Résumé : L’objectif de cet article est de proposer une procédure exacte pour optimiser l’ordonnancement des livraisons de lots d’une chaîne logistique dans un contexte de juste à temps. Ce problème est caractérisé principalement par le fait que les produits doivent être livrés chez le client avant leurs dates dues et aussi tard que possible. Quelques propriétés mathématiques intéressantes sont démontrées pour ce problème dans le cas où un seul transporteur livre les biens. Ces résultats particuliers sont ensuite utilisés pour accélérer de façon considérable la recherche de la solution optimale par une procédure appropriée de séparation évaluation progressive. Des résultats expérimentaux comparatifs illustrent l’efficacité de l’algorithme proposé.

Abstract: The purpose of this paper is to propose an exact procedure to optimize the lots delivery scheduling in a simple supply chain under just in time sitting. This problem is mainly characterized by the fact that the products have to be delivered to the customer before given due dates and as late as possible. Some interesting mathematical properties are given for this problem in case of there is a single transporter to deliver the goods. These results greatly improve the search of the optimal solution by an appropriate branch and bound procedure. Comparative experimental results illustrate the efficiency of the proposed algorithm.

Keywords: optimal load flows, optimization problem, scheduling algorithm, optimal control, synchronization, transportation control, tree search.

1. INTRODUCTION

The Supply Chain can be defined as a set of actors (suppliers, producers, distributors…) which contribute to manufacture, sale and provision of finished products for the customers. The logistic is the management of business operations, such as the acquisition, storage, transportation and delivery of goods along the supply chain. In today’s increasing global and competitive marketplace, it is imperative that members of supply chain work together in an effort to minimize overall times and costs of production and transportation (F.E. Vergara & al, 2002). Thus, the problem of optimization production and transport sequences was largely studied these last years. Indeed, the Economic Lot and delivery Scheduling Problem (ELDSP) was studied by (J. Hahm & C. A Yano, 1992) and an extension of the problem is introduced to hold account the capacity of the transporter. The purposed model was also used in (F.Elizabet Vergara et al., 2002). They generalized this model to a linear supply chain (customers and suppliers in cascade) and they developed a genetic algorithm to find the best scheduling sequence of various products flows through the production line. (M. Khouja, et al. 1998) also used the genetic algorithms to solve the sequences of production scheduling problem of several products in the same production line. In (M.A Hoque & S.K Goyal, 2000), a delivery policy of batches with equal and non equal sizes was developed under a constant demand. (S.Y Chou & S.L Chang, 2001) developed heuristics to find the minimal cost in the context of only one order of production and several deliveries. A dynamic programming was introduced by (K. H Kim & J. B Kim, 2002) in order to select a model for trucks transporting components between two members of logistic chain.

However, all of this research maintained the assumption of a deterministic and constant logistic demand.

In (I. Elmahi, et al., 2002), a max-plus algebra based model is proposed to control a simple supply chain under the assumption that the load sequence of the transporter is given. If the load sequence become an input variable of the model, all the possible load sequences have to be considered to find the optimal one.

The purpose of this paper is to propose an exact procedure to optimize the lots delivery scheduling in a simple supply chain under just in time sitting. The main contribution of this work is to demonstrate some interesting mathematical properties for this
problem in case of there is a single transporter to deliver the goods. These results greatly improve the search of the optimal solution by an appropriate branch and bound procedure.

This paper is organized as follow: In section 2, we present the studied system and the constraints related to its dynamic. The mathematical model of this system and its properties are provided in section 3. We propose in the following section an efficient Branch and Bound Procedure to find the optimal solution of the studied problem. Comparative experimental results show the efficiency of the proposed algorithm in section 5. Section 6 wraps up the paper with conclusions and perspectives.

2. PROBLEM STATEMENT

The system studied in this work is a simple supply chain made up of three actors: one supplier, one customer and one transporter (Fig. 1). The customer needs a number of products which must be available in its stock at given due dates, that is to say that the products can arrive eventually earlier compared with their due dates but never later. The products are manufactured by the supplier who has to respect the requirements of the customer. Once manufactured, they can be transported from the supplier toward the customer by the transporter. Several parameters characterize the transporter such as the capacity of loading, the travel duration between the customer and the supplier and the loading time and unloading time of each product from the transporter.

