Supply Chain Optimization with both Production and Transportation Integration: Multiple Vehicles for a Single Perishable Product

Philippe Lacomme[a], Aziz Moukrim[b], Alain Quilliot[a], Marina Vinot[a1]

[a] Laboratoire d'Informatique (LIMOS, UMR CNRS 6158), Campus des Cézeaux, 63177 Aubière Cedex, France.

[b] Sorbonne Universités, Université de Technologie de Compiègne, (Heudiasyc UMR CNRS 7253), CS 60 319, 60203 Compiègne France.

This paper deals with an extension of the integrated production and transportation scheduling problem (PTSP) by considering multiple vehicles (PTSPm) for optimisation of supply chains. The problem reflects a real concern for industry since production and transportation subproblems are commonly addressed independently or sequentially, which leads to sub-optimal solutions. The problem includes specific capacity constraints, the short lifespan of products and the special case of the single vehicle that has already been studied in the literature. A greedy randomised adaptive search procedure (GRASP) with an evolutionary local search (ELS) is proposed to solve the instances with a single vehicle as a special case. The method has been proven to be more effective than those published and provides shorter computational times with new best solutions for the single vehicle case. A new set of instances with multiple vehicles is introduced to favour equitable future research. Our study extends previous research using an indirect resolution approach and provides an algorithm to solve a wide range of one-machine scheduling problems with the proper coordination of single or multiple vehicles.

Keywords*:* supply chain coordination; transportation; scheduling; vehicle routing problem; integration.

1 Introduction and literature review

1.1 Introduction to integrated problems

The Production and Transportation Scheduling Problem (PTSP) modelled a supply chain problem with production and transportation where product perishability must be addressed (which is common in the food, chemical and pharmaceutical industries). The PTSP is a problem where both scheduling and routing must be jointly solved to have a proper coordination between the production on a single production facility and the transportation, taking the product lifespan into consideration. In this problem, once a lot of products is produced, it must be directly transported to various customer sites within its limited lifespan. A solution to a PTSP is composed of production operations (starting times have to be computed) and transportation operations that define sub-trips (terminology used in the routing community) from the depot (production facility) to a set of customers.

Supply chain management is a cross-functional approach that includes managing the movement of raw materials into an organisation, the processing of materials into finished products (production), and the movement of finished products out of the organisation and toward the end consumers (transport). All these functions (production, storage, transportation) can be seen as independent subproblems or can be integrated under one plan in supply chains. In the literature, a tremendous amount of research considers the production and transportation subproblems successively, and in the vast majority of companies, the scheduling problem is

[1] Corresponding author. e-mail addresses: placomme@isima.fr (P. Lacomme); aziz.moukrim@utc.fr (A. Moukrim); alain.quilliot@isima.fr (A. Quilliot); marina.vinot@isima.fr (M. Vinot)

solved first and the routing problem is then addressed, although this kind of approach does not lead to an optimal solution.

Production and transportation stages at the planning level are often linked by intermediate inventory stages whose management strongly depends on the proper coordination between production and transportation as an integrated resolution. Note that integrated production and distribution has a significant impact on customer service, favouring product delivery in due time. The connection between the production (processing stations) and transportation is a highly desirable goal as stressed in numerous publications, including Belo-Filho et al. (2015), with a problem with several production facilities and perishable products.

The integration of production and transportation can be relevant when the products are perishable. The product perishability is related to an "expiration date", which commonly appears on product labels to indicate the date from which the quality of the product is no longer guaranteed by the manufacturer (Rivera and Lallmahomed, 2015). Roscoe and Baker (2014) underline that perishable goods are quite specific and influence the packing, the storage and the transportation mode. The management of perishable products is critical for integrating production and transportation scheduling in a coordinated manner.

As stressed by Chen (2010), the integrated production and transportation models at a detailed scheduling level are fairly recent and the majority of models attempt to jointly optimise job-by-job production and transportation by considering the customer service level at the individual job level. Integrated production and routing problems are receiving more and more attention, and several surveys are available, including Moons (2017), Chen (2010) and Saramiento et al. (1999). Let us note that integrated production and transportation problems fall into two categories of problems according to Kuhn and Liske (2017): (a) a Production Distribution Problem; or (b) a Sourcing Production Problem. A large number of publications consider the Production Distribution Problem, i.e., problems where the production is completed first and the products are distributed afterwards.

Chen (2010) provides a survey on integrated production and outbound distribution scheduling problems and classifies these existing problems into several different classes. These problems often have different names: Integrated Production and Outbound Distribution Scheduling (IPODS), Integrated Production and Distribution Problem (IPDP) or, in our case, Production and Transportation Scheduling Problem (PTSP). In these problems, a solution is characterised by a production schedule that plans the date and the location where each demand is processed, and a delivery schedule that satisfies the orders in each shipment, the number of vehicles and the delivery date. Problems can vary depending on the number of production facilities with machine configuration (single-machine, parallel-machine, flow shop, etc.), the number of customers, and customers' characteristics, including delivery time windows and deadlines. The number of available vehicles and their capacities may be limited or not. Chen (2010) classifies existing IPODS problems into five classes considering individual and immediate delivery, batch delivery to a single customer by direct shipping, batch delivery to multiple customers by direct shipping, batch delivery to multiple customers by routing, and fixed delivery departure dates.

1.2 Intregrated problem with one production facility and one perishable product

Perishability naturally occurs in the food, chemical and pharmaceutical industries and its management creates several complications in the scheduling of production and transportation. Special efforts are required to reduce the waste and cost of the storage and transportation of perishable products. Chen (2010) mentions studies on the perishable products with time-sensitive constraints (Armstrong et al., 2008; Devapriya et al., 2006; Geismar et al., 2008). The problems studied in these articles belong to the same class: batch delivery to multiple customers with routing. The time-sensitive constraints in these problems are due to industrial chemical

compounds that must be delivered within a certain time limit once produced. In the survey of Moons (2017) on integrated production scheduling and vehicle routing problems, two studies with one perishable product and one production facility are mentionned (Karaoğlan et al., 2017; Devapriya et al., 2017).

Armstrong et al. (2008) study the production and transportation problem with a single vehicle and a single production facility. The order in which customers may receive deliveries is fixed. Each customer requests a delivery quantity with a time window for receiving it. The lifespan of the product begins as soon as the production of a customer's order is completed. The problem then turns into a routing problem, minimising the number of non-serviced customers for a subset of customers.

Devapriya et al. (2006) focus on a problem with one production facility and a large fleet size. The lifespan of the perishable products immediately begins when the last order of a batch has been completed. The delivery of a batch order therefore begins as soon as the production is achieved. The objective is to minimise the total transportation cost, i.e., the cost of delivery time and the number of vehicles required to satisfy all the demands.

Table 1. Integrated problem with one production facility and one perishable product.

	Production				Transportation										Objective		
					Vehicles				Extra constrant								
	One production facility	Batch production	One product	Production time	A single vehicle	Homogeneous fleet	Unlimited number	Limited number	Transportation time	Several trips	Fixed customer order	Time windows	Delivery due date	Produits périssables	Makespan	Cost	Number of satisfied demans
Geismar *et al.* (2008)	•	•	•	•	•				•	•				•	•		
Armstrong *et al.* (2008)	•	•	•	•	•				•	•	•	•		•			•
Karaoğlan and Kesen (2017)	•	•	•	•	•				•	•				•	•		
Devapriya (2006)	•	•	•	•		•	•		•					•		•	
Devapriya *et al.* (2017)	•	•	•	•		•		•	•	•				•		•	

Geismar et al. (2008) address a problem (PTSP) with a network of multiple customers, a single facility with a constant production rate $r > 0$, and a single vehicle with capacity Q. The objective is to minimise the time required to serve all customers, commonly referred to as the makespan. Furthermore, due to a product with a lifespan B, waiting time must be addressed and the problem can be considered as a two-machine flow shop with maximal time lags. Let us note that in Geismar et al. (2008), the problem focuses on a situation where the lifespan is defined by an expiration delay between the end of production and customer delivery. To jointly solve the scheduling and routing problem, the authors introduce a genetic and a memetic algorithm with the algorithm of Gilmore and Gomory (1964) and several lower bounds. The problem has been proved to be NP-hard in the strong sense.

The problem described by Geismar et al. (2008) was recently solved by Karaoğlan et al. (2017) using a branch-and-cut (B&C) algorithm. The algorithm uses several valid inequalities taken from the existing literature, and a local search based on a simulated annealing approach is used to improve upper bounds. Numerical experiments are achieved on the same instances as Geismar et al. (2008) and prove that the B&C is strongly competitive. The objective function consists in the minimisation of the makespan (Table 1).

Recently, Devapriya et al. (2017) addressed an extension (referred to as IPDSP) of the PTSP introduced by Geismar et al. (2008) where the fleet size is a decision variable.

1.3 Solution representation for scheduling and routing problems

Geismar et al. (2008) base their approach on an indirect representation of the solutions with giant trips. The difficulty of integration is addressed with the dedicated (Gilmore and Gomory, 1964) algorithm to solve the no-wait flow shop problem and obtain an optimal order of sub-trips. The final solution is then deduced by relaxing the no-wait constraint, keeping the constraints on product perishability. Let us note that for the PTSP addressed by Geismar et al. (2008), the solution is composed of one trip with several ordered sub-trips with increasing starting times since there is no assignment problem of vehicles to be solved.

The PTSP with the constraint of perishable products can be modelled as a permutation flow shop scheduling problem on two machines with maximal time lags. Fondrevelle et al. (2004) prove that this problem is strongly NP-hard even when all the maximal time lags have the same positive value. Nevertheless, note that this problem is a mix between the classical and the no-wait flow shop scheduling problem that can be polynomially solved with the algorithm of Johnson (1954) and with the algorithm of Gilmore and Gomory (1964), respectively. The first research on the two-machine flow shop problem was proposed by Johnson (1954) who defined rules to obtain the optimal schedule. It is commonly accepted that most of the scheduling problem-solving approaches take advantage of a modelling based on the disjunctive graph of Roy et al. (1964). Numerous authors have introduced approaches for graph generation and search space exploration. Bierwirth's (1995) proposal for the job shop remains within the global trend of indirect representation schemes and proves that it is possible to express the machine selections as a vector by repetitions that define a topological order of nodes. Similar remarks hold for the routing problem that has received a considerable amount of attention in recent years. The idea of splitting a giant trip was introduced by Beasley (1983) and was first included in a global framework for the routing problem by Lacomme et al. (2001). The total number of methods that take advantage of such an approach has strongly increased in recent years (as mentioned in the state of the art of Prins et al. (2014)). The second key feature used in routing approaches lies in the local search procedure, which is commonly based on a swap within a trip, a 2-Opt within a trip, a swap between two trips and a 2-Opt between two trips (see Lacomme et al. (2001) and Prins (2004)).

The remainder of this paper is structured as follows. The following section introduces the problem of interest with multiple vehicles. Section 3 presents a GRASP×ELS framework designed to address the key features for an integrated resolution. Section 4 provides the computational evaluation of the problem by considering a single vehicle. Section 5 illustrates the computational evaluation of the problem with multiple vehicles. The last section provides some conclusions and avenues for further research.

