) τ_0: initial quantity of pheromone,

initialization step
$Depot\_initial \leftarrow \{first\ operations\ of\ jobs\}$
$Chario(r) \leftarrow O_r, \forall r = 1, \ldots, R\ (O_r \subset initial\_deposit)$
$d(r,0) \leftarrow deplacer\_robot(r), \forall r = 1, \ldots, R$ // *heuristic to move the transporter vehicle*
$c(r,0) = d(r,0), \forall r = 1, \ldots, R$
For $(it_1 = 1, \ldots, N1\max)$ *do*
  For $(it = 1, \ldots, N2\max)$ *do*
    // *generate one solution*
    For $(t = 1, \ldots, horizon)$ *do*
      // *cycle of production*
      For $(s = 1, \ldots, S)$ *do*
        $Traiter\_station(s)$ // *treatment of stations*
      End For
      For $(r = 1, \ldots, R)$ *do*
        $d(r,t) \leftarrow traiter\_robot(r)$ // *treatment of transporter vehicles*
        $c(r,t) \leftarrow d(r,t)$
        // *local update of pheromone*
        $\tau_{cd} = \tau_{cd} + \tau_0\ (1 - \rho_0)$
        End For
        For $(s = 1, \ldots, S)$ *do*
          $Mise\_ajour\_station(s)$ // *update of stations*
        End For
  End For
End For
$Sol \leftarrow choisir\_meilleure()$ // *chose of the best solution*
// *Global update of pheromone.*
$\tau_{m(i)m(i+1)} = \tau_{m(i)m(i+1)} + \tau_0\ (1 - \rho_0)$, *for each pair of successive operation of sol*
End For
End algorithm

Algorithm 1

(viii) ρ_0: rate of evaporation,

(ix) τ_{cd}: quantity of the pheromone in the arc connecting the two stations c and d,

(x) S: number of stations,

(xi) R: number of robots.

5. Computational Results

We validate our approach against two types of benchmarks from literature:

(i) 1st type instances: those of Philippe Lacomme, downloadable from the following web page: http://www.isima.fr/~lacomme/Job_Shop_TL.html, for the case of a single transporter vehicle with a capacity of transition equal to 1,

(ii) 2nd type instances: we change benchmarks from instances of type 1 to have several transporter vehicles, by changing the number of transporter vehicles, their capacity of transfer and the capacity of the spaces of input/output storages.

Table 1: Case of a single transporter vehicle.

Instances	n	m	LB	UB	Lacomme solution	ACOJST	DEV (%)
LT133	3	5	178	190	178	192	7.8
LT144	4	5	214	224	214	233	8, 8
LT155	5	5	248	272	254	271	9.2
LT233	3	5	148	148	148	148	0
LT244	4	5	160	164	160	167	4.3
LT255	5	5	169	175	169	170	0.5
LT333	3	5	199	211	199	209	5
LT344	4	5	205	216	205	213	3.9
LT355	5	5	210	227	210	218	3.8
LT433	3	5	271	324	271	350	29.1
LT444	4	5	353	418	353	431	22
LT455	5	5	426	520	431	501	17.6
LT533	3	5	365	474	365	516	41.3
LT544	4	5	477	612	477	629	31.8
LT555	5	5	585	650	585	731	24.9

DEV: deviation of the solution given by ACOJST from LB.

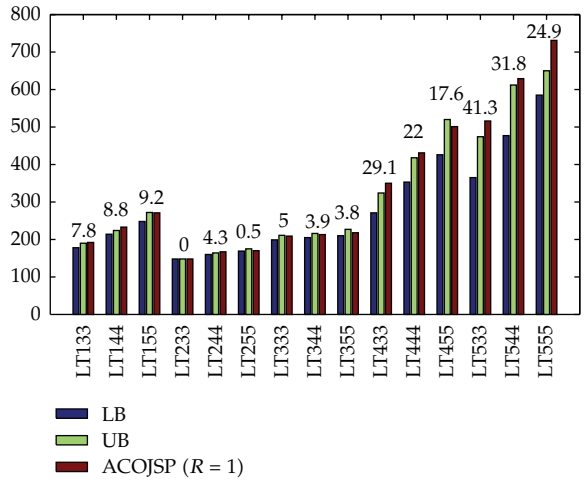

Figure 3: Case of a single transporter vehicle.