![Fig. 1. The studied system.](image)

The purpose of this study is to optimize the lots delivery scheduling in this chain under just in time sitting. This problem is mainly characterized by the fact that the products have to be delivered to the customer before given due dates and as late as possible. Under these assumptions, one has to find the optimal lot sequence that minimizes the global advance of the products, that is to say the sum of the differences between the due date and the real arrival date for all products.

The following section gives a mathematical model of this system and its properties.

3. MATHEMATICAL MODEL

The notations used in this paper are given as follows:

- \( np \): total number of the requested products
- \( c_{\text{max}} \): loading capacity of the vehicle.
- \( t_p \): travel duration from supplier to customer.
- \( t_u \): time duration to return back at the supplier.
- \( T = t_g + t_u \): transport time.
- \( t_l \): loading time per product.
- \( Y_d \): vector of due dates, where :
  \[ Y_d = (y_d(k), k = 1, \ldots np) \]
  \( y_d(k) \): the due date of the \( k^{th} \) product.
- \( Y_r \): vector of the real arrival dates, where:
  \[ Y_r = (y_r(k), k = 1, \ldots np) \]
  \( y_r(k) \): the real arrival date of the \( k^{th} \) product. If the \( k^{th} \) and \( k+1^{st} \) products are in the same lot, we have:
  \[ y_r(k+1) = y_r(k) + t_u \]
- \( s_p \): partial sequence for the last \( p \) products.
- \( A_g(s_p) \): the global advance of the partial sequence \( s_p \).
- \( A_\text{av}(s_p) \): the average advance of the partial sequence \( s_p \).

3.1 Definitions

Definition 1: We define a loading sequence \( s_{np} \) as a series \( (s_{np}^i)_{i=1}^{np} \) which satisfies:

a) \( \forall i \in [1, np] \), \( 0 \leq s_{np}^i \leq c_{\text{max}} \).

b) \( \exists i \in [1, np] \), if \( s_{np}^i = 0 \) then \( \forall j > i, s_{np}^j = 0 \).

c) \( \sum_{i=1}^{np} s_{np}^i = np \).

Definition 2: A partial sequence \( s_p \) is a sequence of \( p \) products such as \( p < np \).

The products of any partial sequence always correspond to the last ones required by the customer. Thus, the first product of the partial sequence \( s_p \) corresponds to the product \( np - p + 1 \) needed by the customer.

Definition 3: The global advance of a given partial sequence is defined as the sum of the differences between the due date and the real arrival date for all products.

\[
A_g(s_p) = \sum_{k=np-p+1}^{np} (y_d(k) - y_r(k))
\]

Definition 4: The average advance for a given partial sequence \( s_p \) is defined as:

\[
A_\text{av}(s_p) = \frac{A_g(s_p) + A_b}{p+1}
\]

with:

\[
A_b = \left[ y_d(np-p) - y_r(np-p+1) - T - t_u - s_{np}^1 t_l \right]
\]

The next section provides the expressions of the real arrival dates for each product and a proposition to compare partial sequences of this problem.

3.2 Model analysis
We first formulate the real arrival dates of the last lot of a partial sequence with the following lemma.

**Lemma 1**

Given a partial sequence $s_p$, the real arrival date of the first product of the last lot is:

$$y_r(np - s_p^k + 1) = \min_{i=1}^{l'} \left[ y_r(np - s_p^k + i) - (i-1) \cdot t_u \right]$$

**Proof:** The real arrival dates of the last lot have to satisfy:

\[
\begin{align*}
y_r(np - s_p^k + 1) &\leq y_r(np - s_p^k + 1) \\
y_r(np - s_p^k + 2) &\leq y_r(np - s_p^k + 2) \\
\vdots &\vdots \\
y_r(np - s_p^k + 1) + t_u &\leq y_r(np - s_p^k + 2) \\
y_r(np - s_p^k + 1) + (s_p^k - 1) \cdot t_u &\leq y_r(np)
\end{align*}
\]

Consequently, we have:

\[
\begin{align*}
y_r(np - s_p^k + 1) &\leq y_r(np - s_p^k + 1) \\
y_r(np - s_p^k + 2) &\leq y_r(np - s_p^k + 2) - t_u \\
\vdots &\vdots \\
y_r(np - s_p^k + 1) &\leq y_r(np - s_p^k + 2) - t_u \\
y_r(np - s_p^k + 1) - s_p^k \cdot t_u &\leq y_r(np)
\end{align*}
\]

The real arrival dates of other products in this lot are obtained by adding the unloading time $t_u$.