2 Problem definition for the PTSPm

2.1 Definition

The PTSPm (PTSP with multiple vehicles) is an extension of Geismar et al.'s (2008) proposal by taking not just a single but multiple vehicles into consideration and that can be formally defined by considering n customers and N vehicles:

E	set of customers
0	production facility located in $(x, y) = (0, 0)$
r	production rate $r \in \{1, 2, 3\}$
B	lifespan of the product
V	set of available vehicles, $k \in \{1, \dots, N\}$
Q	capacity of the vehicles
q_i	demand of customer i, $i \in \{1, \dots, n\}$
(x_i, y_i)	coordinates of location of customer i, $i \in \{1, \dots, n\}$

$\tau_{i,j}$ transportation time from customer i to customer j; this matrix satisfies the triangle inequality

To solve this problem, three subproblems must be jointly solved: the assignment, the routing and the scheduling. When the assignment and the routing problems are solved, the residual problem can be modelled as a two-stage hybrid flow shop with maximal time lags where the machine in the first stage is the single production facility and where each vehicle defines a machine in the second stage. The Johnson (1954) algorithm and the Gilmore and Gomory (1964) algorithm can no longer be used to solve the scheduling part of the PTSPm.

A solution could be composed by a set of jobs J, with $card(J) = nj$ with $nj \leq n$. A job j includes a set of customers to be served and is composed of two operations O_{j1} and O_{j2}^k. Operation O_{j1} is a production operation assigned to the production facility, modelling the production of the demands of the customers in the job j. The operation O_{j2}^k is a transportation operation assigned to the vehicle k, modelling the sub-trip that makes it possible to deliver the demands of the customers in job j from the production facility, also referred to as the depot. The trip of vehicle k is defined by a sequence of all the transport operations (sub-trips) assigned to vehicle k. The following constraints must hold:

- each transportation operation begins and ends at the production facility, also referred to as the depot or depot node, depending on the routing terminology;
- the order of the production operations and of the transportation operations should be identical for a given vehicle between two jobs;
- a vehicle achieves at most one trip;
- all customers of a sub-trip O_{i2}^k must be served within B time units after the end of the production operation O_{i1};
- the total demands in a sub-trip O_{i2}^k cannot exceed the vehicle capacity $\sum_{j \in O_{i2}^k} q_j \leq Q$;
- deliveries must not be split (each customer must be delivered only once).

The following notations are used to caracterise a solution:

C_j set of customers included in job j
O_{j1} production operation of job j
O_{j2}^k transportation operation of job j assigned to vehicle k
p_{j1} duration of the production operation of job j
p_{j2} duration of the transportation operation of job j
p'_{j2} duration of the transportation operation of job j without the empty transport to the depot
s_{j1} starting time of the production operation of job j
s_{j2} starting time of the transportation operation of job j
f_{j1} finishing time of the production operation of job j
f_{j2} finishing time of the transportation operation of job j

Figure 1 represents a solution of a PTSPm with two vehicles and five customers. In this solution, the customers are divided into three groups to define three jobs:

- the first job (with the operations O_{11} and O_{12}^1) includes customers 1 and 2;
- the second job (with the operations O_{21} and O_{22}^2) contains customer 3;
- the third job (with the operations O_{31} and O_{32}^1) includes customers 4 and 5;

The trip of vehicle 1 is composed of two transportation operations (O_{12}^1 and O_{32}^1) and the trip of vehicle 2 is composed of one transportation operation O_{22}^2. The solution presented on the Gantt chart uses the following values for the demands of the customers, $q_1 = 1$, $q_2 = 1$, $q_3 = 6$, $q_4 = 1$, $q_5 = 3$, and the transportation times are equal to $\tau_{0,1} = \tau_{2,0} = \tau_{0,4} = \tau_{4,5} = 3$,

$\tau_{1,2} = \tau_{0,3} = 2$ and $\tau_{5,0} = 1$. For the first job, the duration of the production operation $p_1 = 2$ time units since the production rate $r = 1$ and $q_1 = q_2 = 1$, and the duration of the transportation operation $t_1 = 8$ time units since $\tau_{0,1} + \tau_{1,2} + \tau_{2,0} = 8$. The lifespan of the product is repected: $f_{j2} - f_{11} - \tau_{2,0} = 10 - 2 - 3 = 5 \leq B = 7$.

Let us note that $\tau_{x,y}$ in this problem denotes a transportation time between customer x and customer y that can be precomputed by considering either the minimal distance traveled by the vehicle between customers or the minimal time necessary to travel between two customers that are equal since the speed factor for the vehicles is equal to one. The jobs have to be defined and ordered in order to minimise the makespan C_{max}, i.e., the arrival time of the last vehicle at the depot.

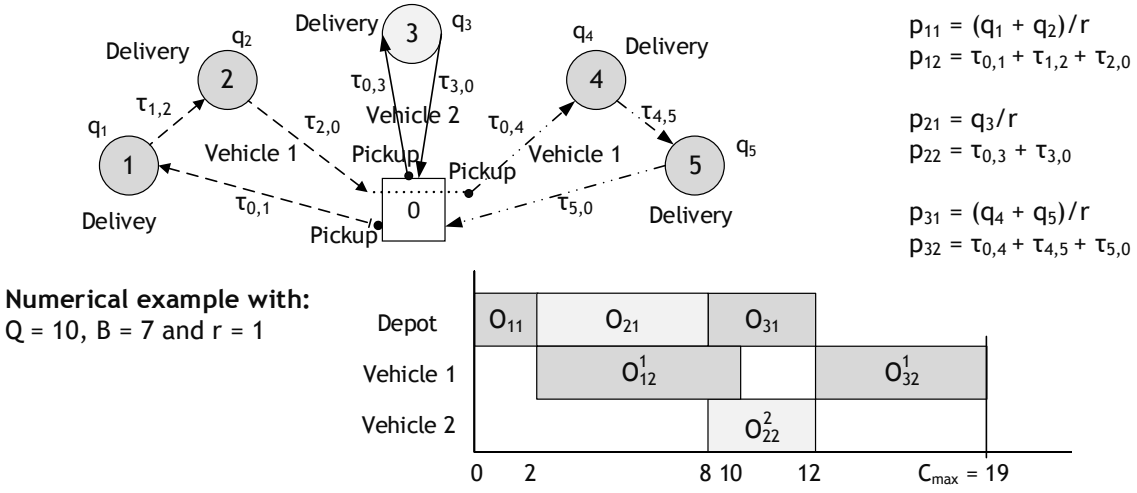

Figure 1. Example of one PTSPm solution with five customers and two vehicles.

2.2 Linear formulation of the PTSPm

The combined production scheduling and vehicle routing problem is formulated as a mixed integer linear programming model based on five binary variables (y_{bi}, h_b, x_{ij}^b, z_{bc}, a_{bk}) and seven integer variables (s_{b1}, s_{b2},p_{b1}, p_{b2}, p'_{b2},q_i^-, C_{max}). The binary decision variables are related to the disjunctions between operations and the integer variables are related to the starting and finishing times or duration of the operations.

Notations for the binary variables:

$y_{bi} = \begin{cases} 1 \\ 0 \end{cases}$	if customer i is in job b otherwise	$i \in E$ $b \in J$
$h_b = \begin{cases} 1 \\ 0 \end{cases}$	if job b is composed of at least one customer otherwise	$b \in J$
$x_{ij}^b = \begin{cases} 1 \\ 0 \end{cases}$	if customer i is serviced immediately before customer j in job b otherwise	$(i,j) \in E^2$ $b \in J$
$z_{bc} = \begin{cases} 1 \\ 0 \end{cases}$	if job b is scheduled before job c otherwise	$(b,c) \in J^2$
$a_{bk} = \begin{cases} 1 \\ 0 \end{cases}$	if job b is assigned to vehicle k otherwise	$b \in J$ $k \in V$

Job assignment requirement: this set of constraints ensures that one customer is assigned to one and only one job.

$$\forall i \in E \qquad \sum_{b \in J} y_{bi} = 1 \qquad (1)$$

Vehicle capacity requirements: this constraint ensures that the customers assigned to a job can be serviced by one vehicle considering the total amount of demands of customers.

$\forall b \in J \qquad \sum_{i \in E} y_{bi} \cdot q_i \leq C \qquad (2)$

Job definition: this constraint ensures that if job b encompasses no customer ($h_b = 0$), no customer is assigned to the job ($y_{bi} = 0$)

$\forall b \in J, \forall i \in E \qquad y_{bi} \leq h_b \qquad (3)$

Job duration for the production: this constraint defines the job processing time p_b on the production considering the set of customers assigned to the job. If customer i is assigned to job b, then $y_{bi} = 1$ and $\sum_{i \in E} q_i . r$ is the sum of q_i moderated with the production rate r.

$\forall b \in J \qquad p_b = \sum_{i \in E} y_{bi} \cdot q_i . r \qquad (4)$

Vehicle assignment to one vehicle: this constraint ensures that if job b is used (there are one or several customers assigned to b), i..e., $h_b = 1$, then one variable a_{bk} is assigned to 1, i.e., one vehicle k is assigned to job b.

$\forall b \in J \qquad h_b = \sum_{k \in V} a_{bk} \qquad (5)$

Customer order in the jobs: this set of constraints ensures that each customer has a single predecessor and a single successor

$\forall j \in E \qquad \sum_{b \in J} \sum_{\substack{i \in E \cup \{0\} \\ i \neq j}} x_{ij}^b = 1 \qquad (6)$

$\forall i \in E \qquad \sum_{b \in J} \sum_{\substack{j \in E \cup \{0\} \\ j \neq i}} x_{ij}^b = 1 \qquad (7)$

Depot definition: this constraint ensures that if job b encompasses at least one customer ($h_b = 1$), the sub-trip starts and ends at the depot.

$\forall b \in J \qquad h_b = \sum_{j \in E} x_{0j}^b \qquad (8)$

$\forall b \in J \qquad h_b = \sum_{i \in E} x_{i0}^b \qquad (9)$

Customer assignment to a job: this constraint ensures that if a customer i is assigned to job b ($y_{bi} = 1$), then customer i has one predecessor, i. e., $\exists j / x_{ij}^b = 1$. Similar remarks hold for constraint 13, considering the successor.

$\forall i \in E, \forall b \in J \qquad y_{bi} = \sum_{\substack{j \in E \cup \{0\} \\ j \neq i}} x_{ij}^b \qquad (10)$

$\forall i \in E, \forall b \in J \qquad y_{bi} = \sum_{\substack{j \in E \cup \{0\} \\ j \neq i}} x_{ji}^b \qquad (11)$

Sub-trip eliminations: these constraints are the Miller-Tucker-Zemlin (MTZ) constraints. This constraint uses q_j^-, referred to as the total amount of products remaining in the vehicle after servicing customer j.

$\forall b \in J, \forall (i,j) \in E^2, i \neq j \quad q_j^- - q_i^- + x_{ij}^b . C \leq C - q_j \qquad (12)$

$\forall b \in J, \forall (i,j) \in E^2, i \neq j \quad q_i^- \leq C - q_i \qquad (13)$

Job duration for transport: this constraint defines the sub-trip duration to service all customers assigned to job b

$\forall b \in J \qquad p_{b2} = \sum_{i \in E \cup \{0\}} \sum_{\substack{j \in E \cup \{0\} \\ j \neq i}} x_{ij}^b \cdot \tau_{ij} \qquad (14)$

Job duration for transport: this constraint defines the trip duration to service all customers assigned to job b not considering the empty transport from the last customer to the depot.

