Elsevier Editorial System(tm) for Computers & Industrial Engineering

Manuscript Draft

Manuscript Number: CAIE-D-19-00406R3

Title: A production and distribution framework: manufacturer dominates

Article Type: Research Paper

Keywords: supply chain; multiple agents; manufacturer; 3PL provider; scheduling; vehicle routing

Corresponding Author: Professor Jean Charles Billaut,

Corresponding Author's Institution:

First Author: Hugo Chevroton

Order of Authors: Hugo Chevroton; Sonja U.K. Rohmer; Jean Charles Billaut

Abstract: This paper presents a two-stage supply chain problem at the operational level involving a manufacturer and a third-party logistics provider (3PL provider). The integrated problem consists of two subproblems: an m-machine permutation flow shop with inventory considerations for the manufacturer and a routing problem for the 3PL provider. Both agents face tardiness penalty costs in case of late delivery to the customer locations. Assuming imperfect information sharing between the two agents and independence in their own decisions, we investigate the scenario in which the manufacturer dominates the negotiation.

In this scenario, the manufacturer imposes the number and composition of vehicles as well as the vehicle departure dates on the 3PL provider. However, lacking the information regarding the actual delivery times he is forced to estimate delivery times at the customer locations in the context of his planning. Following the planning of the manufacturer, the 3PL provider decides on the optimal routing of the vehicles. Both agents aim to minimise their total costs. After formulating the problems as a mixed integer linear program, two metaheuristic algorithms are proposed to solve the scenario. The performance of the heuristics is evaluated on a set of randomly generated instances. The results show that the genetic algorithm is the most performing method.

Cover Letter

Jean-Charles Billaut
Université de Tours
LIFAT, EA 6300, ERL CNRS ROOT 6305
Tours, France

Dear Editor,

You will find attached our new revised submission

“A framework for production and outbound distribution: manufacturer dominates”

by H. Chevroton, S. Rohmer and myself.

for revision in Computers And Industrial Engineering journal.

Sincerely yours,

In Tours, 31 of October 2020

Jean-Charles Billaut

A production and distribution framework: Manufacturer dominates

Hugo Chevroton
Université de Tours
LIFAT, EA 6300, ERL CNRS ROOT 6305
Tours, France
hugo.chevroton@etu.univ-tours.fr

Sonja U.K. Rohmer
Operations Research and Logistics
Wageningen University
Wageningen, Netherlands
sukrohmer@outlook.com

Jean-Charles Billaut
Université de Tours
LIFAT, EA 6300, ERL CNRS ROOT 6305
Tours, France
jean-charles.billaut@univ-tours.fr

January 2, 2020

HIGHLIGHTS

A framework for production and outbound distribution: manufacturer dominates

- The field of the study is the optimization of the supply chain organization
- We consider a cooperation model between a manufacturer and a 3PL provider
- We want to minimize inventory costs, transportation costs and tardiness penalty costs
- We propose MILP models and three metaheuristic resolution methods and compare them

Response to referees

We would like to thank the reviewers once again for their comments. As you will see, the paper has received a very major revision, with important changes in Section 6 (results).

Below we provide our response to their remarks.

Reviewers' comments:

Reviewer #2: Review on CAIE-D-19-00406.R2

A framework for production and outbound distribution: manufacturer dominates

Despite the authors responded to some comments, surprisingly it sounds that they do not intend or cannot apply the left comments. Moreover, the contribution of the paper is not sufficient for CAIE. Accordingly, I recommend to reject the paper, since the authors couldn't respond to the most important and challenging queries.

> We are surprised to receive this feedback at this stage of the review process, as we have provided detailed responses to all of the reviewer's remarks in each of the previous stages. Moreover, we have updated the manuscript throughout this process to address the reviewer's concerns and as a result we think that the paper improved significantly.

Minor point:

1. Some of Tables have not yet been cited in the body of manuscript.

> Thank you. We have addressed this issue and ensured that all tables and figures are now cited in the body of the manuscript.

Some major points:

2. All of used solution approaches, i.e., GA, TS and GRASP are classic methods which to some extent do not have any new contribution to the literature. The authors should compare the obtained results of the employed methods with at least one or two new-born meta-heuristic algorithms. This makes the reader confident regarding the validity of the obtained results.

Once again, we would like to highlight that the main contribution of this paper lies in introducing a novel problem of interest, which we formulate mathematically and solve by applying a number of classic methods, comparing them with regards to their performance. We have replied to this remark in previous rounds, however the reviewer failed to provide us with further insights and furthermore does not seem to be aware that the TS is no longer included in the paper, despite several indications from our side. In the course of the review process, we have moreover improved the applied methods, providing more evidence for the validity of the obtained-results. We do not see how this could be further improved by adding another solution method distorting the focus of this paper.

3. The boxplots provided is not my answer. The authors should conduct appropriate extensive statistical analysis to see whether the employed algorithms have significantly different performances or not. Without doing this, one cannot rely to the obtained results.

> In the process of improving our applied methods we have added further statistical analysis to the manuscript. The analysis, thus, now shows more clearly the differences between the applied methods. We hope this satisfies the reviewer and provides more confidence with regards to the obtained results.

Reviewer #3: I am satisfied with the corrections and queries answered and the manuscript can be accepted.

> We would like to thank the reviewer for the involvement and valuable remarks throughout this review process.

***Credit Author Statement**

H. Chevroton: Methodology, Software, validation, Investigation

S. Rohmer: Conceptualization, Writing- Original draft preparation and revised version

J-C. Billaut: Supervision, Writing - Reviewing and Editing, Project administration

A production and distribution framework: Manufacturer dominates

Hugo Chevroton *[1], Sonja Rohmer †[1], and Jean-Charles Billaut ‡[1]

[1]Université de Tours, LIFAT, EA 6300, ERL CNRS ROOT 6305, Tours, France

October 31, 2020

Abstract

This paper presents a two-stage supply chain problem at the operational level involving a manufacturer and a third-party logistics provider (3PL provider). The integrated problem consists of two subproblems: An m-machine permutation flow shop with inventory considerations for the manufacturer and a routing problem for the 3PL provider. Both agents face tardiness penalty costs in case of late delivery to the customer locations. Assuming imperfect information sharing between the two agents and independence in their own decisions, we investigate the scenario in which the manufacturer dominates the negotiation.

In this scenario, the manufacturer imposes the number and composition of vehicles as well as the vehicle departure dates on the 3PL provider. However, lacking the information regarding the actual delivery times he is forced to estimate delivery times at the customer locations in the context of his planning. Following the planning of the manufacturer, the 3PL provider decides on the optimal routing of the vehicles. Both agents aim to minimise their total costs. After formulating the problems as a mixed integer linear program, two metaheuristic algorithms are proposed to solve the scenario. The performance of the heuristics is evaluated on a set of randomly generated instances. The results show that the genetic algorithm is the most performing method.

Keywords: supply chain, multiple agents, manufacturer, 3PL provider, scheduling, vehicle routing, cooperation

*hugo.chevroton@etu.univ-tours.fr †sukrohmer@outlook.com ‡jean-charles.billaut@univ-tours.fr

1 Introduction

Production and distribution planning presents an integral part of supply chain management and as such often contributes significantly to the overall cost of a company. Efficient planning of these activities can reduce costs and help companies to stay competitive in today's business environment. Scheduling decisions at the manufacturer and the routing of vehicles to the customer locations play an important role in this context. Looking at these decisions in an integrated framework allows companies to streamline their operations, reduce storage times and improve service quality as well as customer satisfaction.

The integration of these decision problems is, however, often complicated through the presence of different actors in the chain, specialising in only one type of activity. In this environment, where different agents work together, but act as individual entities serving there own interests, not all the information may be shared, or be available to all actors at all times. Given this lack of information, actors may be forced to make predictions about the other actor's planning decisions, in order to make decisions regarding their own planning activities. Moreover, depending on the existing power balance, one of the actors may be able to impose certain decisions, representing their own interests, and through this restrict the available choices and associated savings potential of the other actor.

Specific service requirements in the form of predefined delivery dates, due to for example limited shelf-life of products, further complicate these problems and can lead to significant additional costs. Such penalty costs for the late delivery of products are imaginable in different ways and can further impact the dynamics between agents. Efficient production and distribution planning that takes these dynamics into account is thus essential to keep cost low and ensure smooth co-operation between all agents.

The remainder of the paper is organised as follows. Section 2 gives an overview of the related literature, before Section 3 describes the problem in more detail, highlighting the interest of the model presented in this paper. Section 4 introduces the notations and the mathematical formulation of the problem faced by each of the two agents. The two resolution methods applied to the problem are presented in Section 5. Section 6 presents the computational results for a set of randomly generated instances and Section 7 provides a general discussion and conclusion to the paper and possible future research directions.

2 Literature review

Aspects related to the individual problems of production and distribution planning have been studied extensively in the scientific literature and are not reported here. Studies on the integrated problem are, however, still relatively new and differ depending on the structure of the integration, i.e. the position of the transportation phase in the system. According to [Chen, 2010] three different kinds of integration can be distinguished: production with inbound transportation, production with outbound transportation and production with both in- and outbound transportation (as in [Agnetis et al., 2014], [Koç and Sabuncuoglu, 2017], [Marandani and Ghomi, 2019] and [Cheng et al., 2019]). Moreover, different types of production environments have been considered in the literature. The most common are one machine environments ([Devapriya et al., 2017], [Li et al., 2005]) or a set of parallel machines ([Han et al., 2019], [Cakici et al., 2014], [Kergosien et al., 2017]). Integrated problems considering a shop production environment are less common in the literature. In [Scholz-Reiter et al., 2010] the authors present a Mixed Integer Linear Programming (MILP) formulation assuming a flow shop environment with inventory cost but do not propose any resolution approach to solve the problem for larger instances. A bi-objective problem with flow shop production has been addressed in [Hassanzadeh et al., 2016], where the authors consider that jobs are delivered per batch immediately after production taking into account a fixed batch cost. The problem is solved by a multi-objective Particle Swarm Optimization algorithm (PSO) combined with a heuristic mutation operator.