From this result, we give the expression of the real arrival dates of the previous lots in the following lemma:

**Lemma 2**

Given a partial sequence $s_p = \{s_p^1, s_p^2, \ldots, s_p^k\}$, the real arrival date for the $i^{th}$ product in $s_p^j$ ($j < k$) is:

\[
y_r(np - s_p^j + 1) = \min_{i=1}^{l'} \left[ y_r(np - s_p^j + i) - (i-1) \cdot t_u \right],
\]

with $\sigma = \sum_{j=1}^{k} s_p^j$

**Proof:** The real arrival dates in the previous lots have to satisfy the same inequality as in the lemma 1. Moreover, the following constraint has to be satisfied:

$$y_r(np - s_p^j + 1) - s_p^j \cdot t_u - T$$

Indeed, the two lots have to be separated with a necessary temporal interval which allows the transporter to go to the supplier, load the products and then return toward the customer.

Thus:

\[
y_r(np - \sigma + 1) \leq \min_{i=1}^{l'} \left[ y_r(np - \sigma + i) - (i-1) \cdot t_u \right]
\]

This second lemma uses a backward recursion to determine the real arrival date of the first product in the previous lots. Thus, using this lemma, all real arrival dates for all products in the previous lots are calculated. These dates are the latest possible for a given sequence; they thus allow to minimize the total advance of this sequence.

The following lemmas 3 and 4 are provided to compare two partial sequences of the problem. They will be used later in this paper to improve the efficiency of the branch and bound search.

**Lemma 3**

Given two partial sequences $s_p$ and $s'_p$ such as $y_r(np - p + 1 - s_p^i \cdot t_i) \geq y_r'(np - p + 1 - s'_p^i \cdot t_i)$, and a lot of $x$ products, we consider the following partial sequences:

$s_{+x+p} = \{x, s_p^1, s_p^2, \ldots, s_p^k\}$ and $s'_{+x+p} = \{x, s'_p^1, s'_p^2, \ldots, s'_p^k\}$

Let $y_r(i)$ and $y'_r(i)$ be the real arrival dates of the $i^{th}$ product in $s_{+x+p}$ and $s'_{+x+p}$ respectively, then

\[
y'_r(i) \geq y_r(i)
\]

**Proof:** Let formulate the real arrival dates of the first product for the two partial sequences:

\[
\begin{align*}
y_r(np - x + 1) &= \min_{i=1}^{l'} \left[ y_r(np - x + i) - (i-1) \cdot t_u \right] \\
y'_r(np - x + 1) &= \min_{i=1}^{l'} \left[ y'_r(np - x + i) - (i-1) \cdot t_u \right]
\end{align*}
\]

with $np' = np - p$.

Let $y_r(np - x + 1) = m$

\[
y_r'(np - x + 1) = m
\]

We have to consider 4 cases:

1) $y_r'(np - x + 1) = m$

\[
y'_r(np - x + 1) = m
\]

2) $y'_r(np - x + 1) = y_r'(np + 1) - T - s'_p^i \cdot t_u - x \cdot t_u$

So $y'_r(np - x + 1) < m = y_r(np - x + 1)$

\[
y'_r(np - x + 1) \geq y_r'(i)
\]

Thus:

\[
y_r(np - \sigma + 1) \leq \min_{i=1}^{l'} \left[ y_r(np - \sigma + i) - (i-1) \cdot t_u \right]
\]
3) \[ y_i, (np'^{-1}x + 1) = y_i, (np'^{-1}+1) - T - s^i\cdot t_i - x t_u \]

Thus:
\[ y_i, (np'^{-1}x + 1) \leq M \leq y_i, (np'^{-1}x + 1) - T - s^i\cdot t_i - x \cdot t_u \]

Then:
\[ y_i, (np'^{-1}x + 1) \leq y_i, (np'^{-1}x + 1) - s^i\cdot t_i \leq y_i, (np'^{-1}x + 1) - s^i\cdot t_i \]

This last inequality is contradictory with the assumptions of the lemma. This is impossible.

4) \[ y_i, (np'^{-1}x + 1) = y_i, (np'^{-1}+1) - T - s^i\cdot t_i - x t_u \]

We have \[ y_i, (np'^{-1}x + 1) \geq y_i, (np'^{-1}x + 1) - s^i\cdot t_i \]

Then \[ \forall i \in [np - p - x + 1, np - p], \ y_i, (i) \geq y_i, (i) \]

This lemma means that if two different partial sequences are completed by the same lot and the real arrival date of the first product in the first partial sequence is greater or equal to the real arrival date of the first product in the second partial sequence, then, the first partial sequence is a better solution than the second one.