$\forall b \in J \qquad p'_{b2} = p_{b2} - \sum_{i \in E} x_{i0}^b . \tau_{i0} \qquad (15)$

Disjunctive constraints at the production end: this constraint ensures that the production operation of job c and the production operation of job d cannot be performed at the same time at the production facility.

$\forall (b,c) \in J^2, c \neq b \qquad s_{b1} + p_{b1} \leq s_{c1} + (1 - z_{bc}) . H \qquad (16)$

$\forall (b,c) \in J^2, c \neq b \qquad s_{c1} + p_{c1} \leq s_{b1} + z_{bc} . H \qquad (17)$

If $z_{bc} = 1$ (job b is scheduled before job c), constraint (18) can be rewritten as $s_{b1} + p_{b1} - s_{c1} \leq 0$, meaning that $s_{b1} + p_{b1} \leq s_{c1}$, ensuring that production of c on the production facility cannot start before the end of the production operation of job b. If $z_{bc} = 0$, constraint (18) holds, and constraint (19) can be rewritten as $s_{c1} + p_{c1} \leq s_{b1,}$ meaning that job c is processed first and job d second for the production operations.

Disjunctive constraints for transport: this constraint ensures that the transport operations of two jobs b and c are not performed at the same time by the same vehicle k.

$\forall (b,c) \in J^2, c \neq b \qquad s_{b2} + p_{b2} \leq s_{c2} + (3 - z_{bc} - a_{bk} - a_{ck}).H \qquad (18)$

$\forall (b,c) \in J^2, c \neq b \qquad s_{c2} + p_{c2} \leq s_{b2} + (2 - z_{bc} + a_{bk} + a_{ck}).H \qquad (19)$

If $a_{bk} = a_{ck} = 1$, this means that the two transport operations are assigned to the same vehicle k and the constraints can be rewritten as: $s_{b2} + p_{b2} \leq s_{c2} + (1 - z_{bc}).H$ (20) and $s_{c2} + p_{c2} \leq s_{b2} + z_{bc}.H$ (21). If either a_{bk} or a_{ck} are not assigned to 1, the two constraints hold, regardless of the value of z_{bc}.

Precedence constraints per operation: this constraint ensures that the transport operations of one job are performed according to the production-operation sequence first, followed by the transport-operation sequence.

$\forall b \in J \qquad s_{b1} + p_{b1} \leq s_{b2} \qquad (20)$

Lifespan products: this constraint ensures that the lifespan of the products are addressed. The product lifespan defines a maximal delay between the delivery of the last customer of a sub-trip of a job b, i.e., $s_{b2} + p'_{b2}$, and the finishing time of the production referred to as $s_{b1} + p_{b1}$: the difference is upper bounded by B.

$\forall b \in J \qquad s_{b2} + p'_{b2} \leq s_{b1} + p_{b1} + B \qquad (21)$

Makespan. When minimising the makespan, the following constraints should be added to define new integer variables. This constraint ensures that $C_{max} = max_{b \in J}(d_{b2} + p_{b2})$

$\forall b \in J \qquad s_{b2} + p_{b2} \leq C_{max} \qquad (22)$

2.3 Modelling

In the PTSPm, a sub-trip is fully defined by an ordered sequence of operations: (1) starting at the depot with a pickup operation; (2) defining an ordered sequence of delivery operations; and (3) finishing at the depot. The loading of the vehicle can be represented as a decreasing function (see Fig. 2) during each sub-trip.

Figure 2 gives some details of the solution presented in Fig. 1. Figure 2 represents both the vehicle load (on the top) and the proper coordination between production and transportation over time, with an explicit modelling of pickup and delivery operations (at the bottom).

Table 2. Example of solution for the two-vehicle PTSPm.

	O_{11}	O_{12}^1				O_{21}	O_{22}^2			O_{31}	O_{32}^1			
	0	**1**		**2**	**0**		**0**	**3**	**0**		**0**	**4**	**5**	**0**
Starting time	0					2				8				
Time Windows				]-∞;9]				]-∞;15]					]-∞;19]	
Departure time		2	5	7			8	10			12	15	18	
Arrival time			5	7	10			10	12			15	18	19

Time constraints on sub-trips

Time dependency only exists between successive sub-trips of the same trip since all sub-trips of one trip are assigned to the same vehicle. Concerning the trip of vehicle 1 in Fig. 2, the previous remark implies that $s_{32} \geq f_{12}$. Moreover, the earliest starting time of O_{31} depends on the starting time O_{21}, $s_{31} \geq s_{21} + p_{21}$. For the trip of vehicle 2, the earliest starting time of the transportation operation O_{22}^2 does not depend on the finishing time of O_{12}^1 (operations in the diagonal rectangle in Fig. 2). Vehicle 2 can potentially start the transportation operation O_{22}^2 before the transportation operation O_{12}^1, as depicted in Fig. 2 and Table 2. Assignment of

vehicles to sub-trips is a challenging problem that should be solved avoiding extra waiting time on the Gantt chart.

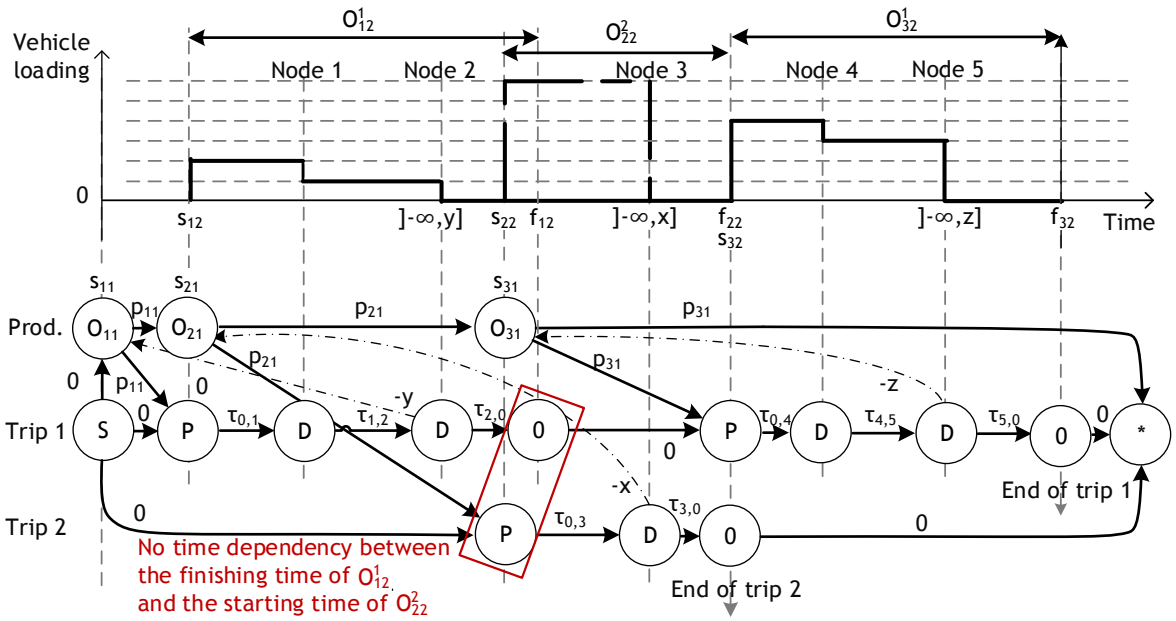

Figure 2. Example of a two-vehicle workload.

Conjunctive and disjunctive arcs to link production and transportation with maximal time lags

The delivery nodes where products are unloaded have a time window constraint that defines the maximal arrival time acceptable for the customer, which is correlated to the product perishability. The time windows are related to the earliest starting time of the production on the facility. They are modelled by maximal time lags that are defined using a negative arc between one delivery node and the production node in this special case.

As shown in Fig. 2, the time window $]-\infty; y]$ on node 2, which corresponds to a delivery node, means that the maximal duration between s_{11} and the arrival time at node 2 must not exceed y units of time. Similarly, node 3 must be served within a delay of x units of time after s_{21}. One positive arc gives the minimal delay between the earliest starting time of two operations. For example, $\tau_{1,2}$ models a minimal duration required between departure time at node 1 and arrival time at node 2. Normally, these so-called conjunctive arcs are valuated with the shortest path value between two nodes modelling transport, and with the duration of production between two production nodes. Disjunctive arcs are required to define the order of sub-trips in a trip. This type of time windows arises: (1) in routing problems including but not limited to the Dial-A-Ride Problem where both maximal route duration and maximal customer riding time must be taken into account; and (2) in scheduling problems.

In the Gantt chart in Fig. 1, the first job provides products for customers 1 and 2 for a production duration of $p_{11} = 2$ time units, the earliest starting time of O_{11} is equal to 0 and vehicle 1 is available at time 0. Therefore, on node 1 (Fig. 2), the time window is defined by $]-\infty; 2+\delta_1]$ and $]-\infty; 2+\delta_2]$ for node 2, where δ_i is the lifespan of products for customer i. The earliest starting time of a transportation operation is the maximal value between the finishing time of the previous transportation operation assigned to the same vehicle and the finishing time of the production operation. Similar considerations make it possible to define the earliest starting time O^2_{22} at time 8 with a time window $]-\infty; 8+\delta_3]$ for nodes 3. Because the problem being considered is a single product PTSPm, the lifespan is equal for all nodes, i.e., $\delta_i = B, \forall i$, implying that all time windows of all delivery nodes of the same sub-trip are

identical and equal to $]-\infty; x+B]$, where x is the duration of the production linked to the transport.

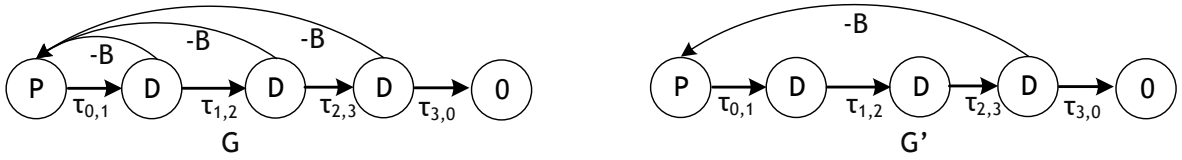

Figure 3. Sub-trip with maximal time lag simplification.

The earliest arrival time of a vehicle increases within a sub-trip. Therefore, if an earliest arrival time exists for the last delivery operation of the sub-trip that meets the time window constraints, all previously scheduled operations with time window constraints will hold. In Fig. 3, sub-trip G with three delivery nodes takes the three maximal time lags modelling the product lifespan explicitly into account. For PTSPm with a single product, which consequently defines the same lifespan for all customers, a new graph G' can be used to model only the time window of the last delivery operation of each sub-trip.

The feature that differentiates this problem from previous ones is the combination of the scheduling decisions with the limited product lifespan and the vehicle routing decisions. These interdependent decisions lead to the possibility that the product may expire before it reaches a customer if an unprofitable scheduling solution is chosen. The objective consists in solving the scheduling and routing problem by minimising the makespan to comply with the classical objective function introduced by Geismar et al. (2008) and providing a semi-active solution, i.e., a left-shifted solution.

3 A PTSPm resolution based on a GRASP× ELS

This proposal is based on a GRASP×ELS, which introduces a new splitting algorithm for the assignment and the routing problem, and a new resolution of the scheduling problem using an approach based on a disjunctive graph. The disjunctive graph is specially designed to efficiently take the perishability constraint into account using maximal time lags.