Focusing on the distribution stage, two main categories of problems can be identified: direct delivery and vehicle routing problems. Direct delivery problems serve a customer or a set of customers considering a constant delivery time and a constant delivery cost. These problems have been studied for example in [Han et al., 2019] and [Chang et al., 2014]. In [Moons et al., 2017] the authors present an overview of integrated problems that model the distribution stage as a vehicle routing problem, where several customer requests can be served consecutively by a single vehicle. Examples of such problems can be found in [Lacomme et al., 2018], [Tavares-Neto and Seido, 2019] and [Farahani et al., 2012]. As in our study, [Armstrong et al., 2008] consider a problem where the delivery step consists of a single trip with strict routing order (i.e. the delivery order is given). Product lifespan and time window constraints are considered and the objective function is to maximise the total number of jobs delivered on time. A Branch-and-Bound algorithm (B&B) is proposed to solve the problem. In [Viergutz and Knust, 2014], the authors consider the same problem and pro-

pose an updated version of the B&B algorithm. Several papers in the literature introduce inventory considerations in scheduling problems. These papers are in the field of Economic Lot Scheduling problems [Santander-Mercado and Jubiz-Diaz, 2016]. In this category of problems, the determination of the economic lot sizing is part of the problem. In [Ouenniche and Boctor, 2001] the authors consider a multi-product lot sizing and scheduling problem in a flow shop environment. The model to compute the inventory cost is the same as in this paper. Another related problem is the Production Routing Problem (PRP) (see [Absi and Archetti, 2018], [Neves-Moreira et al., 2018]), where one or more products are produced in a given time period, considering a set up cost, followed by a routing phase to deliver the produced goods to the customer locations. To reduce the associated set-up and routing costs, inventory is considered at the production site or at the individual customer locations. In [Absi and Archetti, 2018], the authors compare a sequential and an integrated method to solve the problem. The sequential method solves the lot sizing problem first, determining the production quantities for the different periods and then solves an Inventory Routing Problem (IRP), to decide on the delivery schedule. The resolution of the integrated problem is presented in [Absi et al., 2015]. The main differences between this problem and our research is that the production and routing time have no influence on the customer satisfaction. In [Jia et al., 2019] the authors consider a parallel batch machines production system. Jobs are processed and then have to be transported to the customers. They assume that departure times are known and that the 3PL provider makes vehicles available for delivery. The authors aim to minimize the total weighted delivery completion time and propose an Ant Colony Optimization algorithm. However, there is no vehicle routing considered in this paper. In [Wang et al., 2019], the authors consider a parallel machines scheduling problem and a multi trip vehicle routing problem with time windows. The authors assume that travel times to be uncertain and look for the most robust min-cost routes. They propose a mathematical formulation of the problem (formulation of the elastic p-robustness approach) and a memetic algorithm. In [Marandani and Ghomi, 2019] the authors consider a network of factories, where each factory can be considered as a single machine (therefore similar to a parallel machine scheduling context). The jobs that are completed are transported to a central depot, and then delivered to the customers. Vehicles have limited capacity and delivery deadlines are assigned to each vehicle. The objective is to minimize the total delivery and tardiness penalty cost. The authors propose several population based metaheuristics, such as a genetic algorithm and particle swarm optimization algorithms. In [Wang et al., 2020] the authors consider a 3-stage hybrid flow shop scheduling problem followed by a multi-trip vehicle routing problem. The authors

want to minimize the maximum delivery time. To solve the problem, the authors propose a variable neighborhood search-based procedure and a constructive heuristic combined with VNS.

Apart for the PRP, most research only considers a cost for the inventory between the completion time of a job and its departure date, when jobs are served in batches ([Hassanzadeh et al., 2016], [Tavares-Neto and Seido, 2019]) or when fixed departure dates are considered for a vehicle ([Han et al., 2019]). Integrated models focusing on the delivery of products with limited lifespan can be found in [Lacomme et al., 2018], [Kergosien et al., 2017] and [Geismar et al., 2008].

Table 1 gives a synthesis of the papers presented above, dealing with integrated scheduling and routing problems. For each paper, we indicate the production environment, how the inventory cost is considered ('BEF' means before the production starts, 'WIP' means work-in-process inventory and 'FIN' means final inventory, i.e. between the job completion and its departure) and the delivery problem investigated (i.e. direct delivery or vehicle routing problem to solve and if the vehicle capacity is bounded). Moreover, we give an overview of the specific cost components considered in the objective function ('PdC' means production cost, 'IC' means inventory cost, 'RC' means fixed or variable routing cost and 'PenC' stands for tardiness penalty cost) and the proposed resolution methods.

3 Problem Description

In this paper we investigate an integrated supply chain problem involving two actors: a manufacturer and a third-party logistics (3PL) provider [Rohmer et al., 2015]. The integrated problem consists of two sub-problems at the operational level: an m-machine permutation flow shop for the manufacturer and a vehicle routing problem for the 3PL provider. Operating the flow shop, the manufacturer is looking to design a production schedule for a certain set of jobs on the machines under the following assumptions. Each job has given processing times for the different machines and the completion of a job on a machine, generates a work-in-process inventory before processing continues on the next machine. The completion of a job on the last machine results in a finished product inventory that is kept until departure of the job for delivery. The holding costs for the two kinds of inventory are job dependent, and in direct proportion to the holding time. Note, that the number of units of each product is known and thus not to be determined in the model. As such, the determination of the economic lot sizes is not a part of the problem that we consider. In addition to the inventory costs associated with production, the manufacturer also faces transportation costs for

Authors	Production system			Inventory	Delivery		Vehicle with limited capacity	Service requirement		Obj. function (Minimization)	Resolution Method
	One machine	Parallel machines	Other		Direct	VRP		Delivery due date	Lifespan		
[Wang et al., 2019]			•			•	•			$D_{\max}$	VNS
[Marandani and Ghomi, 2019]		•				•	•	•		RC, PenC	Sim. Ann., GA, PSO
[Wang et al., 2019]		•				•	•	•		RC, PenC	Memetic alg.
[Jia et al., 2019]		•			•		•	•		$\sum w_j D_j$	ACO
[Cheng et al., 2019]	•			BEF & FIN	•		•			Prd, IC, RC	Algo poly
[Tavares-Neto and Seido, 2019]		•		FIN		•	•			C_{max}	Iterated Greedy
[Han et al., 2019]			•	FIN	•		•	•		IC, RC, PenC	Algo poly
[Lacomme et al., 2018]	•					•	•		•	C_{max}	GRASP, ELS
[Devapriya et al., 2017]	•					•	•		•	RC	GA
[Kergosien et al., 2017]		•				•		•	•	$L_{\max}$	Benders dec.
[Hassanzadeh et al., 2016]			•	WIP & FIN	•			•		IC, RC, PenC	Greedy, GA
[Cheng et al., 2015]	•				•		•			PdC, RC	ACO
[Cakici et al., 2014]		•			•		•			$\sum w_j D_j$	Heuristics
[Chang et al., 2014]		•				•	•			RC, $\sum w_j D_j$	ACO
[Ullrich, 2013]		•				•	•	•		PenC	GA
[Farahani et al., 2012]		•				•		•		PdC, RC, PenC	Heuristic
[Scholz-Reiter et al., 2010]			•	WIP & FIN		•	•	•		IC, RC, PenC	-
[Armstrong et al., 2008]	•					•		•	•	PenC	B&B
[Geismar et al., 2008]	•				•		•		•	C_{max}	GA
[Li et al., 2005]	•					•	•			$\sum D_j$	Algo Poly
Our paper			•	WIP & FIN		•		•		IC, RC, PenC	Greedy, GA

Table 1: Bibliography synthesis

the number of vehicles used and penalty costs related to late delivery of products at the customer location.

Based on these assumptions, we propose an integrated framework for the production and distribution planning considering the scenario where the manufacturer dominates the decision making in the system. In this scenario, the manufacturer is able to decide on the number of vehicles, impose the composition of batches for the vehicles and set the departure dates. However, lacking the information regarding the actual delivery times – which depends on the routing decisions of the 3PL provider – the manufacturer is forced to estimate delivery times in order to reduce the tardiness penalty cost in case of late delivery. To estimate the delivery times, the manufacturer assumes a routing of the jobs in EDD order (earliest delivery due date first) for each of the batches.

Given the information provided by the manufacturer, the 3PL provider then determines the optimal route for each of vehicle (i.e. each batch). This information includes the number of vehicles used, the batch composition for each vehicle as well as the departure date for each batch. Moreover, the manufacturer also provides the 3PL provider with the estimated delivery dates by which the jobs should be delivered (based on EDD order as explained previously). In case these dates are not respected and delivery occurs late, the 3PL provider has to pay a penalty cost to the manufacturer. The routing for each batch always starts at the manufacturing site with all the vehicles readily available at the location. It is furthermore assumed, that the capacity of each vehicle is large enough to contain all the jobs in a batch. This assumption is supported by the fact, that the supplier knows the batch composition, to be transported, in advance and can therefore adapt the size/volume of the vehicle if necessary before the start of the delivery.

Both of the problems are interconnected and dependent on each other, so that the two agents aim to minimise their individual costs based on the behaviour of the other agent, assuming only limited information sharing between the two agents. The Figure 1 presents an overview of the system, illustrating the contractual relations between the two agents and the customers.

The paper differs from most of the existing literature in two ways: First, we consider a collaboration scenario between two agents with incomplete exchange of information, so that the manufacturer has to make planning decisions without knowing the true final cost associated with these decisions. Second, as we consider a system with two agents, different cost components are taken into account and financial exchanges take places in several directions as highlighted in the Figure 1.

While the proposed model is relatively generic and encompasses many special cases, the problem is highly relevant in the context of food and pharmaceuticals, where products are often perishable

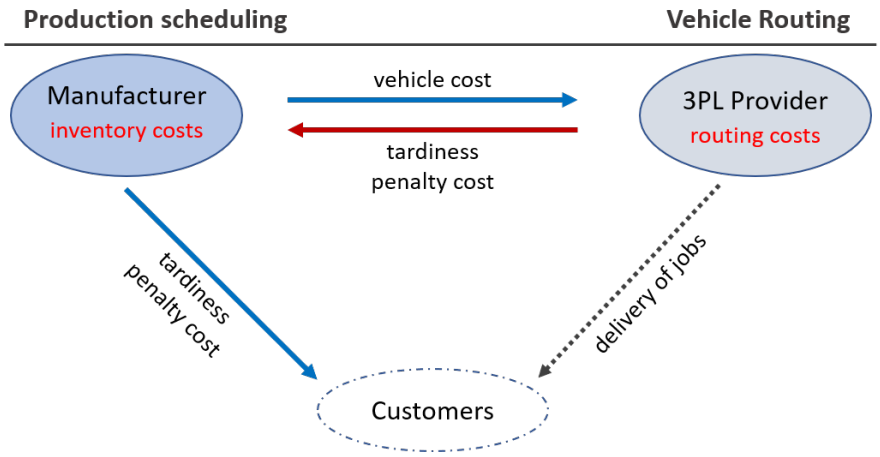


Figure 1: Contractual relations between the agents

and delivery times are of great importance. In these cases, manufacturers often specialise in the production activity and outsource the distribution task to a 3PL provider. A possible application area is for example the production and distribution planning of chemotherapy drugs. In such an application, batch sizes can, moreover, generally be considered very small, making it reasonable to assume an unlimited capacity of the vehicles.