Tests were performed on a Pentium (R) Dual-Core CPU E5200 @ 2, 50 GHz, 1 Go RAM. The number of jobs is between three and five and the number of machines was fixed at four, to which we add two fictitious machines represent the initial and final deposits.

The results below are obtained with the capacity of the transporter vehicle between 1 and 2 and the capacity of the input and the output buffer is fixed to 2. The LB and UB present, respectively, the best known solutions (BKS) obtained by the branch-and-bound procedure coupled with a discrete event simulation model (called B&B/simulation framework) [15] and by the procedure B&B [20], that extends the B&B/simulations.

Table 2: The results of this comparison.

Instances	LB	UB	ACOJST (input management heuristic)	FIFO	LIFO	LPT	SPT	Argmin
LT133	178	190	192	193	192	192	192	193
LT144	214	224	233	233	233	233	233	233
LT155	248	272	271	271	271	271	271	271
LT233	148	148	148	148	148	148	148	152
LT244	160	164	167	167	164	164	167	164
LT255	169	175	170	183	182	187	184	183
LT333	199	211	209	213	211	214	214	212
LT344	205	216	213	219	219	219	219	219
LT355	210	227	218	228	228	228	229	228
LT433	271	324	350	350	350	350	351	350
LT444	353	418	431	431	431	431	431	431
LT455	426	520	501	501	501	501	501	501
LT533	365	474	516	516	516	516	516	516
LT544	477	612	629	629	629	629	629	629
LT555	585	650	731	731	731	731	731	731

Table 3: Case of a set of transporter vehicles.

Instances	n	m	LB	Lacomme solution ($R = 1$)	ACOJSP ($R = 1$)	ACOJSP ($R = 1$)	ACOJSP ($R = 4$)
LT133	3	5	178	178	130	117	117
LT144	4	5	214	214	140	129	132
LT155	5	5	248	254	147	131	125
LT233	3	5	148	148	144	145	145
LT244	4	5	160	160	142	145	145
LT255	5	5	169	169	149	146	146
LT333	3	5	199	199	200	200	212
LT344	4	5	205	205	205	205	205
LT355	5	5	210	210	204	204	204
LT433	3	5	271	271	203	158	144
LT444	4	5	353	353	250	172	159
LT455	5	5	426	431	264	198	173
LT533	3	5	365	365	293	230	198
LT544	4	5	477	477	363	242	231
LT555	5	5	585	585	374	285	223

R: the number of transporter vehicles.

5.1. Transporter Vehicles with a Capacity Equal to 1

5.1.1. Use of a Single Transporter Vehicle

For the case of single transporter vehicle, the results obtained by ACOJST algorithm (see Table 1 and Figure 3) show a small deviation (≤10%) for the first instances (LT133–LT355), for example, for the instance LT233 and LT255, our solutions are identical or nearly identical to the corresponding LB bounds. On the other hand, the deviations increase (>10%) for the

Table 4: (a) Case of 3 transporter vehicles with capacity equal to 2. (b) Case of 3 transporter vehicles with capacity equal to 3.

(a)

Instances	n	M	ACOJSP ($R = 3$, Cap = 1)	ACOJSP ($R = 3$, Cap = 2)	Gap (%)
LT133	3	5	117	126	7.7
LT144	4	5	129	135	4.7
LT155	5	5	131	134	2.3
LT233	3	5	145	151	4.1
LT244	4	5	145	148	2.1
LT255	5	5	146	148	1.4
LT333	3	5	200	203	1.5
LT344	4	5	205	205	0.0
LT355	5	5	204	205	0.5
LT433	3	5	158	185	17.1
LT444	4	5	172	189	9.9
LT455	5	5	198	177	−10.6
LT533	3	5	230	263	14.3
LT544	4	5	242	257	6.2
LT555	5	5	285	264	−7.4

R: the number of transporter vehicles.
Cap: the capacity of transfer of a transporter vehicle.
Gap = $[(B2 - B1)/B1] * 100$.
$B1$: The solution found by ACOJSP in the case of 3 transporter vehicles, capacity 1.
$B2$: The solution found by ACOJSP in the case of 3 transporter vehicles, capacity 2.