3.1 Key features for a PTSPm resolution

The key point for the PTSPm resolution is to alternate between solutions encoded by giant trips (TSP - Traveling Salesman Problem - permutation list on the n customers), set of trips (VRP - Vehicle Routing Problem - on the n customers) that comply with the ordered set of customers defined by the giant trip, and a flow shop resolution by considering jobs linked to the trips (PTSPm on the n customers). The approach is a combination of three search space representations that favour partial enumeration of the whole search space (Fig. 4).

The iterative search space exploration takes advantage, first, of the indirect representation of solutions by using a giant trip. Second, a set of feasible sub-trips minimising the total transportation time is computed using a split-based approach on the sequence of customers that has been defined by the giant trip. Note that the specific local search for node routing can be applied to obtain a high-quality local routing solution.

The resolution framwork is expressed by the following key features:

- a local search for routing with 2-Opt or swap within a sub-trip, and 2-Opt or swap between two sub-trips, based on classical VRP neighbourhoods;
- a splitting procedure for proper coordination between production and transportation;
- a scheduling procedure based on a disjunctive graph with the sub-trips made by the splitting procedure;
- a concatenation procedure to obtain a new giant trip.

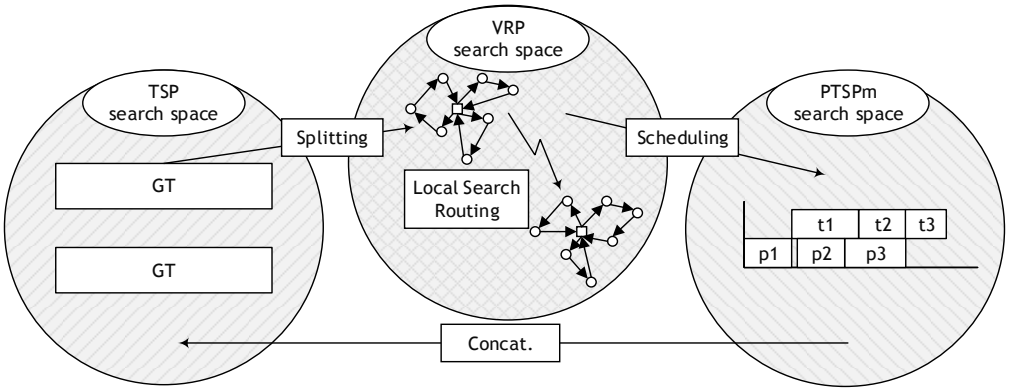

Figure 4. The three search spaces to build a PTSPm solution.

The GRASP×ELS approach is very different from Geismar et al.'s proposal and differs in both scope and the resolution scheme that are summarised in Table 3.

Table 3. GRASP×ELS approach vs. Geismar et al.'s approach

		(Geismar et al., 2008)	GRASP×ELS
Scope		PTSP	PTSP + PTSPm
Resolution scheme	Scheduling	Gilmore-Gomory	Disjunctive graph
	Routing	Split of the classical VRP	Split with production and transport consideration
	Local search for scheduling	/	Based on the proposal of Laarhoven et al. (1992) and Grabowski et al. (1986)
	Local search for routing	2-Opt moves	Inter/extra trips 2-Opt and swap moves
	Solution representation	Chromosome	Giant trip

3.2 The GRASP×ELS Principle

GRASP×ELS is a hybridisation (Prins, 2009) of a GRASP (Greedy Randomised Adaptive Search Procedure) (Feo and Resende, 1995), with an ELS (Evolutionary Local Search) (Wolf and Merz, 2007; Prins, 2004) that takes advantage of both methods. The multi-start approach of the GRASP, which provides *np* initial solutions, is based on a greedy randomised heuristic, and solutions are then improved by a local search procedure. The second metaheuristic can be expressed as an extension of the ILS (Iterated Local Search) (Lourenço et al., 2003), referred to as ELS and proposed by Wolf and Merz (2007). The solution space investigation is achieved by the GRASP that favours diversity, and the intensification phase is devoted to the ELS via a proper local search investigation into the local search space. In addition, to combine GRASP with ELS, another important feature is the alternation between solution spaces as stressed by Prins (2004) and Prins et al. (2014). Converting a PTSPm solution into a giant trip is achieved by a concatenation procedure, and the operation converting a giant trip into a VRP solution is achieved by a dedicated splitting procedure (Split).

Algorithm 1 is composed of a loop from line 13 to 36, which iterates on a new starting solution for the GRASP. For each initial solution, the procedure `Generation_of_initial_solution()`, using a greedy heuristic creates a new initial solution submitted to the Split procedure (line 16) to obtain a routing solution that is improved (line 17) by the local search procedure `Local_Search_on_Routing(S, nl)`. Secondly, the `Scheduling()` procedure computes a solution (line 18). If the cost of the solution S, $f(S)$ is better than the cost of the best PTSPm solution found, S^*, f^* (line 20), S becomes the new best solution. The *while* loop from lines 23 to 35 is the ELS and encompasses the loop (lines 25 to

32) for neighbourhood generations. According to the common Split approach, the mutation operator (line 26) is defined on the giant trip *T* and not on the solution.

The random heuristics, referred to as `Generation_of_initial_solution()`, generates initial solutions. This method uses two heuristics to build the initial solution.

The first one, with a probability of 0.9 is based on a greedy randomised heuristic based on a path-scanning-like approach. The first heuristic builds sub-trips one-by-one, starting from the depot, and extends each sub-trip customer-by-customer using two criteria:

- the extension step at node i considers that the sub-trip moves to the nearest customer not yet served. If vehicle load $\geq Q$, the next customer to be served is selected to minimise the distance to the depot;
- the next customer to be served is the customer that minimises the transportation time.

The second heuristic with a probability of 0.1 classifies the customers by decreasing transportation time.

Algorithm 1. GRASP× ELS for the PTSPm

```
1.  procedure GRASP×ELS
2.  global parameters
3.    np: number of GRASP iterations (initial solutions)
4.    ne: maximal number of iterations per ELS
5.    nr: maximal number of iterations without improvement per ELS
6.    nd: number of diversifications (mutations)
7.    nl: number of local searches on the routing
8.    ng: number of local searches on the scheduling
9.  output parameters
10.   S*: best PTSPm solution found
11. begin
12.   f* := ∞; O := ∅
13.   for p := 1 to np do                  // GRASP loop
14.     S := call Generation_of_initial_solution ()
15.     T := call Concat (S)
16.     S := call Split (T)
17.     S := call Local_Search_on_Routing (S, nl)
18.     S := call Scheduling (S)
29.     T := call Concat (S)
20.     if (f(S) < f*) then f* := f(S); S* := S; // f: the cost of a solution
21.     endif
22.     i, r := 0
23.     while (i < ne) and (r < nr) do   // ELS loop
24.       i := i + 1; f" := ∞
25.       for j := 1 to nd do            // mutation loop
26.           T' := call Mutation (T)
27.           S' := call Split (T')
28.           S' := call Local_Search_on_Routing (S', nl)
29.           S' := call Scheduling (S')
30.           T' := call Concat (S')
31.           if (f(S') < f") then f" := f(S'); T" := T'; S" := S'; endif
32.       endfor
33.       if (f" < f*) then S*:= S"; endif //  if a new best solution update S*
34.       T := T";          // best ELS solution becomes the new initial solution
35.     endwhile
36.   endfor
37. end
```

The local search on the routing (`Local_Search_on_Routing()`) is achieved using several classical VRP neighbourhood moves to improve the initial solution, namely 2-Opt within a sub-trip and insertion within a sub-trip. At each iteration, the first improved move is executed but requires verification of the sub-trip feasibility by considering the capacity of the vehicles and the lifespan. All solutions are converted into a giant trip by random concatenation of their sub-trips and then evaluated by the Split procedure. The key point for the efficiency of GRASP×ELS is to alternate between solutions encoded as giant trips and PTSPm solutions.

3.3 Split procedure

Giant trips are evaluated via the Split procedure that minimises the total trip duration subject to the vehicle capacity, as reported by Beasley (1983) and Prins et al. (2014). Split is a key procedure used to convert a giant trip into a VRP solution (with respect to the sequence) and it is based on the classical Split procedure adjusted to address the specific PTSPm constraints. The algorithm provides an efficient solution by building an auxiliary graph $G = (X, A)$, where X represents n + 1 nodes numbered from 0 to n. Node 0 is a dummy node, while the nodes 1 ... n correspond to the sequence of the giant trip $GT = (\sigma_1, \dots, \sigma_n)$. An arc (i, j) belongs to A if a sub-trip serving customers σ_{i+1} to σ_j (inclusive) is both weight-feasible and lifespan-feasible. An initial label is set at node 0 and the labels are propagated from node to node in G using arcs. The best label at node n is kept as the best solution.

The routing problem has a resource constraint since there is a limited number of available vehicles, each vehicle defining one specific machine to be scheduled. The Split procedure for the PTSPm is an extension of the split introduced for the HVRP (Heterogeneous Vehicle Routing Problem) by Duhamel et al. (2012) because there is the earliest finishing time of the production to take into account as well as the earliest finishing time of vehicles for the previous sub-trip. Computation of the resource-constrained shortest path in the graph is typically achieved by a label-correcting algorithm. According to Desrocher (1988), several labels per node have to be handled and the key point consists in defining the label structure with the cost and the resource availability. The generic Split procedure of Duhamel et al. (2012) is based on three key points:

- a label description based on the resource availability;
- a dominance rule between labels to improve the running time by keeping only promising labels on the node;
- a propagation rule to define a label from node i to node j depending on the customer sequence $(\sigma_{i+1}, \dots, \sigma_j)$.

Let $L_i^p = (d_{Pi}, d_{V1i}, \dots, d_{VNi})$ be the p^{th} label assigned to node i when the number of vehicles $V = N$, where $L_i^p(j)$ is the earliest finishing time of vehicle j for $j = 2..N + 1$ and $L_i^p(1)$ is the earliest finishing time of the production. This corresponds to a feasible split of the initial customers $(\sigma_1, \dots, \sigma_i)$ into sub-trips, where d_{Pi} is the earliest production finishing time and d_{Vki} is the transportation finishing time for the vehicle V_k. The initial label at node 0 is defined as $L_0^1 = (0,\ 0, \dots, 0)$. It corresponds to the empty solution where the finishing time is equal to 0. Given the arc $(i, j) \in A$ and if this arc is put on vehicle V_k, label L_i^p generates label $L_j^q = \left(d_{Pj},\ d_{V1j}, \dots, d_{VNj}\right)$ using the following propagation rule:

- $d_{Pj} = max(\ d_{Pi} + \sum_{l=i+1}^{j} p_{\sigma_l}\ ;\ d_{Vki}\ ) = max(\ d_{Pi} + \sum_{l=i+1}^{j} \frac{q_{\sigma_l}}{r}\ ;\ d_{Vki}\ )$
- $d_{Vkj} = \ d_{Pj} + \ \tau_{0,\sigma_{i+1}} + \sum_{l=i+1}^{j-1} \tau_{\sigma_l,\sigma_{l+1}} + \tau_{\sigma_j,0}$.