The consideration of penalty costs is furthermore often highly relevant in practice and can be imagined in different ways. A delay in delivery can for example result in a complete rejection of the delivery by the customer, leading to what is called "leftover merchandise". In such a case, if the delay is not due to unforeseen circumstances (e.g. strike, weather, accident, ...) and if the delay causes a significant harm, the 3PL provider is not paid by the customer or the manufacturer, incentivising the 3PL provider to deliver in time. In this scenario, the manufacturer also faces a penalty cost towards the customer as the delivery is not received and thus not paid. In other cases, companies (e.g. Amazon, Zalando, ...) may guarantee their customers delivery within a very short time, while delivery is made by an external delivery provider (e.g. Chronopost, DHL, UPS, FedEx, ...). In case of late delivery, the service of the 3PL provider is still paid (its penalty cost is zero), but the producer must compensate his customer, for example by a free purchase order. The tardiness penalty cost for the manufacturer is high, whereas the only consequence for the 3PL provider is the bad image that it gives to the manufacturer and the risk of no longer being selected as a future distributor. These two examples can be seen as different instanciations of the model presented in this paper and provide an illustration to the contractual relations outlined in the Figure 1. Following from this, our aim is to formulate the problem mathematically and solve it heuristically by comparing two different metaheuristics in terms of their performance.

4 Mathematical Formulation

This section introduces the notations and parameters used and presents a formal description of the Mixed Integer Linear Programming (MILP) formulation for the two agents based on the assumptions outlined in Section 3.

4.1 Production Problem

The notations used to describe the manufacturer problem are presented in Table 2.

Table 2: **Summary of the notation – Manufacturer**

Parameters	
n	number of jobs
m	number of machines
$p_{i,j}$	processing time of job J_j on machine M_i, $1 \leq j \leq n$, $1 \leq i \leq m$
d_j	delivery due date of job J_j, $1 \leq j \leq n$
t_{s_1,s_2}	travel time between site s_1 and site s_2, $s_1, s_2 \in \{0, 1, ..., n+1\}$
HV	an arbitrary high value
$h_{i,j}^{WIP}$	work-in-process inventory holding cost of job J_j on machine M_i, $1 \leq j \leq n$, $1 \leq i \leq m$
h_j^{FIN}	inventory holding cost for finished product J_j, $1 \leq j \leq n$
π_j^M	penalty cost of the manufacturer for late delivery of job J_j, $1 \leq j \leq n$
c^V	cost per vehicle
Variables	
$y_{j1,j2}$	$= 1$ if job J_{j1} is scheduled before job J_{j2}, 0 otherwise, $1 \leq j1, j2 \leq n$
Z_k	$= 1$ if vehicle k is used, 0 otherwise, $1 \leq k \leq n$
$z_{j,k}$	$= 1$ if job J_j departs on vehicle k, 0 otherwise, $1 \leq j \leq n$, $1 \leq k \leq n$
$x_{j1,j2,k}$	$= 1$ if job J_{j1} and job J_{j2} are transported in vehicle k and J_{j1} is delivered before J_{j2}, 0 otherwise, $1 \leq j1, j2 \leq n$, $1 \leq k \leq n$
D_j^M	estimation of the delivery date of job J_j, $1 \leq j \leq n$
$C_{i,j}$	completion time of job J_j on machine M_i, $1 \leq j \leq n$, $1 \leq i \leq m$
F_k	departure time of vehicle k, $1 \leq k \leq n$
f_j	departure time of J_j, $1 \leq j \leq n$
$PT_j^M \geq 0$	estimation of the tardiness of job J_j for the manufacturer, $1 \leq j \leq n$

IC	total inventory costs
PPC^M	pseudo penalty cost of the manufacturer
Variables – known after the 3PL provider gives the delivery dates to the manufacturer	
D_j^{3PL}	delivery completion time of job J_j
T_j^M	tardiness of job J_j
VC	final vehicle cost

The scheduling problem of the manufacturer is solved by the following MILP model with the objective function (1).

$$\text{Minimize } PTC^M = IC + PPC^M + VC \tag{1}$$

We denote by IC the total inventory cost. Its expression (2) illustrated in the shaded area of the Figure 2 is the following:

$$IC = \sum_{j=1}^{n} \sum_{i=1}^{m-1} (C_{i+1,j} - p_{i+1,j} - C_{i,j}) h_{i,j}^{WIP} + \sum_{j=1}^{n} (f_j - C_{m,j}) h_j^{FIN} \tag{2}$$

where f_j denotes the departure time of J_j.

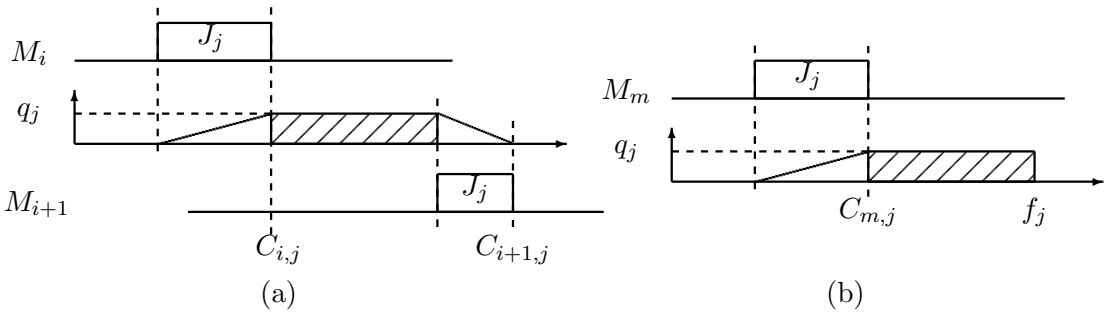


Figure 2: (a) Work-in-process inventory between machine M_i and M_{i+1} and (b) Finished product inventory

We denote by PC^M the penalty cost for tardiness. The expression (3) of PC^M is:

$$PC^M = \sum_{j=1}^{n} \pi_j^M T_j^M \tag{3}$$

The actual tardiness cost T_j^M depends on the delivery dates that will be given by the 3PL provider (denoted D_j^{3PL} for J_j) and that are not known at the moment. For this reason, the manufacturer assumes each batch will be served with the EDD delivery order by the 3PL provider. Based on this assumption, he considers the Pseudo Penalty Cost PPC^M (expression (5)) involving an estimation of the tardiness PT_j^M (expression (4)) of J_j, based on an estimation of the delivery time denoted D_j^M of J_j, defined by:

$$PT_j^M = \max(0, D_j^M - d_j) \tag{4}$$

We have:

$$PPC^M = \sum_{j=1}^{n} \pi_j^M PT_j^M \tag{5}$$

We denote by VC the vehicle cost by the expression (6):

$$VC = c^V V \tag{6}$$

where c^V is the cost of one vehicle and V is the number of vehicles required by the manufacturer.

The relative order between two jobs J_{j1} and J_{j2} ($\forall j1, j2 \in \{1, ..., n\}, j1 \leq j2$) is given by the constraints (7):

$$y_{j1,j2} + y_{j2,j1} = 1 \tag{7}$$

The resource constraints (8): (disjunctive constraints) allow to define the completion time of a job on any machine M_i ($\forall i \in \{1, ..., m\}$, $\forall j1, j2 \in \{1, ..., n\}$, $j1 \neq j2$):

$$C_{i,j2} \geq C_{i,j1} + p_{i,j2} - HV(1 - y_{j1,j2}) \tag{8}$$

If $y_{j1,j2}$ is equal to 1 (means that J_{j1} is scheduled before J_{j2}), then the constraint becomes $C_{i,j2} \geq C_{i,j1} + p_{i,j2}$, which means that J_{j2} cannot start before the end of J_{j1}. If $y_{j1,j2}$ is equal to 0, $C_{i,j2}$ is greater than a negative number and the constraint is cancelled.

The routing constraints (9) are given on any machine M_i ($i \in \{2, ..., m\}$) and for any job J_j ($j \in \{1, ..., n\}$):

$$C_{i,j} \geq C_{i-1,j} + p_{i,j} \tag{9}$$

With constraints (10), each job J_j is transported in a vehicle ($\forall j \in \{1, ..., n\}$):

$$\sum_{k=1}^{n} z_{j,k} = 1 \tag{10}$$

With constraints (11), each vehicle k ($\forall k \in \{1, ..., n\}$) leaves after the completion time of all jobs transported by this vehicle ($\forall j \in \{1, ..., n\}$):

$$F_k \geq C_{m,j} - HV(1 - z_{j,k}) \tag{11}$$

With constraints (12), a job cannot leave before its vehicle leaves ($\forall j \in \{1, ..., n\}$, $\forall k \in \{1, ..., n\}$):

$$f_j \geq F_k - HV(1 - z_{j,k}) \tag{12}$$

A vehicle k ($\forall k \in \{1, ..., n\}$) is used once it transports a job, constraints (13):

$$HV \times Z_k \geq \sum_{j=1}^{n} z_{j,k} \tag{13}$$

With constraints (14), in the same batch k ($\forall k \in \{1, ..., n\}$), job J_{j1} ($\forall j1 \in \{1, ..., n\}$) has a predecessor with a smaller due-date, or has the manufacturer site as predecessor:

$$\sum_{i \in \{0\} \cup \{j / d_j \leq d_{j1}\}} x_{i,j1,k} = z_{j1,k} \tag{14}$$

With constraints (15), in the same batch k ($\forall k \in \{1, ..., n\}$), job J_{j1} ($\forall j1 \in \{1, ..., n\}$) has a successor with a greater due-date, or has the depot site as successor:

$$\sum_{i \in \{n+1\} \cup \{j / d_j \geq d_{j1}\})} x_{j1,i,k} = z_{j1,k} \tag{15}$$

With Constraints (16), the manufacturer site has a successor during the route of vehicle k ($\forall k \in \{1, ..., n\}$) only if vehicle k is used:

$$\sum_{j=1}^{n} x_{0,j,k} \leq Z_k \tag{16}$$

The estimation of the delivery date is given by the following constraints (17) ($\forall j1, j2 \in \{1, ..., n\}$ and $\forall k \in \{1, ..., n\}$):

$$D_{j2}^{M} \geq D_{j1}^{M} + t_{j1,j2} - HV(1 - x_{j1,j2,k}) \tag{17}$$

The following constraints (18) give a lower bound of the delivery date of job J_j ($\forall j \in \{1, ..., n\}$) transported in vehicle k ($\forall k \in \{1, ..., n\}$):

$$D_{j}^{M} \geq F_k + t_{0,j} - HV(1 - z_{j,k}) \tag{18}$$

The estimation of the tardiness for job J_j ($\forall j \in \{1, ..., n\}$) is given by the following constraint (19):

$$PT_{j}^{M} \geq D_{j}^{M} - d_j \tag{19}$$

This model contains $O(n^3)$ binary variables, $O(nm)$ continuous variables, $O(n^3)$ big-M constraints and $O(n^2 m)$ constraints.