**Definition 5:** We define \( \zeta(s_p) \) the set of solutions (complete sequences) constructed from the partial sequence \( s_p \) by adding in upstream \( s_p \) necessary lots to satisfy the customer request. We denote \( s'(s_p) \) the sequence belonging to \( \zeta(s_p) \) which global advance is the smallest.

\[ \forall s \in \zeta(s_p), \ A_g(s) \geq A_g(s'(s_p)) \]

**Proposition 1:**
Let \( s_p \) and \( s'_p \) be two partial sequences for the same number of products \( p \).
Assume that:
\[ \begin{align*}
A_g(s_p) & \leq A_g(s'_p) \\
y_i, (np - p + 1) - s^i_p \times t_i & \geq y_i, (np - p + 1) - s^i_p \times t_i
\end{align*} \]

then
\[ A_g(s'(s_p)) \leq A_g(s'(s'_p)) \]

**Proof:** We suppose that:
\[ A_g(s'(s_p)) > A_g(s'(s'_p)) \]

This means:
\[ \forall s \in \zeta(s_p), \ A_g(s) > A_g(s'(s'_p)) \]

The sequence \( s'(s'_p) \) is made up of two parts: the first one is the partial sequence \( s'_p \). The second one is another partial sequence which composed of lots \( sc_{np-p} \) necessary to satisfy the total request of the customer (fig 2). We note this last one \( sc_{np-p} \) with \( sc_{np-p} = \{SC_{np-p}^1, SC_{np-p}^2, \ldots, SC_{np-p}^d \} \).

![Fig. 2. loading sequence \( s'(s'_p) \).](image)

We complete the partial sequence \( s_p \) with the partial sequence \( sc_{np-p} \) as shown in figure 3. We note this obtained sequence \( r_{np} \).

![Fig. 3. Completion of partial sequences.](image)

The lemma 3 applied to \( r_{np} \) and \( s'(s'_p) \) gives:
\[ \forall i \in [1, np - p], \ y_i, (i) \geq y_i, (i) \]

Then:
\[ y_i, (i) \geq y_i, (i) \]

\[ \sum_{i=1}^{np-p} (y_i, (i) - y_i, (i)) \leq \sum_{i=1}^{np-p} (y_i, (i) - y_i, (i)) \]

\[ \leq \sum_{i=1}^{np-p} (y_i, (i) - y_i, (i)) + [A_n(s_p) - A_n(s'(s'_p))] \]

\[ = A_n(r_{np}) < A_n(s'(s'_p)) \]

The last inequality means that we find one sequence belonging to \( \zeta(s_p) \) whose the global advance is smallest than \( A_n(s'(s'_p)) \). Hence, the assumption of \( A_n(s'(s'_p)) > A_n(s'(s'_p)) \) was false. Consequently, we have:
\[ A_n(s'(s_p)) \leq A_n(s'(s'_p)) \]

The mathematical model given above allows to compute the latest real arrival dates for a given sequence. If the load sequence become an input variable of the model, all the possible load sequences have to be considered to find the optimal one. As the size of problem increases exponentially with the number of products, it becomes impossible to explore all search space. We propose in the following section, a Branch and Bound procedure based on the above results which allows, in addition, to find the optimal sequence in very reducing time.

**4. BRANCH AND BOUND PROCEDURE**

The BBP belongs to the exact resolution methods which thus guarantee the optimality of the found solution. The BBP has techniques to detect the failures as soon as possible and thus obtaining a considerable gain in computing time. It allows to reduce the search space by an implicit enumeration of complete branches of tree search (Hao J.K., et al., 1999). The tops of the tree search correspond to subsets which are divided into levels. There is only one top with level zero which represents the root of the aborescence. A minimal evaluation of the branch
(called limit) is needed to construct the tops of the other levels. Otherwise say, all the other tops which obtained from the current node will have an evaluation greater or, at best, equal to this limit.

The BBP is based on two elementary techniques which act considerably on the quality and the speed of the algorithm: The first one is the evaluation of the solution during its construction. The second one is the separation which consists in stopping the construction of a solution as soon as the evaluation of this last one become greater than the limit evaluation.