The max operator, which appears in the propagation rule, is explained in Fig. 5. In the first case, O_{j1} has a duration $\left(p_{j1}\right)$ greater than the duration between the end of the production operation O_{i1} at time d_{Pi} and the end of the previous sub-trip assigned to vehicle $V1$ (d_{V1i}). The finishing time of O_{i1} therefore corresponds to the starting time of O_{j1} and, consequently, $d_{Pj} = d_{Pi} + \sum_{l=i+1}^{j} \frac{q_{\sigma_l}}{r}$. In the other case, if O_{j1} has a duration $\left(p_{j1}\right)$ lower than the duration between the end of the production operation O_{i1} at time d_{Pi} and the end of the previous sub-trip assigned to vehicle $V1$ (d_{V1i}), then to be compliant with the no-wait constraint, the constraint $d_{Pj} = d_{V1i}$ must hold and the production planning will encompass a period of inactivity.

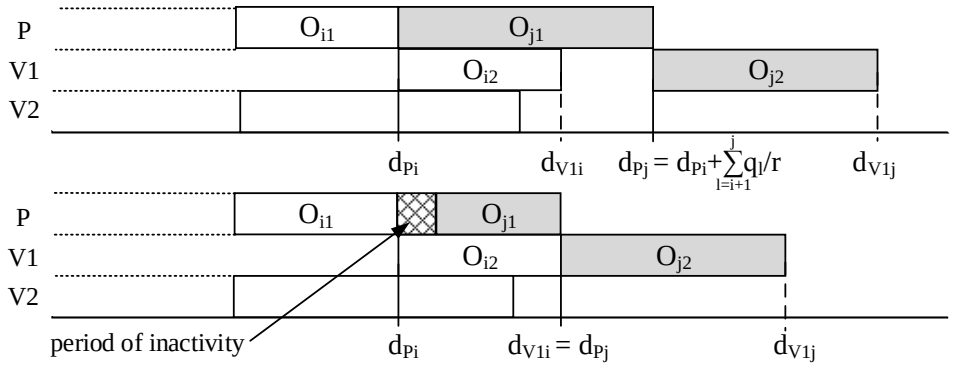

Figure 5. The two cases of label propagation in the Split procedure.

Since a large number of labels may be generated, dominance criterion is used to improve the running time, keeping only non-dominated labels on the node thanks to the following dominance propriety.

Dominance property

Label $L = (d_P,\ d_{V1}, \dots, d_{VN})$ dominates label $P = (d_P,\ d_{V1}, \dots, d_{VN})$ when the two labels are sorted and is therefore equal to $\left(d_{\sigma_1}^{\ p}, \dots, d_{\sigma_{N+1}}^{\ p}\right)$ and $\left(d_{\sigma_1}^{\ q}, \dots, d_{\sigma_{N+1}}^{\ q}\right)$, respectively, and all of the following conditions hold:

- $\exists i^* \in \{1, \dots, N+1\}, \mathrm{d}_{\sigma_{i^*}}^{\ p} < \mathrm{d}_{\sigma_{i^*}}^{\ q}$;
- $\forall j \in \{1, \dots, N+1\}, \mathrm{d}_{\sigma_j}^{\ p} \leq \mathrm{d}_{\sigma_{j^*}}^{\ q}$.

The dominance rule limits the number of labels stored at each node but several authors, including (Duhamel et al., 2012), have reported that a large number of labels could still be generated. Another time-saving approach consists in limiting the maximal number N_{max} of labels generated during the split process and the maximal number NB_{max} of labels stored at each node. This can reduce the CPU time, but discarding some labels may cause the algorithm to miss the optimal split. Such an approach offers a compromise between split quality and the number of labels kept on nodes and provides time-efficient sub-optimal split solutions.

The Split procedure is detailed in Algorithm 2 and uses low-level procedures to handle the label on the graph:

- `Propagation_label_to_vehicle(L,v,`i`)` propagates the label L using the vehicle v at node i and returns a new label;
- `CheckDomination(L, i, N`$_i$`)` performs the dominance check between the label L and all labels stored at node i, N_i. It returns 0 if L is dominated by at least one label from i. Thus, L must not be saved. It returns 1 if L dominates at least one label from i. It returns 2 otherwise;
- `InsertLabel(L, i, CD, N`$_i$`, NB`$_{max}$`)` attempts to add the label L at node i if $CD \in \{1,2\}$ and by considering that the number of labels N_i must not exceed the maximal number NB_{max}. If CD=1, all labels from i dominated by L are deleted. The list of labels is ordered by decreasing cost;
- `Extract_trip()` checks the shortest path into the graph for the last node to the first node, and returns the set of trips.

The Split algorithm (Algorithm 2) is composed of two parts: the initialisation part where local variables are initialised (lines 14-15) and a *for* loop from lines 16 to 52, which iterates for the ordered set of customers defined by the giant trip T. The *while* loop from lines 19 to 51 iterates and makes it possible to evaluate the partial sequence $(\sigma_{i+1}, \dots, \sigma_j)$. If $i = j$, a new sub-trip is created and the initial sub-trip costs are assigned (lines 22- 23) and are updated in lines 25-26.

The condition line 28 makes it possible to determine if the constraints hold and the loop is stopped early thanks to the condition on the Boolean condition (stop=true). Because a set of labels is saved at each node, the *for* loop of line 29 iterates over all labels of node *i* and addresses the label L_i^P.

Algorithm 2. Split

```
1.  procedure Split
2.  input parameters
3.    T: giant tour
4.  output parameters
5.    S: VRP-PTSPm solution
6.  global parameter
7.    Q : maximal vehicle weight capacity
1.    B : lifespan of the product
8.    q_i : total items ordered by customer i
9.    τ_ij : cost from customer i to j
10.   n : number of customers
11.   N : number of vehicles
12.   NB_max : maximal number of labels in each node
13.  begin
14.    L_0^1:=(0,0,…,0), S := ∅
15.    for i := 1 to n do NB_i := 0 endfor
16.    for j := 0 to n do
17.     j := i
18.     stop := false
19.     while (j < n and stop=false)
20.       customer := T_j
21.       if (j = i) then
22.         production_cost := d_customer/r
23.         transport_cost := c_depot,customer + c_customer,depot
24.       else
25.         production_cost += d_customer/r
26.         transport_cost += c_customer-1,customer +c_customer,depot −c_customer-1,depot
27.       endif
28.       if ((production_cost*r < Q) and (transport_cost -c_customer,depot ≤ B)) then
29.         for p := 1 to NB_i do
30.           if (d_Pi-1 = 0) then
31.             insertion of the label in first position with the first vehicle
32.           else
33.             v:=0
34.             do
35.               v:=v+1
36.               L:=Propagation_label_to_vehicle (L_i^P,v,i)
38.               if (NB_i=0)
39.                 insertion of L in first position
40.               else
41.                 CD:=CheckDomination(L, j, N_j)
42.                 call InsertLabel(L, j, CD, N_j, NB_max)
43.               endif
44.             while (v < N)
45.           endif
46.         endfor
47.       else
48.         stop := true
49.       endif
50.       j := j + 1
51.     endwhile
52.   endfor
53.   S := call Extract_trips ()  //save the best solution
54.   endif
55. end
```

A full example is given in detail on the Web page:
http://fc.isima.fr/~vinot/Research/PTSP_Results.html

3.4 Scheduling procedure

After the splitting procedure, each job is fully defined by an ordered sequence of customer demands that are assigned to a specific vehicle. The previous sections refer to a problem that can be defined as a two-stage hybrid flow shop with one machine at the first stage and N machines at the second stage. At the first stage, the term 'machine' is a general term that refers to the production (production machine) and to the transportation for the second stage (also referred to as transportation machine).

A job simultaneously models the production (the first operation on the production machine) and a sub-trip (the transportation operation on one transportation machine among the N machines available).

The problem is modelled as a disjunctive graph model first defined by Roy et al. (1964) using a directed graph $\mathrm{G} = (\mathrm{V}, \mathrm{A}, \mathrm{E_P}, \mathrm{E_T})$, where V represents the set of nodes that contains one element for each operation O_i, a source node 0 connected to the first operation of each job, and a sink node $*$ linked to the last operation of each job (Fig. 6). The set A represents the set of conjunctive arcs, $\mathrm{E_P}$ the set of pairs of disjunctive arcs between the production nodes, and $\mathrm{E_T}$ the set of pairs of disjunctive arcs between the transportation nodes.

With the specific graph characteristic, the classical disjunctive graph can be adapted using maximal time lags between the starting time of the production operation O_{i1} and the starting time of the transportation operation O_{i2}, which is consistent with the perishability constraints. The value of the maximal time lag defines the larger gap between the starting time of O_{i1} and the starting time of O_{i2}, and is defined by $-(B + p_{i1} - p'_{i2})$.

Conjunctive arcs are used to connect each pair of consecutive operations of the same job. Each pair of disjunctive arcs on the production connects operation O_{i1} to O_{j1} (belonging to different jobs), and has a duration p_{i1}. Each pair of disjunctive arcs on the transportation connects two operations O_{i2}, O_{j2} in this order belonging to different jobs that are to be processed on the same transportation machine (vehicle), and has a duration p_{i2} (Fig. 6). A feasible solution corresponds to an acyclic graph, and an evaluation procedure can be defined using a Bellman-like longest path algorithm to obtain the earliest starting time of each operation, including the makespan C_{max}.

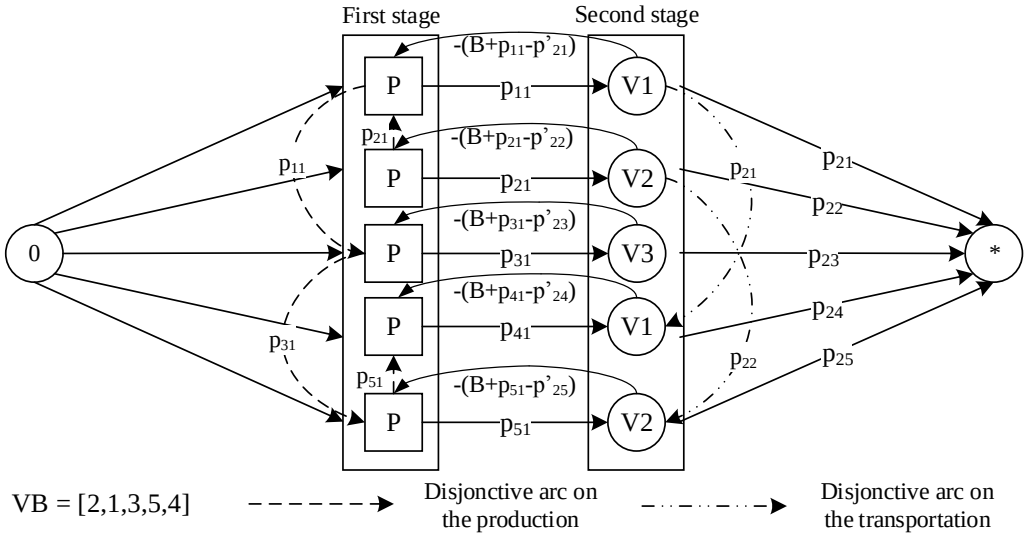

Figure 6. Example of the scheduling procedure with five sub-trips and V=3.

In Fig. 6, there are two disjunctive arcs on the transportation due to jobs using the same vehicle. For vehicle 1 assigned to jobs 1 and 4, the value on the disjunctive arc is equal to p_{21}. Figure 7 is a graphical representation of relations between both the earliest and finishing times

of operations with the lifespan constraint. The Bierwirth vector used in the scheduling procedure gives an order on the job using the sub-trips created by the splitting procedure. An efficient local search algorithm can be defined considering the critical path including the neighbourhood of Laarhoven et al. (1992) and Grabowski et al. (1986) with the introduction of blocks.