4.2 Distribution Problem

We assume that the optimization of the routing for each of the vehicles is independent and can thus be optimized separately. Therefore, for each trip, the set of delivery dates D_j^M is given by the manufacturer and we assume that vehicle k has to leave the manufacturer site at time F_k, which is also given. Furthermore, we assume that a batch contains s jobs to deliver. The notations used to described the 3PL provider problem are presented in Table 3.

Table 3: **Summary of notation – 3PL provider**

Parameters	
s	number of sites to visit (sites $1, .., s$ are the customer sites, site 0 is the manufacturer site and site $s+1$ is the 3PL provider site)
D_j^M	estimated delivery date for site j, $1 \leq j \leq s$
F_k	departure time from the manufacturer site of vehicle k, $1 \leq k \leq n$
$t_{j1,j2}$	travel time between site $j1$ and site $j2$, $0 \leq j1, j2 \leq s+1$
HV	an arbitrary high value (can be set to $2 \times \sum_{j1} \sum_{j2} t_{j1,j2}$)
$c_{j1,j2}$	cost of travel time between site $j1$ and site $j2$, $0 \leq j1, j2 \leq s+1$
π_j^{3PL}	penalty cost of 3PL provider for an excessive tardiness of job J_j, $1 \leq j \leq s$
Variables	
$x_{j1,j2}$	$= 1$ if site $j1$ is visited just before site $j2$, 0 otherwise, $0 \leq j1, j2 \leq s+1$
D_j^{3PL}	delivery date of J_j, $1 \leq j \leq s$
$T_j^{3PL} \geq 0$	tardiness of delivery of J_j, $1 \leq j \leq s$
RC	routing cost
PC^{3PL}	total penalty cost of 3PL provider

The routing of the 3PL provider is determined by solving the following Mixed Integer Linear Programming model with the following objectif value (20):

$$\text{Minimize } TC^{3PL} = RC + PC^{3PL} - VC \tag{20}$$

where:

The routing cost RC is the sum of the costs for all the routes, leaving from the manufacturer location and returning to the depot of the 3PL provider. For each trip, the 3PL provider bears the costs for the routing, denoted by RC_k for the route of vehicle k, which depends on the total travel time and a penalty cost per time unit to the manufacturer if the final delivery date (D_j^{3PL})

is greater than D_j^M. The total routing cost is then equal to expression (21):

$$RC = \sum_{k=1}^{V} RC_k \tag{21}$$

The penalty cost is denoted by PC^{3PL}. We assume that this function is related to the total tardiness, i.e. we denote by T_j^{3PL} (described by the expression (22)) the tardiness of delivery of J_j from the point of view of the 3PL provider (related to the delivery date D_j^M estimated by the manufacturer):

$$T_j^{3PL} = \max(0, D_j^{3PL} - D_j^M) \tag{22}$$

PC^{3PL} is defined by the expression (23):

$$PC^{3PL} = \sum_{j=1}^{n} \pi_j^{3PL} T_j^{3PL} \tag{23}$$

Finally, the manufacturer pays the vehicle cost VC to the 3PL provider, who considers this as a reward (negative cost), and which is included in the objective function (same definition as in (6)).

With constraints (24), any site s_1 ($\forall s_1 \in \{0, ..., s\}$) (excluding 3PL depot), has one successor site in $\{1, ..., s+1\}$:

$$\sum_{s2=1}^{s+1} x_{s1,s2} = 1 \tag{24}$$

With constraints (25), any site s_1 ($\forall s_1 \in \{1, ..., s+1\}$) (excluding manufacturer site), has one predecessor site in $\{0, ..., s\}$ constraints (25):

$$\sum_{s2=0}^{s} x_{s1,s2} = 1 \tag{25}$$

With constraints (26), a lower bound of the delivery date of job J_j ($\forall j \in \{1, ..., s\}$) is the following (remember that F_k is a data here):

$$D_j^M \geq F_k + t_{0,j} \tag{26}$$

The arrival date of each job at the customer site is given by the following constraints (27) ($\forall j1 \in \{1, ..., s\}$, $\forall j2 \in \{1, ..., s\}$):

$$D_{j2}^{3PL} \geq D_{j1}^{3PL} + t_{j1,j2} - HV(1 - x_{j1,j2}) \tag{27}$$

The tardiness expression and the costs are given in the following expressions (28) (remember that D_j^M is a data here):

$$T_j^{3PL} \geq D_j^{3PL} - D_j^M, \quad \forall j \in 1, .., s \tag{28}$$

This model contains $O(s^2)$ binary variables, $O(s)$ continuous variables, $O(s^2)$ big-M constraints and $O(s)$ constraints.

4.3 Integrated Problem

To solve the integrated problem, the following resolution algorithm is proposed. The MILP model of the manufacturer is solved first, determining the number of batches and their composition, the production sequence, the starting and completion time of the different jobs on the different machines and the vehicle departure dates. Based on this information, the 3PL provider solves the routing problem associated with each batch, deciding on the delivery order and the final delivery dates for each job. With this information from the 3PL provider, the manufacturer then updates his total cost calculation, replacing the initial pseudo penalty cost to be paid to the customers by the actual penalty cost, minus the penalty cost paid by the 3PL provider (that is considered as a reward, i.e. a negative cost).

The whole process is described more formally in Alg. 1.

Algorithm 1 General framework

The manufacturer optimizes his production schedule:

Solve MILP manufacturer minimising $PTC^M = IC + PPC^M + VC$

for k **in** 1 **to** V **do**

The 3PL provider optimizes the delivery of the batch:

Solve MILP 3PL provider minimising $TC_k^{3PL} = RC_k + PC_k^{3PL}$.

Compute the total cost of the 3PL provider: $TC^{3PL} = \sum_{k=1}^{V}(RC_k + PC_k^{3PL}) - VC$

Compute the total cost of the manufacturer: $TC^M = IC + PC^M - PC^{3PL}$+VC

5 Heuristic algorithms

Due to its complexity, the exact resolution of the manufacturer problem is very difficult within reasonable computation time and we propose two heuristic methods to find good solutions to the problem. The problem of the 3PL provider can be divided into the individual batches, resulting in a number of small VRPs with soft delivery due dates, that can be solved to optimality by means of the MILP presented in Section 4. The focus in this section is thus on the manufacturer problem.

In this context, we first describe the data structure used to present a solution and the method to evaluate the minimal total cost (inventory cost, pseudo penalty cost and routing cost) of a solution

in Section 5.1. Following from this, we present the two metaheuristic algorithms proposed to solve the problem: a GRASP algorithm in Section 5.2 and a Genetic Algorithm (GA) in Section 5.3.

5.1 Coding and evaluation of a solution

5.1.1 Coding of a solution

A complete solution to the problem is defined by the starting times of the jobs on the machines and the composition of batches to deliver. From this, it is possible to determine in polynomial time the inventory costs, the departure time of the batches for delivery knowing their composition – and applying the EDD rule for delivery – the estimation of the delivery dates. Then, all the elements are known to evaluate a solution.

A solution is represented as a sequence of batches. The order in which the jobs get assigned to the batches is the same as the processing sequence on the machines. However, the order in which the jobs are delivered in each batch is given by EDD order.

Example: We consider the following solution coding for a problem with $n = 8$ jobs.

$$\sigma = \{\{J_4, J_2, J_3\}, \{J_1, J_6\}, \{J_8, J_5, J_7\}\}$$

This solution corresponds to a production sequence equal to $(J_4, J_2, J_3, J_1, J_6, J_8, J_5, J_7)$ and to the composition of three batches, the first one being composed by the jobs J_4, J_2 and J_3.

5.1.2 Evaluation of a solution with optimal timing procedure

From the production sequence of the jobs, given by the coding of a solution, it is possible to find the optimal starting and completion times of the jobs that minimise the total cost. This can be done by an "optimal timing" procedure.

Let's consider a solution coding σ. We denote by ν the number of batches in σ. This corresponds to the number of vehicles which are used. Note, that the cost VC is a constant in this case $VC = c^V \nu$ and therefore it does not appear in the objective function. We assume w.l.o.g. that the jobs in σ are numbered from J_1 to J_n.

We know which is the last job of each batch and we denote by E_k the index of the last job in batch k, $1 \leq k \leq \nu$. We also know the assignment of jobs to batches and we denote by B_j the batch assigned to job J_j, $1 \leq j \leq n$.

With a simple preprocessing step, it is also possible to determine the time needed to deliver a job after the departure date of its vehicle. We denote by TT_j the travel time to deliver J_j.

The optimal schedule of the jobs (i.e. the optimal starting times of jobs) can then be determined by solving the following Linear Programming model called LP^{Fit}.

The data and variables used have been previously defined in Section 4.

$$LP^{Fit}\text{: min } IC + PPC^M \tag{29}$$

$$\text{s.t. } (2), (5)(19)$$

$$C_{1,1} \geq p_{1,1} \tag{30}$$

$$C_{i,j} \geq C_{i,j-1} + p_{i,j} \qquad \forall j \in \{1, ..., n\}, \forall i \in \{1, ..., m\} \tag{31}$$

$$C_{i,j} \geq C_{i-1,j} + p_{i,j} \qquad \forall j \in \{1, ..., n\}, \forall i \in \{1, ..., m\} \tag{32}$$

$$F_k = C_{m,E_k} \qquad \forall k \in \{1, ..., \nu\} \tag{33}$$

$$f_j = F_{B_j} \qquad \forall j \in \{1, ..., n\} \tag{34}$$

$$D_j^M = f_j + TT_j \qquad \forall j \in \{1, ..., n\} \tag{35}$$

In this problem, the number of vehicles is already determined, so VC is a constant and the objective function (29) is equivalent to the objective function (1). Constraint (30) determines the completion time of the first job on the first machine. Constraints (31) and (32) are equivalent to constraints (8) and (9) with y_{j_1,j_2} as known parameters. Constraints (33) and (34) are equivalent to constraints (11) and (12) with $z_{j,k}$ as known parameters and constraints (35) replace constraints (17) and (18). This LP model gives an optimal timing, i.e. allows to determine the optimal completion times of jobs of a given sequence.