(b)

Instances	n	m	ACOJSP ($R = 3$, Cap = 1)	ACOJSP ($R = 3$, Cap = 3)	Gap (%)
LT13335	3	5	117	126	7.7
LT14445	4	5	129	129	0.0
LT15555	5	5	131	157	19.8
LT23335	3	5	145	153	5.5
LT24445	4	5	145	148	2.1
LT25555	5	5	146	143	−2.1
LT33335	3	5	200	200	0.0
LT34445	4	5	205	205	0.0
LT35555	5	5	204	203	−0.5
LT43335	3	5	158	189	19.6
LT44445	4	5	172	194	12.8
LT45555	5	5	198	223	12.6
LT53335	3	5	230	252	9.6
LT54445	4	5	242	292	20.7
LT55555	5	5	285	268	−6.0

Gap = $[(B2 - B1)/B1] * 100$.
$B1$: The solution found by ACOJSP in the case of 3 transporter vehicles, capacity 1.
$B2$: The solution found by ACOJSP in the case of 3 transporter vehicles, capacity 3.

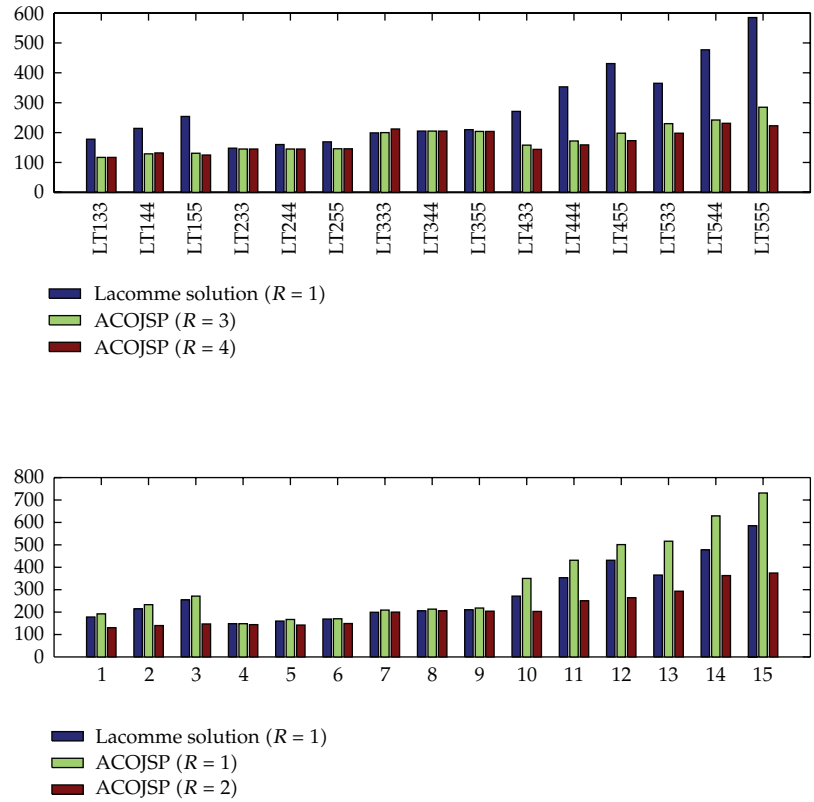

Figure 4: Study of the influence of the number of transporter vehicle in a shop with transport of a set of transporter vehicles.

large instances (LT433–LT555). This difference is justified by the fact that our algorithm is dedicated to several transporter vehicles.

To validate our input management heuristic for the case of single transporter vehicle, we compared it with the proposed Argmin heuristic and four priority rules FIFO, LIFO, SPT, and LPT. The input management heuristic gives the best solutions in most cases.

Table 2 shows the results of this comparison.

5.1.2. Case of Several Transporter Vehicles

In this case, we use only the input management heuristic.

By comparing the results obtained by ACOJST in the case of multiple transporter vehicles with the LB given for the case of a single transporter vehicle (see Table 3 and Figure 4), we note that the numerical results show that our algorithm is powerful in the case of several transporter vehicles and especially for instances of great dimension such as LT444, LT555.