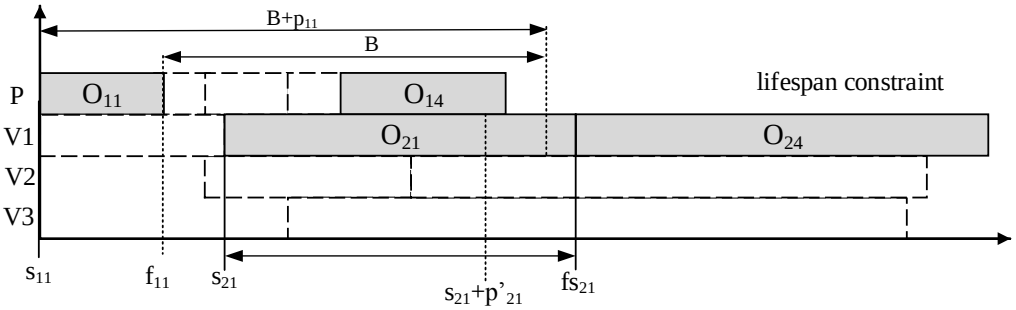

Figure 7. Representation of the constraint due to the disjunctive arcs of the transportation.

The neighbourhood of Laarhoven consists of swapping two consecutive operations assigned to the same machine along the critical path, leading to a modification of the machine disjunction. Due to the indirect representation of solutions, the operation swap is achieved in the Bierwirth vector by switching the two job numbers of the two operations in the disjunctive graph. The permutation of operations based on the block definition and located at the critical path is included in a depth first search local search investigates the move from the end of the critical path to the dummy node. If one permutation leads to a lower cost solution, the critical path exploration is restarted to the end of the graph. Note that due to computation time considerations, a maximal number of iterations *ns* is reached. A full example is given in detail on the Web page.

3.5 Hash function for PTSPm

To avoid premature convergence of the global algorithm and to stimulate the investigation of the part of the search space not previously investigated, it is necessary to use an efficient clone detection system. Previous research into scheduling/routing problems based on indirect representation approaches has proved that efficient clone detection must be defined considering the indirect representation or the solution. For the giant trip GT, the related hash function $H(t)$ is defined as follows: $H(t) = \sum_{i=1}^{n} GT[i] \times i \; mod \; K$. $GT[i]$ is the i^{th} customer in the giant trip and K is a constant. The value of K impacts the size of the map and the probability of collision (consequently, K must be as large as possible considering the memory available).

3.6 Lower bound on the makespan

Three PTSP lower bounds were introduced by Geismar et al. (2008), leading to a poor evaluation with multiple vehicles. Considering sub-trips with only one customer associated with immediately available vehicles, $N \geq n$, a lower bound can be defined using Jackson's algorithm (1955) that considers the one-machine scheduling problem with tails. To obtain the lower bound, the proposal is to extend Jackson's earliest due date rule. The lower bound based on the Jackson rule creates an optimal schedule that minimises the maximal lateness by ordering the jobs by non-decreasing due dates because the release dates are equal to 0.

Line 18 in Algorithm 3 makes it possible to define the sum of the processing times of previously scheduled jobs. At line 19, the release date of the job is updated as the maximum between the current release date and the sum of the processing time plus the due date of the job. These two lines do not appear in Jackson's original algorithm; they have been added to the algorithm in order to match the lower bound of the PTSPm.

Algorithm 3: A Jackson rule-based lower bound

```
1.  procedure A lower bound based on the Jackson Rule
2.  input parameters
3.    J: set of job/sub-trip
4.  global parameter
5.    r_j : release date of the job/sub-trip j
6.    p_j : processing time of the job/sub-trip j
7.    q : sum of the processing time already scheduled
8.    d_j : due date of the job/sub-trip j
9.    n : number of customers/sub-trip
10. output parameters
11.   C_max : lower bound
12.  begin
13.    L := {1, …, n}, q := 0
14.    if  (J ≠ ∅) then
15.      while (L ≠ ∅) do
16.       u := min_j{r_j}, j in L
17.       choose i in L such as r_i = u and d_i maximal
18.       q :=  q + p_i
19.       for all k in L do r_k:= max{r_k, q + d_i}, C_max := r_k
20.       L := L\{i}
21.      endwhile
22.    endif
23.  end
```

To apply Algorithm 3 on the PTSPm, a parallel should be drawn between the processing time and the production time of the PTSPm, as well as between the due date and the PTSPm transportation time. The lower bound is then given by C_{max} at the end of the algorithm.

4 Computational evaluation for the PTSP

The benchmark introduced by Geismar et al. (2008) is composed of three small-scale instances with 40 customers and three medium-scale instances with 50 customers, and customers' demands are uniformly distributed between 100 and 300. The locations of the customers in the three instances with 40 or 50 customers are randomly generated in the square of 200, 300 and 400 side lengths, which denotes 12 sets of parameters (SP is detailed in Table 5a). As a consequence, there are six datasets (DS is is detailed in Table 5b). For each instance, the value of three parameters is set: $r \in \{1,2,3\}$ is the production rate; $Q \in \{300,600\}$ is the truck capacity; and $B \in \{300,600\}$ is the product lifespan. We therefore have 72 resulting instances in the benchmark. Note that the distance is the transportation time between customers by considering the Euclidean distance and a vehicle speed of 1 unit of distance per unit of time.

Table 4 proposes speed factors that are used in the following comparative tables to ensure a fair comparative study and that were established by previous research articles including Dongarra, (2014), and on http://www.roylongbottom.org.uk/linpackresults.htm. Since the MIPS performance is not the only influence on the CPU time, Table 4 also provides the information available about the computer, the operating system and the language.

Table 4. Comparative performance of processors.

	(Geismar et al., 2008)	**(Karaoğlan et al., 2017)**	**Our framework**
Computer	Intel Pentium 4 3.2GHz	Intel Xeon 3.16GHz	Intel Core i7 3.4GHz
OS	Windows XP Pro.	Windows 7	Windows 7
Language	BASIC	C++ IBM ILOG CPLEX 12.6	Visual C++
MFlops	1573	1892	2671
Speed factor	1.0	1.2	1.7

Five replications (rep) are processed for each instance and Tables 6, 7 and 8 give the average of the best solutions found over the five replications with the average CPU time required for each run. The parameters used in the GRASP×ELS are given below. They remain unchanged for all instances and were obtained after preliminary experiments:

- np Number of GRASP/ELS/ neighbourhood iterations 150/30/15
- lr Number of local searches on the routing/scheduling 40/500
- NB_{max} Maximal number of labels per node 5

The following notations are used in the tables below:

C	Number of customers
S	Size of the square for the customer's location
Avg	An average value
%Gap	Percentage gap
N/A	Data not available
Nb. Opt	Number of instances where the solution has the same value as the lower bound. In this case, the solution obtained is an optimal solution of the PTSPm.
LB_i^k	Lower bound of the problem with $k \in SP$ and $i \in DS$
$h_{i,j}^k$	Best solution found for an instance with $(i;j;k) \in (DS;SP;[\![1,rep]\!])$
$tt_{i,j}^k$	Total CPU time in seconds coupled with $h_{i,j}^k$
$t_{i,j}^k$	CPU time in seconds to get $h_{i,j}^k$
$\overline{LB_*^k} = \underset{i \epsilon DS}{\text{avg}}\, LB_i^k$	Average lower bound for one $k \in SP$, over all $i \in DS$
$\overline{h_{*,*}^k} = \underset{i \epsilon DS, j=[\![1,rep]\!]}{\text{avg}}\, h_{i,j}^k$	Average best solution found for one $k \in SP$, over all $(i;j)/(i \in DS; j = [\![1,rep]\!])$
$\overline{tt_{*,*}^k} = \underset{i \epsilon DS, j=[\![1,rep]\!]}{\text{avg}}\, tt_{i,j}^k$	Average total CPU time for one $k \in SP$, over all $(i;j)/(i \in DS; j = [\![1,rep]\!])$
$\overline{t_{*,*}^k} = \underset{i \epsilon DS, j=[\![1,rep]\!]}{\text{avg}}\, t_{i,j}^k$	Average CPU time for one $k \in SP$, over all $(i;j)/(i \in DS; j = [\![1,rep]\!])$
$\overline{LB_*^*} = \underset{k \epsilon SP, i \epsilon DS}{\text{avg}}\, LB_i^k$	Average lower bound over all $(i;k)/(i \in DS; k \in SP)$
$\overline{h_{*,*}^*} = \underset{k \epsilon SP, i \epsilon DS, j=[\![1,rep]\!]}{\text{avg}}\, h_{i,j}^k$	Average best solution found over all $(i;j;k)/(i \in DS; j = [\![1,rep]\!]; k \in SP)$
$\overline{tt_{*,*}^*} = \underset{k \epsilon SP, i \epsilon DS, j=[\![1,rep]\!]}{\text{avg}}\, tt_{i,j}^k$	Average total CPU time over all $(i;j;k)/(i \in DS; j = [\![1,rep]\!]; k \in SP)$
$\overline{t_{*,*}^*} = \underset{k \epsilon SP, i \epsilon DS, j=[\![1,rep]\!]}{\text{avg}}\, t_{i,j}^k$	Average CPU time for all $(i;j;k)/(i \in DS; j = [\![1,rep]\!]; k \in SP)$
h_i^k	Best solution found among all replications, for one $(k, i) \in (SP,DS)$
tt_i^k	Total CPU time in seconds coupled with h_i^k
t_i^k	CPU time in seconds to obtain h_i^k
$\overline{h_*^*} = \underset{k \epsilon SP, i \epsilon DS}{\text{avg}}\, h_i^k$	Average of the h_i^k over the all datasets $i \epsilon DS$ and for all $k \in SP$
$\overline{tt_*^*} = \underset{k \epsilon SP, i \epsilon DS}{\text{avg}}\, tt_i^k$	Average of the tt_i^k over the all datasets $i \epsilon DS$ and for all $k \in SP$
$\overline{t_*^*} = \underset{k \epsilon SP, i \epsilon DS}{\text{avg}}\, t_i^k$	Average of the t_i^k over the all datasets $i \epsilon DS$ and for all $k \in SP$

Table 5a gives the definition of the 12 sets of parameters $SP = (r, Q, B)$, and Table 5b gives the six dataset definition $DS = (C, S)$. In Table 6, each value is an average value over the 12 sets of parameters ($SP = [\![1,12]\!]$), for the six datasets ($DS = [\![1.6]\!]$) and over the five replications for our proposal. Note that in Table 6, Table 7 and Table 8, the values reported by Geismar et al. (2008), Karaoğlan et al. (2017), and for our proposal are rounded off to the nearest integer value to ensure a fair comparative study.

Table 5. Instances characteristics

a. Definition of the 12 sets of parameters (SP).

SP	r	Q	B
1	1	300	300
2	2	300	300
3	3	300	300
4	1	300	600
5	2	300	600
6	3	300	600
7	1	600	300
8	2	600	300
9	3	600	300
10	1	600	600
11	2	600	600
12	3	600	600

b. Definition of the 6 datasets (DS).