5.1.3 Evaluation of a solution without optimal timing procedure

From the production sequence of the jobs, given by the coding of a solution, it is possible to define heuristically the starting and completion times of the jobs in order to minimise the total cost. This procedure provides no guaranty of finding the optimal timing of jobs, minimising the total inventory cost.

With a solution coding σ, this function proceeds in two steps:

1. According to the production sequence, the production of each task of the jobs is scheduled as soon as possible. This method provides the earliest possible departure dates of vehicles.

Therefore, this function also tends to minimise the penalty cost PC.

2. According to the vehicle departure dates computed in step 1, the production of each task of the different jobs are then scheduled as late as possible. This operation minimises the inventory cost IC^{FIN}.

Figures 3 and 4 illustrate the two steps described above. In this example, five jobs J_1, J_2, J_3, J_4 and J_5 are split in two batches, B_1 (dotted batch) and B_2 (grey batch) and must be scheduled. Figure 3 presents the result of step 1. The production is scheduled as soon as possible and the vehicle departure dates are determined.

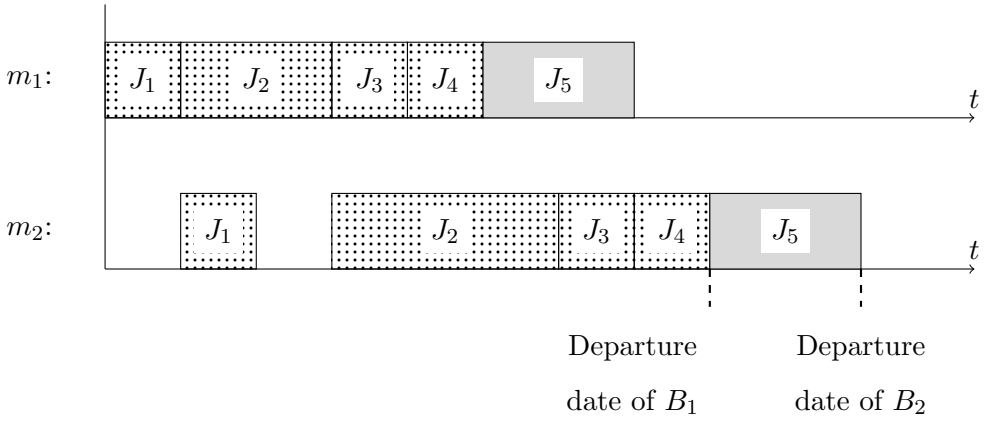


Figure 3: Scheduling as soon as possible

Figure 4 presents the result of step 2. The production dates of J_3, J_4 and J_5 on the first machine and the production date of J_1 on the second machine are rescheduled one unit later. This operation reduces the inventory cost.

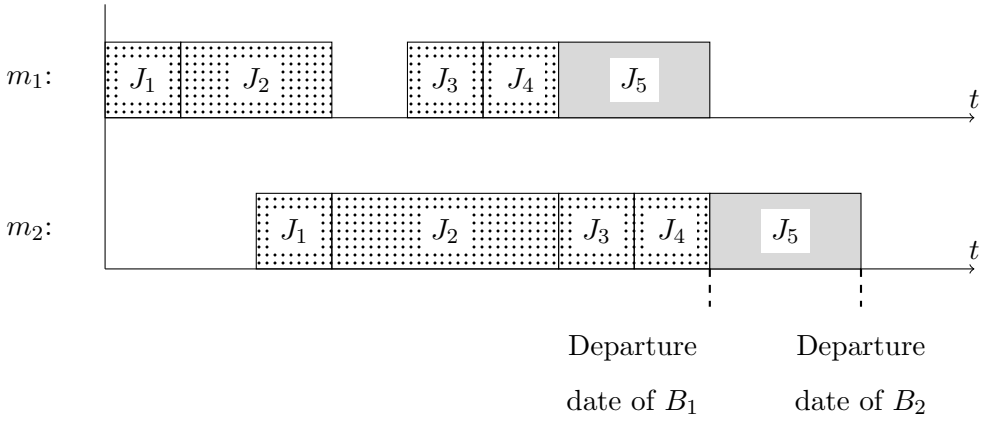


Figure 4: Scheduling as late as possible according to the vehicle departure dates

5.2 GRASP algorithm

First, we propose a GRASP (Greedy Randomized Adaptive Search Procedure) algorithm. This method typically consists of iterations made up from successive constructions of a greedy randomized solution and subsequent iterative improvements of it through a local search. The construction methods for an initial solution and the local search applied to improve this solution are now presented. The global method is given at the end of this section.

5.2.1 Initial solution generation methods

In order to build an initial solution, we determine first the sequencing of jobs and then we choose the composition of vehicles (or batching).

A parameter λ ($0 \leq \lambda \leq 1$) is given. From this parameter, an interval length related to the delivery due dates distribution is defined: $\lambda \times (d^{max} - d^{min})$, where d^{max} and d^{min} are respectively the maximum and the minimum delivery due dates of the remaining jobs to schedule. A *restricted list* of jobs is defined by the jobs having their delivery due date in the interval $[d^{min}, \lambda \times (d^{max} - d^{min})]$. The jobs are consecutively and randomly selected in this restricted list. The details of the algorithm are given in Alg. 2.

Algorithm 2 *sequencing_heuristic* algorithm

Parameter: λ
$\sigma = \emptyset$, $\mathcal{J} = \{J_j, j \in \{1, ..., n\}\}$
while $\mathcal{J} \neq \emptyset$ **do**
  $d^{min} = \min_{J_j \in \mathcal{J}} d_j$
  $d^{max} = \max_{J_j \in \mathcal{J}} d_j$
  List $= \{J_j / d^{min} \leq d_j \leq d^{min} + \lambda \times (d^{max} - d^{min})\}$
  Select randomly J_j in List
  $\sigma \leftarrow \sigma + J_j$
  $\mathcal{J} \leftarrow \mathcal{J} \setminus \{J_j\}$
return (σ)

5.2.2 Dispatching of jobs

After defining the production sequence on the machines, we propose a method to dispatch the jobs into batches. This method is inspired by the Split algorithm of Prins [Prins, 2004].

The Split algorithm [Prins, 2004] is presented for the $DVRP$ (*Distance-constraint Vehicle Routing Problem*) in order to determine an optimal set of trips for customers, respecting an already known sequencing order. The principle is to depict the problem by an oriented graph representing all the possible trips, with the associated costs on edges. The shortest path in this graph gives the optimal set of trips and, therefore, the batching.

In the adapted version of this algorithm, the input is the production sequence σ computed by the scheduling heuristic (Algorithm 2). The jobs are indexed according to this order. All the production dates associated with the jobs are scheduled as soon as possible. Let consider a graph $G = (V, E)$. V has $n + 1$ vertices, one vertex for each job (J_0 is a dummy job). E has one edge for each couple of vertices (J_1, J_2) with $J_1 < J_2$. Each edge (J_1, J_2) represents a batch. This batch contains all the jobs between $J_1 + 1^{th}$ job to the J_2^{th} job of the production sequence. The cost of an edge is the sum of two costs related to the batch : (1) the delivery penalty cost computed by the manufacturer, (2) the fixed cost of a vehicle. For a batch, to compute its tardiness penalties (and so the delivery date of its jobs), we consider that the departure date of the vehicle is equal to the maximum production date of the jobs in this batch.

After building graph G, we use Bellman's algorithm to find a shortest path from vertex J_0 to J_n. This evaluation is reasonably fast because G has no cycle. The worst case complexity is $O(n^2)$. The edges used in the shortest path represent the batches used in the solution.

This method does not consider the inventory costs, but minimises the objective function $VC + PPC^M$ for a specific production sequence.

Example: Let's consider a two-job and two-machine instance with $p_{11} = p_{12} = p_{22} = 1$, $p_{21} = 2$, $h_{1,1}^{WIP} = h_{1,2}^{WIP} = 1$, $h_1^{FIN} = h_2^{FIN} = 2$ and $c^V = 5$. The production sequence is (J_1, J_2), the associate graph contains 3 vertices and 3 edges for the 3 possible batches $B_1 = \{J_1\}$, $B_2 = \{J_2\}$ and $B_3 = \{J_1, J_2\}$. The jobs are left shifted to give the departure date of the batch (see Figure 5).

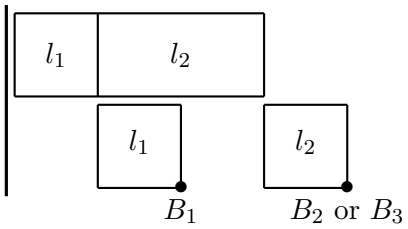
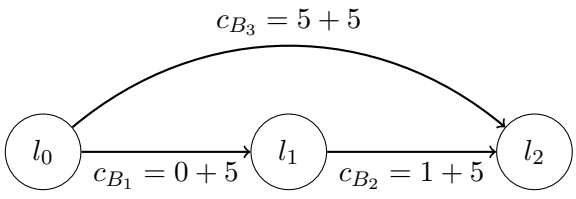


Figure 5: Production and departure date of batch B_1, B_2 and B_3



$c_X = PPC_X^M + c^V$

Figure 6: Graph of the delivery cost

The departure time of the batch B_1 finishing by job J_1 is equal to 2 and the departure time of batches B_2 and B_3 finishing by job J_2 is equal to 4. If we denote by c_X the cost associated to batch B_X, we have $c_X = IC_X + PPC_X^M + c^V$.

The data that are used for computing PPC^M lead to the following results (not detailed here for sake of simplicity): $PPC_1^M = 0$, $PPC_2^M = 1$ and $PPC_3^M = 4$.

Using Bellman's algorithm on the graph of Figure 6, we obtain the shortest path equal to $\{(0,1),(1,2)\}$ with an associate cost of 11. It means that J_1 and J_2 are in two separate batches. If c^V increases from 5 to 7, the shortest path changes to $\{(0,2)\}$ with an associated cost of 14.