The loading sequences optimization problem can be modelled in the shape of tree search. Each node models the quantity of products loaded in a given lot. With each node, we associate an evaluation corresponding to the advance of the partial sequence of products delivered until this top. The algorithm has to be able to stop as soon as possible the construction of a branch which will end inevitably to a bad solution.

The proposition 1 is used by the developed BBP to explore the search arborescence and find the optimal loading sequence. In following section, we explain how the BBP proceeds to choose the best lot among the various possibilities in the same level.

4.1 Branching scheme

The search arborescence holds at most \( np \) levels corresponding to the maximal number of trips necessary to satisfy the customer request. In each level, all the nodes that correspond to the various possibilities of loading remaining products, are considered. Thus, the first level of tree contains \( n_1 \) nodes where \( n_1 = \min (np \cdot ch_{\text{max}}) \). These nodes correspond to the last lot in the sequence.

In the next levels, the number of nodes depends of the path from the root node to the parent node of the previous level. So on, in each level, the number of choices is calculated using:

\[
n_j = \min(np_j \cdot ch_{\text{max}}) \quad \text{with} \quad np_j = np - \sum_{i=j+1}^{k} \delta_{np}^i
\]

For the last lot, we have to select among a set of nodes the best one. Thus, the corresponding average advance is calculated for each loading. This advance is computed using definition 4. Hence, the selected loading is the one with the smallest average advance.

The following bounding scheme is used by the BBP to stop the exploration of a given branch. It is based essentially on the proposition 1 given above.

4.2 Bounding scheme

If the choice of the last lot is equal to \( \delta_{np}^k \), the best partial sequence to deliver the last \( \delta_{np}^k \) products is to group them in the same lot. Indeed, to calculate the average advance for a given partial sequence, we need to calculate the global advance for this partial sequence and then, the real arrival date for product preceding the sequence. In proposition 1, we have proved that for a same number of products, the partial sequence with the smallest global advance and greatest real arrival date will give inevitably the optimal solution. Thus, it is not necessary to explore the other possibilities for the same number of products. Otherwise say, this partial sequence is the best manner to deliver that number of products.

Once the last loading is chosen, we obtain with the same manner the previous loading. The different average advances for the partial sequences \( \delta_{np}^{k-1} \) are computed where \( \delta_{np}^{k-1} \) is the different possibilities of loading remaining products \( \delta_{np}^{k-1} \in [1, \min(np - \delta_{np}^k \cdot ch_{\text{max}})] \). And so on, all the other previous lots are obtained.

During the exploration of tree search, the best partial sequence of each number of products is registered. The best global advance and the latest real arrival dates are compared with the global advance and real arrival date of the current branch. If they not satisfy the two conditions of the proposition 1, the exploration of this branch will stop immediately.

Otherwise, the current branch becomes the new best partial sequence. So on, we obtain the best loading sequence for all products.

To explain this scheme, the following example is given:

Let to deliver 8 products (\( np = 8 \)) from the supplier to the customer. The vehicle capacity is \( ch_{\text{max}} = 5 \) and the vector of due dates is:

\[ Y = \{100, 101, 102, 120, 12, 13, 131, 132\} \]

The temporal parameters are:

\[ t_g = t_e = 5 \Rightarrow T = 10 \quad , \quad t_i = t_y = 1 \]

We start by finding the last lot. So, the supplier can deliver 1, 2, 3, 4 or 5 products.

For 1 product, the real arrival date is \( y_{1n}(8) = 132 \), then, the average advance is:

\[
A_n(1) = \frac{Ag(1) + [y_{1n}(7) - (y_{1n}(8) - T - t_u - t_i)]}{2} = \frac{0 + [131 - (132 - 10 - 1)]}{2} = 5.5
\]

Thus, in the same manner, we obtain the following table 1:

<table>
<thead>
<tr>
<th>Products number</th>
<th>Real arrival dates</th>
<th>Global advance</th>
<th>Average advance</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>( y_2(7) = 131 )</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>( y_2(8) = 132 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>( y_3(6) = 130 )</td>
<td>0</td>
<td>1.5</td>
</tr>
<tr>
<td></td>
<td>( y_3(7) = 131 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>( y_3(8) = 132 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>( y_4(5) = 122 )</td>
<td>21</td>
<td>6.8</td>
</tr>
<tr>
<td></td>
<td>( y_4(6) = 123 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>( y_4(7) = 124 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>( y_4(8) = 125 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>( y_5(4) = 120 )</td>
<td>25</td>
<td>3.8</td>
</tr>
<tr>
<td></td>
<td>( y_5(5) = 121 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>( y_5(6) = 122 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>( y_5(7) = 123 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>( y_5(8) = 124 )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

We conclude that the best manner to deliver the last three products is to take them on the same lot. Then the last trip is equal to 3.
In the previous trip, we repeat the same process to calculate the number of loaded products. The selected partial sequence is 2–3 and the average advance corresponding is \( A_{23} = 1.66 \).