5.2. Transporter Vehicles with a Capacity Equal to 2 and 3

The Figures 5(a) and 5(b), associated with Tables 4(a) and 4(b), show the behavior of our approach according to the capacity of transfer of the transporter vehicle. Our study is done in

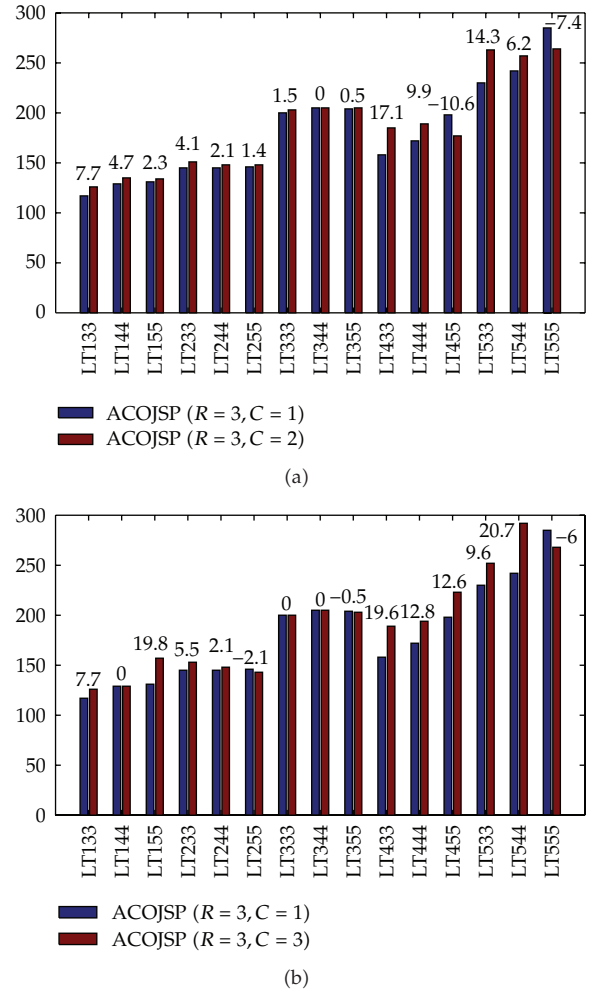

Figure 5: (a) Study the influence of transfer capacity in the case of a 3 transporter vehicles with 2 capacities. (b) Study the influence of transfer capacity in the case of a 3 transporter vehicles with 3 capacities.

the case of 3 transporter vehicles with 2 and 3 capacities. The results are less good than those obtained using 3 transporter vehicles with capacity 1 and this is due to the constraint of conflict of transporter vehicles, taking into account in the implementation.

6. Conclusion

In this work, we studied the general JSSP including temporal constraints related to the transportation of tasks between machines, the capacity of the transporter vehicle, and also their number and the spaces of storage input/output. In the first part of this work, we give a mathematical formulation of the problem and in the second part, we tried to adapt an ant colony algorithm with several heuristics. The work may be extended further by taking into account other real constraints like the management of the topology of stations in the shop and studying the dynamic case.

References

[1] A. S. Jain and S. Meeran, "Deterministic job-shop scheduling: past, present and future," *European Journal of Operational Research*, vol. 113, no. 2, pp. 390–434, 1999.

[2] W. Xia and Z. Wu, "An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problem," *Computers & Industrial Engineering*, vol. 48, pp. 409–425, 2005.

[3] G. Cavory, R. Dupas, and G. Goncalves, "A genetic approach to solving the problem of cyclic job shop scheduling with linear constraints," *European Journal of Operational Research*, vol. 161, no. 1, pp. 73–85, 2005.

[4] M. A. Adibi, M. Zandieh, and M. Amiri, "Multi-objective scheduling of dynamic job shop using variable neighborhood search," *Expert Systems with Applications*, vol. 37, no. 1, pp. 282–287, 2010.

[5] J. Gu, M. Gu, C. Cao, and X. Gu, "A novel competitive co-evolutionary quantum genetic algorithm for stochastic job shop scheduling problem," *Computers & Operations Research*, vol. 37, no. 5, pp. 927–937, 2010.

[6] J. Chao-Hsien Pan and J.-S. Chen, "Mixed binary integer programming formulations for the reentrant job shop scheduling problem," *Computers & Operations Research*, vol. 32, no. 5, pp. 1197–1212, 2005.

[7] A. Rossi and G. Dini, "Flexible job-shop scheduling with routing flexibility and separable setup times using ant colony optimization method," *Robotics and Computer-Integrated Manufacturing*, vol. 23, no. 5, pp. 503–516, 2007.