5.2.3 Local search algorithm

A local search algorithm is used in order to improve the current solution quickly, applying a first improvement rule with a random start for each operator. The local search has three parameters. The first parameter is denoted by τ^{eval} and corresponds to the evaluation function applied to each new solution. If $\tau^{eval} = OPT$, the evaluation function LP^{Fit} with optimal timing is used to evaluate the solutions. If $\tau^{eval} = HEU$, the evaluation function without optimal timing (see Section 5.1.3) is used. The second parameter is denoted by τ^{eff} and describes the intensity of the local search. The last parameter is Δ, the size of the neighbourhood operator.

Four different neighbourhood operators are proposed in this local search to improve the solution.

The *Swap* operator exchanges two jobs in the sequence. To avoid symmetry, let i and j be the positions of two jobs, these jobs can be swapped if and only if $i < j$. For example :

$$\text{Swap of 2 and 5: } \{\{1,\underline{2},3,4\}\{\underline{5},6,7\}\} \rightarrow \{\{1,\underline{5},3,4\}\{\underline{2},6,7\}\}$$

The *EBSR/EFSR* (Extract and Backward / Forward Shift Reinsertion) operator extracts a job from position i and reinserts it in the sequence at a position $j < i$ (backward), or at a position $j > i$ (forward).

$$\textit{EFSR} \text{ of 2 at position 5: } \{\{1,\underline{2},3,4\}\{5,6,7\}\} \rightarrow \{\{1,3,4\}\{5,\underline{2},6,7\}\}$$
$$\textit{EBSR} \text{ of 5 at position 2: } \{\{1,2,3,4\}\{\underline{5},6,7\}\} \rightarrow \{\{1,\underline{5},2,3,4\}\{6,7\}\}$$

This operator can empty a batch if the job which is extracted was the last in the batch. The two operators are combined in the *EBFSR* operator by testing for a job i a reinsertion at a position j smaller than i, and then greater than i. To avoid an explosion of computation time associated with our local search, the operators are limited by a parameter Δ which reduces the distance of

the positions of the jobs considered by *SWAP* and *EBFSR*. A move associated to the positions i and j is accepted if $|j - i| \leq \Delta$.

The *Merging* operator merges two consecutive batches and the *Division* operator splits a batch in two batches just after a target job (this operator is the only one which can increase the number of batches).

$$\text{Merging of } B_1 \text{ and } B_2: \quad \{\underline{\{1,2,3\}\{4,5\}}\{6,7\}\} \rightarrow \{\underline{\{1,2,3,4,5\}}\{6,7\}\}$$

$$\text{Division of } B_1 \text{ after job } J_3: \quad \{\{1,2,\underline{3},4,5\}\{6,7\}\} \rightarrow \{\{1,2,3\}\{4,5\}\{6,7\}\}$$

The different operators are always used in the following order: *Swap - EBFSR - Merging - Division*. The local search stops when no operator can improve the current solution.

The neighbourhood operators are applied once and after that, the behaviour of the algorithm depends on the value of the τ^{eff} parameter:

- if $\tau^{eff} = O$, the local search stops.
- if $\tau^{eff} = S$, the local search tests all the neighbourhood operators another time. It comes to an end when no operator can improve the current solution. The found solution in this case presents a local optimum.

This method is denoted by LS.

5.2.4 Global GRASP algorithm

The idea of this algorithm is to quickly generate a considerable number of good quality solutions and to return the best one after a certain amount of time. Solutions are generated by using the previous sequencing and batching procedures (Alg. 2 with parameter $\lambda \in [0, 1]$). Then, the solution is improved by LS until a local minimum ($\tau^{eff} = S$) is found (see Alg. 3). The result of the algorithm is the best solution found after $CPUmax$ time units. If $\tau^{eval} = HEU$, all the solutions considered during the local search are determined without the optimal timing, but we choose to apply the procedure LP^{Fit} to improve each final solution returned by the local search.

5.3 Genetic Algorithm

A Genetic Algorithm (GA) [Holland, 1975] manages a population of solutions using rules inspired by the natural evolution law, such as reproduction, competition and selection, in order to adapt and improve over the generations. In the proposed algorithm, the population is composed of δ_{pop}

Algorithm 3 GRASP heuristic

Parameters λ, $CPUmax$
$f(S^*) = \infty$
while $CPU \leq CPUmax$ **do**
  // Initial solution creation
  $\sigma = sequencing\_heuristic(\lambda)$ (Alg. 2)
  $S = batching\_heuristic(\sigma)$
  // First Improvement
  $S = LS(S)$
  // Update best solution
  if $f(S) \leq f(S^*)$ **then**
    $S^* = S$
  Update CPU
if $\tau^{eval} = HEU$ **then**
  $S^* = LP^{Fit}(S^*)$
return S^*

solutions. As for the GRASP algorithm, the encoding of a solution is the production sequence of jobs and the same two evaluation functions are used.

5.3.1 Initial population

The solutions of the initial population are built using the sequencing procedure of the GRASP algorithm (Section 5.2.1). This procedure uses the parameter λ. Solutions are generated in order to obtain δ_{pop} unique solutions (more than δ_{pop} generations can be required). Two solutions are considered identical if they have the same objective function. Depending on the setting of parameter τ^{eval}, the resulting solutions are then fully determined with or without the optimal timing procedure LP^{Fit}.

5.3.2 Generate a new offspring

A new solution (an offspring) is the combination of two individuals. Each parent is chosen by using a binary tournament: two solutions are picked at random from the population and the one with the best fitness value is conserved. A couple of parents is composed by two distinct solutions and

each couple generates two offsprings.

A crossover procedure defines the offsprings from the two selected parents. Each parent has a role, Parent 1 or Parent 2. We use the LOX operator (Linear Order crossover). The principle is to keep the central part of the first parent (delimited by two random indices) and to move the other elements backward and forward, based on the sequence of the second parent, in order to preserve a part of the supposedly good solution as well as the relative order of the other jobs. An example presented in Table 4 illustrates the crossover with a sequence of seven jobs.

Parent 1 :	1	2	3	4	5	6	7
Parent 2 :	5	6	4	1	2	3	7
Random indices: $c_1 = 2,\ c_2 = 5$							
Parent 1 :	1	2	$\lvert\underline{3}$	$\underline{4}$	$\underline{5}\rvert$	6	7
Parent 2 :	$\underline{5}$	6	$\underline{4}$	1	2	$\underline{3}$	7
Offspring:	-	-	3	4	5	-	-
Parent 2 :	X	6	X	1	2	X	7
Offspring:	6	1	X	X	X	2	7
Offspring:	6	1	3	4	5	2	7

Table 4: LOX crossover example for offspring 1

During the first step of the cross-over, index 2 and 5 are selected. The sequence between index 2 and 5 of Parent 1 is copied in the offspring. Then the offspring sequence is completed according to the production sequence of Parent 2.

To define the second offspring, Parent 1 and Parent 2 roles are exchanged.

After the cross-over, each offspring mutates with a probability of p_{mut}. This mutation is an application of the LS local search on the solution (parameters τ^{eff} and Δ are required).

5.3.3 Population management

For each generation, $\delta/2$ couples of parents are generated by binary tournament. Both solutions in a couple are unique, but a solution can be part of several couples.

An offspring is added to the population only if it is a new solution in the population. The offspring then replaces a randomly selected solution among the half of the population with the lowest objective functions. This procedure avoids replacing (and thus losing) good solutions. The population is then sorted to proceed efficiently with the addition of offsprings.

New generations are performed until the computation time limit is reached. After that, the GA returns the solution of the population with the best fitness function.

5.4 Genetic algorithm parameters

The GA uses the following parameters: δ_{pop} denotes the population size and $p_{mut} \in [0,1]$ the offspring mutation probability. The parameter $\tau^{eval} \in \{OPT, HEU\}$ is used to determine whether to apply the optimal timing procedure or not. Parameter $\lambda \in [0,1]$ influences the initial solution generation. Moreover, the mutation phase depends on the parameters τ^{eff} and Δ of the local search LS.

6 Computational experiments

We present in this section the computational results which have been performed on a computer with an Intel Core i7-7820HQ and 16 Go RAM. The program is developed in C++ with Visual Studio. The solver is Cplex 12.7.1.

6.1 Data generation

The results presented in this section have been obtained with random instances generated according to the following specifications. We consider a flow shop problem with $m = 5$ machines. We set a new parameter noted $\alpha = 100$. For each job J_j and each machine M_i, the processing time $p_{i,j}$ is randomly chosen in $[1, \alpha]$ and the delivery due date d_j is randomly chosen in $[1, n \times \alpha]$. The different sites are placed randomly on a square of size $(3\alpha \times 3\alpha)$ and the distance t_{j_1,j_2} between sites j_1 and j_2 is the classical euclidean distance.

The costs are fixed so that the number of required vehicles in an optimal solution will not be equal to 1 (one vehicle for all the jobs) or n (one vehicle per job). For each command J_j, the inventory costs increase by 1 or 2 units from one machine to the next: the inventory cost $h_{1,j}^{WIP}$ is randomly generated in $[1,2]$, $h_{i+1,j}^{WIP}$ is randomly generated in $[h_{i,j}^{WIP}+1, h_{i,j}^{WIP}+2]$ and the final inventory cost is generated in $[h_{m,j}^{WIP}+1, h_{m,j}^{WIP}+2]$. The tardiness penalty cost π_j is randomly

generated in $[5, 10]$. The cost of a vehicle c^V is equal to 4000.

Sets of 10 instances are generated for each value of $n \in \{6, 7, 8, 9, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$. Instances with $n \in \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$ are used for testing the heuristic algorithms and instances with $n \in \{6, 7, 8, 9, 10\}$ are used for testing the MILP model.

6.2 Metaheuristic algorithms evaluation

The heuristic algorithms solve the first part of the problem, where the manufacturer determines the schedule of the jobs and the batch composition. The computation time limit has been fixed to $\lceil \frac{n}{10} \rceil$ minutes per instance.

6.2.1 GRASP algorithm evaluation

The GRASP has the following parameters: λ is used to generate the initial production sequence of solutions, the neighbourhood size Δ is used during the local search and the method used to fix the jobs starting times is defined by $\tau^{eval} \in \{OPT, HEU\}$. For the GRASP, parameter τ^{eff} = "S".