DS	C	S
1	40	200
2	40	300
3	40	400
4	50	200
5	50	300
6	50	400

The genetic algorithm of Geismar et al. (2008) has an average time of 169 seconds, while the average computing time of our method is 141 seconds (average time scale of Table 6). The average computing time of the branch-and-cut of Karaoğlan et al. (2017) is significantly larger, with approximately 3249 seconds.

Table 6. Average results on the instances of Geismar et al. (2008) with five replications.

	$\overline{LB_*^*}$	(Geismar et al., 2008)			(Karaoğlan et al., 2017)			Our proposal					
		$\overline{h_{*,*}^*}$	$\overline{tt_{*,*}^*}$	$\overline{t_{*,*}^*}$	$\overline{h_{*,*}^*}$	$\overline{tt_{*,*}^*}$	$\overline{t_{*,*}^*}$	$\overline{h_{*,*}^*}$	$\overline{tt_{*,*}^*}$	$\overline{t_{*,*}^*}$	$\overline{h_*^*}$	$\overline{tt_*^*}$	$\overline{t_*^*}$
Avg.	5383	8045			7891			7884			7876		
Avg.time			169	N/A		2708	N/A		82.7	40.7		81.1	42.7
Avg. time scale			169	N/A		3249	N/A		141	69		138	73
Nb. Opt		16/72			13/72			16/72			18/72		

Our method provides an average value $\overline{h_{*,*}^*}$ of about 7884, which is better than Geismar et al. (2008) (average value of approximately 8045) and Karaoğlan et al. (2017) (average value of approximately 7891). The analysis of the full solution set (available on the Web page) makes it possible to confirm that the GRASP×ELS provides better solutions in 31 instances, equal solutions in 35 instances and worse solutions in only six instances. In general, when the solutions are improved, they are improved by approximately 4.7%, whereas when they are worsened, they are worsened by approximately 0.04%.

Table 7. Average results on the instances of Geismar et al. (2008).

$k \in SP$	$\overline{LB_*^k}$	(Geismar et al., 2008)		(Karaoğlan et al., 2017)		Our proposal	
		$\overline{h_{*,*}^k}$	%Gap	$\overline{h_{*,*}^k}$	%Gap	$\overline{h_{*,*}^k}$	%Gap
1	8781	10040	12.5	10049	12.6	10039	12.5
2	4412	9157	51.8	9179	51.9	9148	51.7
3	2956	9125	67.6	9153	67.7	9118	67.5
4	8781	10041	12.5	10048	12.6	10026	12.4
5	4412	9171	51.9	9180	51.9	9151	51.7
6	2956	9153	67.7	9152	67.7	9122	67.6
7	8781	8781	0.0	8782	0.0	8782	0.0
8	4412	5744	23.1	5346	17.5	5347	17.5
9	2956	5421	45.4	4887	39.5	4919	39.9
10	8781	8781	0.0	8782	0.0	8781	0.0
11	4412	5724	22.9	5276	16.4	5300	16.7
12	2956	5403	45.2	4861	39.2	4875	39.3
Avg.	5383	8045	33.4	7891	31.4	7884	31.4
Nb. Opt		16/72		13/72		16/72	

The efficiency of the GRASP×ELS algorithm is more significant for the instances in which $Q = 600$ (SP from 7 to 12) particularly for instances with the sets of parameters $SP = 9$ and

$SP = 12$ that have a high production rate (Table 7). For instances from $SP = 7$ to $SP = 12$, the transportation time dominates the production time, as underlined by Geismar et al. (2008). Therefore, improvements on routing phase lead to better results for these instances. In Table 7, with $SP = 12$, the gap has been significantly reduced from 45.2% (Geismar et al., 2008) to 39.3% for our proposal. With the method used by Karaoğlan et al. (2017), the gap has also been reduced to 39.2%.

For the PTSP, our method provides 18 optimal solutions vs. 16 optimal solutions for the dedicated method of Geismar et al. (2008), as illustrated in Table 7. The method also outperforms the method of Karaoğlan et al. (2017), providing 18 solutions that reached the lower bound vs. 13 solutions for Karaoğlan et al. (2017).

5 Computational evaluation for the PTSPm

5.1 Benchmarks for the PTSPm

The instances of Geismar et al. (2008) are extended to tackle up to six vehicles and to evaluate the impact of routing with regard to the scheduling solutions (Table 8). It can be observed that by increasing the number of vehicles from one to two, the average makespan for all the instances is reduced from 25% (decreasing from 7,884 to 5,820) and then decreased by 6% (from 5,820 to 5,478) when increasing the number of vehicles from two to three.

Table 8. Average results with the GRASP×ELS for the PTSPm.

$k \in SP$	$\overline{LB_*^k}$	2 vehicles			3 vehicles			4 vehicles			5 vehicles			6 vehicles		
		$\overline{h_{*,*}^k}$	$\overline{tt_{*,*}^k}$	$\overline{t_{*,*}^k}$	$\overline{h_{*,*}^k}$	$\overline{tt_{*,*}^k}$	$\overline{t_{*,*}^k}$	$\overline{h_{*,*}^k}$	$\overline{tt_{*,*}^k}$	$\overline{t_{*,*}^k}$	$\overline{h_{*,*}^k}$	$\overline{tt_{*,*}^k}$	$\overline{t_{*,*}^k}$	$\overline{h_{*,*}^k}$	$\overline{tt_{*,*}^k}$	$\overline{t_{*,*}^k}$
1	8781	8781	0	0	8781	0	0	8781	0	0	8781	0	0	8781	1	1
2	4412	5114	38	17	4452	23	10	4412	6	3	4412	1	1	4412	2	2
3	2956	4658	51	25	3461	41	19	3061	37	22	2968	30	15	2956	16	14
4	8781	8781	1	1	8781	1	1	8781	1	1	8781	1	1	8781	1	1
5	4412	5116	40	22	4435	27	8	4412	2	2	4412	2	2	4412	3	3
6	2956	4674	53	21	3467	46	20	3059	32	18	2969	36	15	2956	6	6
7	8781	8781	3	3	8781	2	2	8781	2	2	8781	2	2	8781	2	2
8	4412	4426	70	41	4419	50	28	4413	49	31	4412	44	25	4412	47	39
9	2956	3175	142	86	2988	131	56	2973	103	54	2968	106	51	2968	108	58
10	8781	8781	2	2	8781	1	1	8781	1	1	8781	1	1	8781	1	1
11	4412	4422	54	23	4412	52	39	4414	57	27	4414	61	28	4414	62	29
12	2956	3137	151	93	2979	123	41	2969	115	52	2968	133	66	2969	136	49
Avg.	5383	5820			5478			5403			5387			5385		
Avg.time			50	28		41	19		34	18		35	17		32	17
Nb. Opt		41/72			53/72			63/72			63/72			66/72		

After intensive numerical experiments, we found that a fleet of six vehicles is sufficient to reach the lower bound of each instance with the strongest constraints, $r \in \{1,2,3\}$, $Q = 300$ and $B = 300$. The instances of Geismar et al. (2008) and the details of our solutions are available on the Web page. A careful analysis of the ratio between the average customer demand and the vehicle capacity remains constant (approximately 0.3) for all instances. Consequently, the average number of customers per sub-trip remains quite low (approximately two on average).

5.2 Proposal of new large-scale instances with multiple vehicles

A new set of instances with a broad range of parameters is defined considering:

- customer distribution to model both urban areas and rural surroundings;
- random depot node location (centered, peripheral location, etc.);
- large vehicle capacities and, consequently, larger sub-trips.

Each instance is characterised by:

- the number of customers that follows a discrete uniform distribution $U(50,100)$ for the large instances or $U(100,200)$ for the very large instances;
- an interest area that contains all the customers and the plant. The length X and the width Y of this area can vary between 100 and 500, which represents 25 possibilities (100×100, 100×200, …, 500×400 and 500×500). This area is centered in (0,0) and divided into four sectors $[-X, 0] \times [0, Y]$, $[0, X] \times [0, Y]$, $[0, X] \times [-Y, 0]$ and $[-X, 0] \times [-Y, 0]$;
- the number of customer centers $c = \{1,2,3\}$.

Moreover, the following parameters are necessary to define an instance:

- the capacity of the trucks is equal to 1000;
- the demand of each customer i, $d_i = \{50, 100, 200, 300, 500\}$ with the associated probabilities $\{0.1, 0.2, 0.4, 0.2, 0.1\}$;
- the position of each customer is uniformly distributed in one customer center;
- the position of each customer center is randomly selected among the sector centers with a variation of $\pm 10\%$;
- the size of each customer centre is equal to the minimal distance between the customer centre and the edge of the interest area with a variation of $\pm 20\%$;
- the position of the depot has a probability of 0.7 to be in the centre of one sector centres with a variation of $\pm 10\%$, and 0.3 to be centred in the interest area with a variation of $\pm 10\%$.

With all these parameters, we have 150 instances. An instance is caracterised by a triplet (x, z, c) with: $x \in \{L, VL\}$ to indicate if the size of the instance is large L or very lage VL, $z \in [\![1,25]\!]$ to define the size of the interest area, and $c = \{1,2,3\}$ to give the number of customer centres.

5.3 Computational results on the new instances

Five replications are achieved for each instance and Tables 9 gives the average of the best solutions found over the five replications and on the three customers centers (column $\overline{h_{*,*}}$) with the average total CPU time (column $\overline{tt_{*,*}}$) and the average time required to find the best solution (column $\overline{t_{*,*}}$). Note that the lower bound ($\overline{LB_*}$ column) refers to the lower bound introduced in Section 3.6. The parameter setting remains identical for all instances and was obtained after preliminary experiments:

- np/ne Number of GRASP/ELS/ neighbourhood iterations 50/10/5
- lr/ns Number of local searches on the routing/scheduling 40/500
- NB_{max} Maximal number of labels per node 5

The GRASP×ELS on the $(L,*,*)$ instances has an average time ranging from 1.13 to 0.1 seconds, depending on the number of vehicles, whereas the average computing time on the $(VL,*,*)$ instances is equal to 6.63 seconds for one vehicle and 0.86 seconds for two vehicles (Tables 9a and 9b). The computing time has thus been multiplied by six between the large and the very large instances. Moreover, on average, the gap between our results and the lower bound is very small, with 0.09% on large instances and 0.03% on very large instances for one vehicle. With two vehicles, the gap between our results and the lower bound is even smaller, with 0.005% on large instances and 0.003% on very large instances

The analysis of trips makes it possible to prove that the trip design is linked to the sub-graph defined by the densities of nodes, referred to as customer centres. All the trips and a graphical representation for instance 1 are available on the Web site.