[8] H. Gröflin and A. Klinkert, "A new neighborhood and tabu search for the blocking job shop," *Discrete Applied Mathematic*, vol. 157, no. 17, pp. 3643–3655, 2009.

[9] J. Boukachour and A. Benabdelhafid, "Résolution d'un probléme d'ordonnancement de type job-shop par les algorithmes génétiques ," in *Proceedings of the Troisiéme Conférence Internationale de Mathématiques Appliquées et des Sciences de l'Ingénieur*, CIMASI, Casablanca, Maroc, Octobre 2000.

[10] K.-L. Huang and C.-J. Liao, "Ant colony optimization combined with taboo search for the job shop scheduling problem," *Computers & Operations Research*, vol. 35, no. 4, pp. 1030–1046, 2008.

[11] A. Soukhal, A. Oulamara, and P. Martineau, "Complexity of flow shop scheduling problems with transportation constraints," *European Journal of Operational Research*, vol. 161, no. 1, pp. 32–41, 2005.

[12] C.-Y. Lee and Z.-L. Chen, "Machine scheduling with transportation considerations," *Journal of Scheduling*, vol. 4, no. 1, pp. 3–24, 2001.

[13] J. Hurink and S. Knust, "A tabu search algorithm for scheduling a single robot in a job-shop environment," *Discrete Applied Mathematics*, vol. 119, no. 1-2, pp. 181–203, 2002.

[14] J. Hurink and S. Knust, "Tabu search algorithms for job shop problems with a single transport robot," *European Journal of Operational Research*, vol. 162, pp. 99–111, 2005.

[15] P. Lacomme, A. Moukrim, and N. Tchernev, "Simultaneously job input sequencing and vehicle dispatching in a single vehicle AGVS: a heuristic branch and bound approach coupled with a discrete events simulation model," *International Journal of Production Research*, vol. 43, no. 9, pp. 1911–1942, 2005.

[16] G. El Khayat, A. Langevin, and D. Riopel, "Integrated production and material handling scheduling using mathematical programming and constraint programming," *European Journal of Operational Research*, vol. 175, no. 3, pp. 1818–1832, 2006.

[17] S. E. Kesen and Ö. F. Baykoç, "Simulation of automated guided vehicle (AGV) systems based on just-in-time (JIT) philosophy in a job shop environment," *Simulation Modelling Practice and Theory*, vol. 15, no. 3, pp. 272–284, 2007.

[18] J. Yuan, A. Soukhal, Y. Chen, and L. Lu, "A note on the complexity of flow shop scheduling with transportation constraints," *European Journal of Operational Research*, vol. 178, no. 3, pp. 918–925, 2007.

[19] P. Brucker, S. Heitmann, J. Hurink, and T. Nieberg, "Job-shop scheduling with limited capacity buffers," *OR Spectrum*, vol. 25, no. 2, pp. 151–176, 2006.

[20] A. Caumond, P. Lacomme, A. Moukrim, and N. Tchernev, "A MILP for scheduling problems in an FMS with one vehicle," *European Journal of Operational Research*, vol. 199, pp. 706–722, 2009.

[21] P. Baptiste, M. Flamini, and F. Sourd, "Lagrangian bounds for just-in-time job-shop scheduling," *Computers & Operations Research*, vol. 35, no. 3, pp. 906–915, 2008.

[22] R. Cheng, M. Gen, and Y. Tsujimura, "A tutorial survey of job-shop scheduling problems using genetic algorithms-I. Representation," *Computers & Industrial Engineering*, vol. 30, no. 40, pp. 983–997, 1996.

[23] R. Abounacer, J. Boukachour, B. Dkhissi, and A. El Hilali Alaoui, "A hybrid Ant Colony Algorithm for the exam timetabling problem," *Revue ARIMA*, vol. 12, pp. 15–42, 2010.

[24] G. Bencheikh, J. Boukachour, and A. El Hilali Alaoui, "Improved Ant Colony Algorithm to solve the aircraft landing problem," *International Journal of Computer Theory and Engineering*, vol. 3, no. 2, pp. 224–233, 2011.

[25] G. Bencheikh, J. Boukachour, A. El Hilali Alaoui, and F. El Khoukhi, "Hybrid method for aircraft landing scheduling based on a Job Shop formulation," *International Journal of Computer Science and Network Security*, vol. 9, no. 8, 2009.