According to preliminary results, the evaluation function of a solution (parameter τ^{eval}) has a significant impact on the GRASP results. Indeed, when $\tau^{eval} = OPT$, the LP model is called a large number of times during the local search, thus slowing down the search procedure. In contrast, when $\tau^{eval} = HEU$, the search is able to visit more solutions during the same computation time.

Therefore, two sets of parameters are tested depending on the value of τ^{eval}. For $\tau^{eval} = OPT$, the parameters of GRASP are the following: $\lambda = 0.1, \tau^{eff} = S$ and $\Delta = 5$. The notation for this method is $GRASP^{OPT}$. For $\tau^{eval} = HEU$, the parameters of GRASP are set to: $\lambda = 0.2, \tau^{eff} = S$ and $\Delta = 20$. The notation for this method is $GRASP^{HEU}$.

Table 5 shows the number of solutions explored by each method within the computation time limit.

Additional studies show that the computation time of the evaluation method without optimal timing is around 10 microseconds for every instance size. For the evaluation method with optimal timing, the computation time is between 1 and 20 milliseconds. The computation time ratio between the methods is around 1000.

The number of solutions generated by each method decreases significantly with the instance size. It is divided by 18 for $GRASP^{HEU}$ (from 12096 for instances of 10 jobs to 691 for instances of 100 jobs) and 56 for $GRASP^{OPT}$ (from 168 for instances of 10 jobs to 3 for instances of 100 jobs). These results are explained by the large computation time of the LP model LP^{Fit} as the instance

n	Nb. of generations	
	$GRASP^{HEU}$	$GRASP^{OPT}$
10	12096	168
20	6670	61
30	3572	32
40	3585	18
50	2432	12
60	1757	8
70	1300	6
80	1034	5
90	860	4
100	691	3

Table 5: Number of generations of methods $GRASP^{HEU}$ and $GRASP^{OPT}$

size increases. At the end, the number of solutions generated by $GRASP^{OPT}$ for instances with 60 jobs or more does not exceed 10. This number of solutions (built randomly) is not sufficient for a good efficiency of the GRASP algorithm.

Table 6 compares the quality of the solutions returned by the two methods. The column "Improvement" provides some average relative improvements. Let z_1 denote the initial solution value of a method, z_2 the value of the solution when $\tau^{eval} = HEU$, just before the last step (improvement by LP^{Fit}), z_3 the value of the final solution of a method. We define:

$A1 = \frac{z_1 - z_2}{z_2}$ the average relative deviation between the initial solution and the solution before improvement,

$A2 = \frac{z_2 - z_3}{z_3}$ the deviation provided by the application of the evaluation method with optimal timing on the final solution found by $GRASP^{HEU}$,

$A3 = \frac{z_1 - z_3}{z_3}$ the average relative deviation between the initial solution and the final solution.

Columns "Nb. of best" show the number of best solutions found by each method (10 instances per set) and columns "Gap" provide the average gap between the solutions found by the methods and the best solutions found.

As shown in columns "Improvement", the improvement provided by the local search increases with the instance size for both methods (from 27.8% and 26.2% for instances of 10 jobs up to

n	Improvement (%)			Nb. best		Gap (%)	
	$GRASP^{HEU}$		$GRASP^{OPT}$	$\#G^{HEU}$	$\#G^{OPT}$	$GRASP^{HEU}$	$GRASP^{OPT}$
	A1	A2	A3				
10	27.8 %	0.0 %	26.2 %	9	6	0.1 %	0.3 %
20	36.0 %	0.1 %	33.0 %	10	4	0.0 %	1.2 %
30	40.4 %	0.3 %	34.8 %	9	1	0.1 %	2.1 %
40	45.7 %	0.4 %	37.8 %	9	1	0.0 %	2.8 %
50	62.3 %	0.5 %	52.5 %	10	0	0.0 %	3.2 %
60	78.0 %	0.4 %	64.9 %	8	2	0.1 %	2.3 %
70	94.9 %	0.5 %	79.6 %	10	0	0.0 %	2.7 %
80	112.3 %	0.5 %	90.3 %	10	0	0.0 %	2.5 %
90	132.0 %	0.5 %	111.1 %	10	0	0.0 %	3.4 %
100	158.2 %	0.5 %	136.9 %	10	0	0.0 %	2.9 %

Table 6: Average performance of methods $GRASP^{HEU}$ and $GRASP^{OPT}$

158.2% and 136.9% for instances of 100 jobs). This characteristic explains the decrease in the number of solutions generated by the methods. Indeed, with a higher potential of improvement of the initial solution for larger instance sizes, the local search takes more time to improve the solution. For the $GRASP^{HEU}$ method, the column "A2" shows the improvement between a locally optimal solution evaluated with the method without optimal timing and the same solution evaluated with the method with optimal timing. This improvement increases with the instance size but remains rather low (under 0.5%). The last columns ("Nb. best" and "Gap") clearly indicate a better performance of the $GRASP^{HEU}$ for all instance sizes.

These results show that using a heuristic approach to determine the jobs starting times has several advantages. This evaluation allows to apply a local search with a larger neighbourhood, which leads to a better performance (comparison between columns "$A1$"). In conclusion, the results for the $GRASP^{HEU}$ are better than the results for the $GRASP^{OPT}$.

6.2.2 Genetic algorithm evaluation (GA)

The GA has the following parameters: pop_{size} is the population size, λ is used to generate the production sequence of the first solutions of the population, p_{mut} is the probability of mutation for each new generated offspring, Δ defines the neighbourhood size used, $\tau^{eff} \in \{O, S\}$ defines if the neighbourhood exploration is used only once ("O") or until a local optimum is found ("S"), and the parameter $\tau^{eval} \in \{HEU, OPT\}$ indicates if the evaluation method uses optimal timing or not.

Preliminary tests show that a small population size pop_{size} (equal to 20) gives better results for most of the configurations. In order to have some diversity in the initial population, the parameter λ is set to 0.2. Likewise, the mutation probability is set to 0.3. The remaining parameters are tested with $\tau^{eval} = OPT$ and $\Delta = 20$. This combination does not create enough generations of the GA within the computational time limit. Note that a generation corresponds to the creation of δ_{pop} offsprings (by crossover and mutation) from the current population. In the following, the name of each method corresponds to the value of its parameters: $GA^{\tau^{eval},\tau^{eff},\Delta}$.

Table 7 gives the average number of generations for each method.

n	$GA^{OPT,S,5}$	$GA^{OPT,O,5}$	$GA^{HEU,S,20}$	$GA^{HEU,S,5}$	$GA^{HEU,O,20}$	$GA^{HEU,O,5}$
10	22	44	920	958	917	965
20	9	21	859	991	924	1056
30	4	12	565	863	746	925
40	2	8	402	657	603	752
50	2	6	224	452	486	615
60	2	5	142	345	413	544
70	1	4	107	280	349	375
80	1	4	80	235	299	325
90	1	3	68	202	264	385
100	1	2	55	166	228	343

Table 7: Average number of generations by method

In the same way as for the GRASP, this table shows that τ^{eval} is the parameter with the highest impact on the number of generations. For $\tau^{eval} = OPT$, whatever the other parameters, the associated methods do not create more than 10 generations for instances with more than 30 jobs. The second parameter with high impact is τ^{eff} ($\tau^{eff} = S$ creates less generations), followed by the parameter Δ with the lowest impact.

Table 8 focuses on the analysis of parameter τ^{eff}.

For each combination of parameters $(\tau^{eval}, \Delta) \in \{(OPT, 5), (HEU, 5), (HEU, 20)\}$, the two settings of $\tau^{eff} \in \{O, S\}$ are tested and compared based in Table 8. In the Table, the common parameters for the two methods are indicated on the first line, the value of τ^{eff} on the second line. The average gaps, presented in the Table, are calculated as follows. For an instance and a value of (τ^{eval}, Δ), let z_O be the value of the solution found with $\tau^{eff} = O$, z_S the value of the solution

found with $\tau^{eff} = S$ and $z^* = \min(z_O, z_S)$. The gap we measure here is then the gap between the result obtained with $\tau^{eff} = O$ and $\tau^{eff} = S$ and the best result found by the two methods, i.e. $\frac{z_O - z^*}{z^*}$ and $\frac{z_S - z^*}{z^*}$.

n	$\tau^{eval} = OPT, \Delta = 5$		$\tau^{eval} = HEU, \Delta = 5$		$\tau^{eval} = HEU, \Delta = 20$	
	$\tau^{eff} = O$	$\tau^{eff} = S$	$\tau^{eff} = O$	$\tau^{eff} = S$	$\tau^{eff} = O$	$\tau^{eff} = S$
10	0.0 %	0.0 %	0.4 %	0.0 %	0.1 %	0.1 %
20	0.7 %	0.2 %	0.6 %	0.3 %	0.1 %	0.0 %
30	0.4 %	0.7 %	1.3 %	0.1 %	1.4 %	0.0 %
40	1.5 %	0.3 %	2.3 %	0.1 %	1.0 %	0.1 %
50	4.7 %	0.0 %	3.8 %	0.0 %	2.7 %	0.0 %
60	8.3 %	0.0 %	7.1 %	0.0 %	4.8 %	0.0 %
70	11.2 %	0.2 %	9.6 %	0.0 %	4.8 %	0.0 %
80	18.4 %	0.0 %	11.6 %	0.0 %	6.9 %	0.0 %
90	27.4 %	0.0 %	13.5 %	0.0 %	9.7 %	0.0 %
100	32.3 %	0.0 %	15.6 %	0.0 %	9.2 %	0.0 %

Table 8: Average gap to the best solution found when $\tau^{eff} = O$ or $\tau^{eff} = S$.

For each value of (τ^{eval}, Δ), this table shows that the parameter $\tau^{eff} = O$ gives poor results in comparison to the parameter $\tau^{eff} = S$. For instance sizes larger than 40 jobs, $\tau^{eff} = O$ never finds a larger number of best solutions than $\tau^{eff} = O$ (there are 10 instances per set). Notice that each time a gap associated to a method is equal to 0.0% for an instance set, this method finds all the best solutions for this instance set. Furthermore, additional tests are carried out on methods with $\tau^{eff} = O$. For these methods, the time constraint is relaxed to 30 minutes for solving some instances of 30 jobs. Even with this extra computation time, the results show that the methods with $\tau^{eff} = O$ are not able to find better (or equivalent) solutions compared to methods with $\tau^{eff} = S$ (with a 3-minute computation time limit). Indeed, the dispatching method (Section 5.2.2) is applied for each new offspring added in the population and may cause a loss in solution quality (this method is a heuristic). Important modifications may be required to correct this loss. This modification can take place during the mutation. However, a local search with the parameter $\tau^{eff} = O$ does not provide a sufficient improvement.