Table 9. Results on the large and very large-scale instances with one or two vehicle.

a. Large instances (***L***,∗,∗)

		1 vehicle			2 vehicles		
Inst.	$\overline{\mathrm{LB}_*}$	$\overline{h_{*,*}}$	$\overline{tt_{*,*}}$	$\overline{t_{*,*}}$	$\overline{h_{*,*}}$	$\overline{tt_{*,*}}$	$\overline{t_{*,*}}$
(L,1,∗)	15119	15119	0.01	0.00	15119	0.00	0.00
(L,2,∗)	17572	17572	0.00	0.00	17572	0.01	0.00
(L,3,∗)	15929	15929	0.01	0.01	15929	0.00	0.00
(L,4,∗)	12396	12437	3.63	0.03	12396	0.00	0.00
(L,5,∗)	17458	17459	2.06	0.01	17458	0.00	0.00
(L,6,∗)	15173	15173	0.01	0.00	15173	0.00	0.00
(L,7,∗)	15985	15985	0.00	0.00	15985	0.00	0.00
(L,8,∗)	15851	15857	1.57	0.00	15851	0.00	0.00
(L,9,∗)	19815	19815	0.01	0.00	19815	0.00	0.00
(L,10,∗)	16333	16346	3.36	0.04	16333	0.00	0.00
(L,11,∗)	17774	17774	0.00	0.00	17774	0.00	0.00
(L,12,∗)	15607	15608	1.32	0.03	15607	0.01	0.00
(L,13,∗)	15235	15331	2.53	0.28	15235	0.03	0.03
(L,14,∗)	13497	13501	1.28	0.00	13497	0.01	0.00
(L,15,∗)	15425	15431	1.16	0.01	15425	0.01	0.00
(L,16,∗)	13687	13688	1.04	0.00	13687	0.00	0.00
(L,17,∗)	14273	14273	0.06	0.06	14273	0.00	0.00
(L,18,∗)	14449	14452	1.30	0.00	14449	0.00	0.00
(L,19,∗)	16579	16579	0.00	0.00	16579	0.01	0.01
(L,20,∗)	14415	14442	2.32	0.20	14415	0.00	0.00
(L,21,∗)	17012	17012	0.00	0.00	17012	0.00	0.00
(L,22,∗)	16300	16312	1.66	0.33	16300	0.01	0.00
(L,23,∗)	14421	14530	1.48	0.00	14439	2.48	0.06
(L,24,∗)	17364	17364	0.00	0.00	17364	0.01	0.00
(L,25,∗)	**14167**	**14188**	**3.33**	**0.74**	**14167**	**0.01**	**0.00**
Avg.	15673	15687			15674	0.10	0.00
Gap LB		0.09%			0.005%		
Avg.time			1.13	0.07		0.10	0.00
Nb. Opt		55/75			74/75		

b. Very large instances (***VL***,∗,∗)

		1 vehicle			2 vehicles		
Inst.	$\overline{\mathrm{LB}_*}$	$\overline{h_{*,*}}$	$\overline{tt_{*,*}}$	$\overline{t_{*,*}}$	$\overline{h_{*,*}}$	$\overline{tt_{*,*}}$	$\overline{t_{*,*}}$
(VL,1,∗)	27859	27859	0.02	0.01	27859	0.02	0.01
(VL,2,∗)	30401	30401	0.02	0.01	30401	0.02	0.01
(VL,3,∗)	37341	37341	0.03	0.03	37341	0.03	0.03
(VL,4,∗)	34229	34229	0.03	0.02	34229	0.02	0.01
(VL,5,∗)	23003	23069	8.64	6.57	23018	8.51	0.02
(VL,6,∗)	34625	34625	0.02	0.02	34625	0.02	0.02
(VL,7,∗)	34923	34923	0.03	0.02	34923	0.02	0.02
(VL,8,∗)	32175	32175	0.02	0.02	32175	0.02	0.02
(VL,9,∗)	35041	35046	13.59	0.02	35041	0.03	0.02
(VL,10,∗)	34739	34751	27.00	0.03	34739	0.02	0.02
(VL,11,∗)	31357	31357	0.02	0.02	31357	0.02	0.02
(VL,12,∗)	32843	32843	0.03	0.02	32843	0.02	0.02
(VL,13,∗)	34371	34371	0.02	0.02	34371	0.02	0.02
(VL,14,∗)	32165	32179	32.41	0.02	32165	0.02	0.02
(VL,15,∗)	31361	31361	8.27	0.34	31361	0.02	0.02
(VL,16,∗)	30529	30529	0.02	0.02	30529	0.02	0.02
(VL,17,∗)	30507	30507	0.02	0.01	30507	0.02	0.01
(VL,18,∗)	35015	35015	0.03	0.02	35015	0.03	0.02
(VL,19,∗)	33901	33901	0.02	0.02	33901	0.02	0.02
(VL,20,∗)	29647	29651	6.95	0.58	29647	0.02	0.02
(VL,21,∗)	35893	35893	0.03	0.03	35893	0.03	0.02
(VL,22,∗)	33157	33175	24.80	0.29	33157	0.02	0.02
(VL,23,∗)	31203	31229	11.92	0.02	31203	0.02	0.02
(VL,24,∗)	32379	32385	8.58	1.26	32379	0.02	0.02
(VL,25,∗)	33524	33576	23.19	11.86	33531	12.60	0.28
Avg.	32487	32496			32488		
Gap LB		0.03%			0.003%		
Avg.time			6.63	0.85		0.86	0.03
Nb. Opt		60/75			72/75		

6 Concluding remarks

Proper integration of production planning and routing is the key feature in a supply chain since the coordination of these two functions has a significant impact on the customer service level. This paper addresses the PTSP with multiple vehicles (PTSPm) in order to extend the PTSP with a single vehicle in Geismar et al. (2008) and can be efficiently used to compute coordinated solution in supply chain. This approach proves it is possible to solve the two problems in a coordinate way and permits to obtain better solution than classical approach where the two problems are solved sequentially.

The framework we propose takes advantage of an indirect representation of the solutions using a split-based approach with search space alternation between TSP solutions, VRP solutions and PTSPm solutions. This indirect approach is one of the key features of the proposal. The framework efficiency leads to a special disjunctive graph for the trips and two kinds of disjunctive arcs due to a single production facility and the vehicles. The method has proven to be efficient in the one-vehicle instances, providing better solutions than Geismar et al. (2008) and Karaoğlan et al. (2017), with shorter computation times. The method also generates new solutions for the multi-vehicle extension. In order to ensure fair comparative studies, a new set of instances has been introduced.

Our research is now directed towards a bi-objective resolution where a second criterion could be introduced for the quality of service, which could be defined as the delay between the arrival time of one vehicle at a customer node and the upper bound of the time window. This second criterion should be relevant for the quality of service, provided that there is a large enough difference between the delivery date and the expiration date for customer products modelled by the perishability constraint.

Acknowledgements: This work was carried out and funded within the framework of the ATHENA project (Ref.: ANR-13-BS02-0006) and Labex MS2T. It was supported by the French government through the “Investments for the Future" program managed by the French National Research Agency (Ref.: ANR-11-IDEX-0004-02).

References:

Armstrong R, Gao S, Lei L. (2008). A zero-inventory production and distribution problem with a fixed customer sequence, Annals of Operations Research, 159(1): 395-414.

Beasley J.E. (1983). Route-first cluster-second methods for vehicle routing, Omega; 11: 403-408.

Belo-Filho M.A.F, Amorim P, Almada-Lobo B. (2015). An adaptive large neighbourhood search for the operational integrated production and distribution problem of perishable products, International Journal of Production Research, 53(20), 6040-6058.

Bierwirth C. (1995). A generalized permutation approach to job-shop scheduling with genetic algorithms, OR Spektrum; 17: 87-92.

Chen Z.L. (2010). Integrated Production and Outbound Distribution Scheduling: Review and Extensions, Operations Research, 58(1): 130-148.

Cheng R, Gen M, Tsujimura Y. (1996). A tutorial survey of job-shop scheduling problems using genetic algorithms – I representation Computers and industrial engineering; 30: 983-997.

Desrochers M. (1988). An Algorithm for the Shortest Path Problem with Resource Constraints. Research Report G-88 127; GERAD, Montreal, Canada.

Devapriya P, Ferrell W, Geismar H.N. (2006). Optimal fleet size of an integrated production and distribution scheduling problem for a perishable product, IIE Annual Conference and Exposition, Orlando, Florida.

Devapriya P, Ferrell W, Geismar H.N. (2017). Integrated production and distribution scheduling with a perishable product, European Journal of Operational Research, 259(3); 906-916.

Dongarra J. (2014). Performance of various computers using standard linear equations software. Report CS-89-85, University of Manchester.

Duhamel C, Lacomme P, Prodhon C. (2012). A hybrid evolutionary local search with depth first search split procedure for the heterogeneous vehicle routing problems, Engineering Applications of Artificial Intelligence, 25(2): 345-358.

Feo T.A, Resende M.G.C. (1995). Greedy Randomized Adaptive Search Procedures, Journal of Global Optimization, 6(2): 109-133.

Fondrevelle J, Oulamara A, Portmann M-C. (2004). Permutation Flowshop Scheduling Problems with Maximal and Minimal Time Lags. Computers & Operations Research, Elsevier, 33(6):1540-1556.

Geismar H.N, Laporte G, Lei L, Sriskandarajah C. (2008). The Integrated Production and Transportation Scheduling problem for a Product with a Short Lifespan, INFORMS Journal on Computing, 20(1): 21-33.

Gilmore P, Gomory R. (1964). Sequencing a one-state variable machine: A solvable case of the traveling salesman problem. Operations Research, 12:655-679.

Grabowski J, Nowicki E, Zdrzalka S. (1986). A block approach for single machine scheduling with release dates and due dates. European Journal of Operational Research, 26: 278-285.

Jackson R. (1955). Scheduling a production line to minimize maximum tardiness. Technical Report 43, Management research project, University of California, Los Angeles.

Johnson S.M. (1954). Optimal Two and Three Stage Production Schedules with Set-Up Times, Included, Naval Research Logistics, 1(1), 61-68.

Karaoğlan I, Erhan Kesen S. (2017). The coordinated production and transportation scheduling problem with a time-sensitive product: a branch-and-cut algorithm, International Journal of Production Research, 55(2); 536-557.

Lacomme P, Prins C, Ramdane-Chérif W. (2001). Competitive Memetic Algorithms for the Capacitated Arc Routing Problem and its Extensions. Lecture Notes in Computer Science; 2037: 473-483.

Lourenço H, Martin O, Stützle T. (2003). Iterated local search, Handbook of Metaheuristics; 321-353.

Moons S., K. Ramazkezrs, A. Caris and Y. Arda. Integrated production scheduling and vehicle routing decisions at the operational decision level: A review and discussion. Computers and Industrial Engineering. 2017; 104: 224-245.

Prins C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. Computers & Operations Research, 31(12): 1985-2002.

Prins C. (2009). A GRASP × Evolutionary Local Search Hybrid for the Vehicle Routing Problem. Bio-inspired Algorithms for the Vehicle Routing Problem; 161: 35-53.

Prins C, Lacomme P, Prodhon C. (2014). Order-first split-second methods for vehicle routing problems: A review. Transportation Research Part C: Emerging Technologies; 40:179-200.

Rivera J.L, Lallmahomed A. (2016). Environmental implications of planned obsolescence and product lifetime: a literature review, International Journal of Sustainable Engineering, 9(2), 119-129.

Roscoe S, Baker P. (2014). Supply chain segmentation in the sporting goods industry, International Journal of Logistics: Research and Applications, 17(2), 136-155.

Roy B, Sussmann B. (1964). Les problèmes d'ordonnancement avec contraintes disjonctives; In: Note DS N°9 bis, SEMA, Paris, France.

Sarmiento A.M, Nagi R. (1999). A review of integrated analysis of production-distribution systems; IIE Transaction. 31, 1061-1074.

Van Laarhoven P.J.M, Aarts E.H.L, Lenstra J.K. (1992). Jobshop scheduling by simulated annealing. Operations Research, 40: 113-125.

Wolf S, Merz P. (2007). Evolutionary local search for the super-peer selection problem and the p-hub median problem, Lecture notes in computer science; 4771: 1-15.