So on, we find finally that the best sequence to deliver the 8 products is 3–2–3. The global advance for this sequence is \( A_g = 3(2-3) = 11 \). The following figure 4 resumes the different leafs explored by the BBP and the corresponding average advance for each one.

![Fig. 4. Explored nodes for the example.](image)

We see that we didn’t explore all possibilities in each level. Indeed, the average advance for each leaf allows algorithm to choose the way to follow during the exploration of tree search. The proposition 1 makes it possible to stop the construction of a given branch as soon as its evaluations end inevitably to a bad solution.

5. EXPERIMENTAL RESULTS

To test the performance of the developed BBP, we have considered several problems with different sizes and different parameters. For each problem size, 10 problem instances with different data \((c_{\text{max}}, T, \ldots)\) are generated. The minimum, the maximum and the average computation times of the branch and bound algorithm are reported in the table 2, as well as the computation time needed for an enumeration procedure.

### Table 2 Average run time for enumeration procedure and BBP.

<table>
<thead>
<tr>
<th>Products number</th>
<th>BPP</th>
<th>Run time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Min</td>
<td>Max</td>
</tr>
<tr>
<td>20</td>
<td>15 ms</td>
<td>16 ms</td>
</tr>
<tr>
<td>30</td>
<td>15 ms</td>
<td>16 ms</td>
</tr>
<tr>
<td>40</td>
<td>15 ms</td>
<td>32 ms</td>
</tr>
<tr>
<td>50</td>
<td>15 ms</td>
<td>47 ms</td>
</tr>
<tr>
<td>80</td>
<td>31 ms</td>
<td>157 ms</td>
</tr>
<tr>
<td>100</td>
<td>47 ms</td>
<td>1.15 sec</td>
</tr>
<tr>
<td>200</td>
<td>187 ms</td>
<td>46.4 sec</td>
</tr>
<tr>
<td>300</td>
<td>2.12 sec</td>
<td>1.2min</td>
</tr>
</tbody>
</table>

Note that these results are obtained with a computer which has a P4 processor with a frequency equal to 2.66GHz. In addition, it is interesting to note the difference between running time of the BBP as opposed to the enumeration procedure. As the table shows, as the problems become larger, the BBP quickly becomes much faster than the enumeration procedure. For example, for 40 products, the BBP took on average 20 ms versus more than one day for the enumeration procedure. However, we note that the running time for a given number of products differs of a problem to another. It is explaining by the fact that data \((c_{\text{max}}, T, \ldots)\) differ of a problem to another.

This BBP is also applied to problem with large size in order to find the limit of the procedure. We find that problems of 1000 products can be resolved easily by this BBP. The running time to find the optimal solution is very tolerable. That time which borders 48mn 11sec is roughly the same one as that carried out by heuristics like the "genetic algorithms" to find an approached solution.

6. CONCLUSION

In this paper, we have proposed a very efficient optimization procedure for the lots delivery scheduling problem in a simple supply chain under a just in time criterion. This problem is mainly characterized by the fact that the products have to be delivered to the customer before given due dates and as late as possible. Under these assumptions, one has to find the optimal lot sequence that minimizes the global advance of the products. Some interesting mathematical properties are given for this problem which act considerably on the speed of the algorithm. Indeed, these results applied to a branch and bound procedure, allow to find the optimal solution of this problem in a very reducing computing time. We have provided some comparative experimental results which illustrate the efficiency of the proposed algorithm. Indeed, this BBP open the field to ensure the exactitude of the found solution of problems with large size.

However, this study has to be generalized to take in account supply chains with several suppliers and customers and large fleet of transporters. The transport, the loading, the unloading and the holding costs has also to be integrated in the evaluation function.

REFERENCES


ACKNOWLEDGMENTS

A first version of this paper was published at IFAC Workshop on Discrete Event Systems, WODES’04, 22-24 September 2004.