Table 9 evaluates the method with different values of Δ when $\tau^{eval} = HEU$ and $\tau^{eff} = S$. The column "Gap" (resp. "Nb. best") shows the average gap to the best solution found (resp. the

number of best solutions found) for each method with two different settings of Δ.

	Gap (%)		Nb. best	
n	$\Delta = 20$	$\Delta = 5$	$\Delta = 20$	$\Delta = 5$
10	0.1 %	0.0 %	9	10
20	0.2 %	0.2 %	5	8
30	0.0 %	0.9 %	9	1
40	0.0 %	1.1 %	10	0
50	0.0 %	1.7 %	10	0
60	0.0 %	1.7 %	10	0
70	0.0 %	2.1 %	10	0
80	0.0 %	2.5 %	10	0
90	0.0 %	3.0 %	10	0
100	0.0 %	3.5 %	10	0

Table 9: Evaluation of parameter Δ for $\tau^{eval} = HEU$ and $\tau^{eff} = S$

The parameter setting $\Delta = 5$ finds no best solution for instance sizes higher than 30 jobs. However, this parameter finds a larger number of best solutions for small instances (10 and 20 jobs). For smaller instance sizes, a small Δ is enough to allow an exhaustive exploration of the solution neighbourhood. Furthermore, according to Table 7, methods with parameter $\Delta = 5$ generate 4% and 15% more solutions than methods with parameter $\Delta = 20$. However, even on small instance sizes, the gap to the best solution found remains small (under 0.2%).

The previous results show that the best value of the parameters for $\tau^{eval} = OPT$ are $\tau^{eff} = S$ and $\Delta = 5$ and for $\tau^{eval} = HEU$ are $\tau^{eff} = S$ and $\Delta = 20$.

So, Table 10 compares both methods: $AG^{OPT,S,5}$ and $AG^{HEU,S,20}$. The column "Gap" (resp. "Nb. best") gives the average gap to the best solution found (resp. the number of best solutions found) for each of the two methods.

This table shows that $AG^{HEU,S,20}$ has better results than $AG^{OPT,S,5}$ (or equivalent on small instance sizes). The main explanation for this result is the small number of generations of the method $AG^{OPT,S,5}$ (no more than 10 on average for instance sizes larger than 10). So, the mechanisms of the genetic algorithm do not have enough time to be efficient and converge to good solutions.

	Gap (%)		Nb. best	
n	$AG^{HEU,S,20}$	$AG^{OPT,S,5}$	$AG^{HEU,S,20}$	$AG^{OPT,S,5}$
10	0.1 %	0.2 %	9.0	9.0
20	0.3 %	0.3 %	6.0	6.0
30	0.1 %	2.1 %	8.0	2.0
40	0.0 %	2.9 %	10.0	0.0
50	0.0 %	4.6 %	10.0	0.0
60	0.0 %	5.4 %	10.0	0.0
70	0.0 %	7.2 %	10.0	0.0
80	0.0 %	5.6 %	10.0	0.0
90	0.0 %	6.2 %	10.0	0.0
100	0.0 %	4.9 %	10.0	0.0

Table 10: Comparison of methods $AG^{HEU,S,20}$ and $AG^{OPT,S,5}$

6.3 GRASP and genetic algorithm comparison

In this section, GRASP and GA are compared on their best sets of parameters. For GRASP, it is the $GRASP^{HEU}$ method ($\tau^{eval}HEU,\ \lambda = 0.2,\ \tau^{eff} = S$ and $\Delta = 20$). For the genetic algorithm, it is the $AG^{HEU,S,20}$ method ($pop_{size} = 20$, $p_{mut} = 0.3$, $\tau^{eval}HEU,\ \lambda = 0.2,\ \tau^{eff} = S$ and $\Delta = 20$).

Table 11 compares the performances of both methods. We denote by z_{GRASP} the value of the solution found by the GRASP, z_{GA} the value of the solution found by the genetic algorithm and z^* the minimum of z_{GRASP} and z_{GA}. Column "Gap" (resp. "Nb. best") gives the average deviation to the best solution found (resp. the number of best solutions found) for each methods, i.e. $\frac{z_{GRASP}-z^*}{z^*}$ and $\frac{z_{GA}-z^*}{z^*}$.

Both methods are equivalent on instances of size 10. They find the same results for 9 instances out of 10 and the gap for the last solution remains low (around 0.6%). For instances of size 20 and 30, the method $GRASP^{HEU}$ finds a larger number of best solutions (9 and 6 against 4) with a lower average gap (0.03% and 0.16% against 0.46% and 0.28%). However, for larger instance sizes, the method $AG^{HEU,S,20}$ has better performances with at least 8 best solutions found out of 10 and an average gap lower than 0.17%. For these instances, $GRASP^{HEU}$ finds only twice the best solution with an average gap between 0.28% and 1.24%.

According to the gaps presented in this table, both methods give solutions with close objective function values. The parameters $\tau^{eval} = HEU$, $\tau^{eff} = S$ and $\Delta = 20$ are identical for the

	Gap (%)		Nb. best	
n	$GRASP^{HEU}$	$AG^{HEU,S,20}$	$GRASP^{HEU}$	$AG^{HEU,S,20}$
10	0.06 %	0.00 %	9	10
20	0.03 %	0.46 %	9	4
30	0.16 %	0.28 %	6	4
40	0.28 %	0.17 %	1	9
50	0.58 %	0.00 %	0	10
60	1.02 %	0.00 %	0	10
70	0.82 %	0.02 %	2	8
80	1.24 %	0.01 %	1	9
90	0.96 %	0.16 %	2	8
100	0.79 %	0.02 %	1	9

Table 11: Comparison of methods G^{HEU} and $AG^{HEU,S,20}$

local searches of both methods. However, the evolutionary mechanisms of $AG^{HEU,S,20}$ give better solutions on large instance sizes. Indeed, the method $GRASP^{HEU}$ applies local search on randomly generated instances, whereas $AG^{HEU,S,20}$ applies local search on a solution from a population improved at each generation. This difference gives an advantage to the $AG^{HEU,S,20}$ method. This small deviation in terms of objective functions can also be explained by the high value of the total cost as the number of jobs increases (in the range of 150000 for instances with 90 jobs and 170000 for instances of 100 jobs).

6.4 Comparison of the methods on small size instances

This section compares the performances of the resolution of the model presented in Section 4.1 by a commercial solver with the performances of the different versions of the GRASP and the genetic algorithm method. These tests are done on small size instances ($n \in \{5, 6, 7, 8, 9, 10\}$). The computation time limit is 60 minutes for the solver and 1 minute for the other methods. About the computation time of the solver, instances of 5 jobs are solved on average in 1.7 seconds and 48 seconds for instances of 6 jobs. Only 7 instances of 7 jobs are solved by the solver before the time limit and no instance with 8 jobs is solved.

Table 12 gives the number of optimal solutions found by each method.

The average gap of the heuristic methods with the best solution found by all the methods are

Mehtod	$n=5$	$n=6$	$n=7$	$n=8$	$n=9$	$n=10$
MILP	10	10	10	9	0	2
$GRASP^{HEU}$	10	10	10	9	9	8
$GRASP^{OPT}$	10	10	10	9	10	8
$AG^{HEU,S,20}$	10	10	10	10	10	9
$AG^{HEU,S,5}$	10	10	10	10	10	10
$AG^{HEU,O,20}$	10	10	10	9	10	9
$AG^{HEU,O,5}$	10	10	10	9	10	7
$AG^{OPT,S}$	10	10	10	10	10	10
$AG^{OPT,O}$	10	10	10	10	10	10

Table 12: Number of optimal solutions found by each method

very small (less than 0.3%). For the exact resolution method, the gap is zero for instances of 8 jobs or less, and equal to 3.3% and 2.1% for instances of 9 and 10 jobs respectively.

7 Conclusion

This paper deals with an integrated production and distribution scheduling problem. The model that we consider has two main agents, the manufacturer and the 3PL provider and different costs are considered between them such as vehicle cost and inventory costs for the manufacturer, routing cost for the 3PL provider and tardiness penalty costs for both of them. In this context, we consider the scenario where the manufacturer dominates the negotiation with the 3PL provider, by imposing the number of vehicles used, the batch composition and the departure times of vehicles.

A Mixed Integer Linear Programming model is proposed for the manufacturer problem and the 3PL provider problem. Computational experiments show that this MILP can only solve very small instances in a reasonable computation time. The vehicle routing problem of the 3PL provider can be solved optimally by using the MILP model, because each instance is generally small (each problem instance for the 3PL provider corresponds to a single batch containing few jobs). Two metaheuristic algorithms are proposed to solve the manufacturer problem: a GRASP algorithm and a Genetic Algorithm. These methods are evaluated and computational results are provided. The results clearly show that the Genetic Algorithm outperforms the GRASP method on large instances.

The study presented in this paper can be developed and improved in three main directions: the model, the methods and the instances.

The model that is presented in this paper can be extended and other assumptions can be considered. For example, introducing vehicle capacities will introduce new constraints in the model and in the resolution methods, but without really increasing the difficulty of solving the problem. On the other side, introducing time windows for delivery of jobs or piecewise linear tardiness cost functions, will lead to a more realistic model, but will make the problem much more complicated.

Other resolution methods can also be proposed and compared to the Genetic Algorithm, such as Particle Swarm Optimization methods. An attempt to propose a Tabu Search algorithm was made, but the results were not satisfactory in comparison with the GA. But this track is certainly to be deepened.

Another key point for this study and for the comparison of methods is the data generation. It is not easy to generate data which does not lead to too easy instances (a single batch for all jobs or one batch per job), and at the same time which are not too far from a possible real life application. Maybe considering real life applications of the model will be a good starting point. For example, chemotherapy drugs production and delivery, with high inventory costs and low transportation costs, and milk production and delivery with low inventory costs and high transportation costs.

Finally, in a future research, we will also investigate other cooperation scenario as for example a scenario where the 3PL provider dominates the negotiation. This case appears in contexts of pooling 3PL providers. Another interesting direction is the case where the two agents cooperate in order to see if a global profit can be realized.

References

[Absi and Archetti, 2018] Absi, N. and Archetti, C. (2018). Comparing sequential and integrated approaches for the production routing problem. *European Journal of Operational Research*, 269(2):633–646.

[Absi et al., 2015] Absi, N., Archetti, C., Dauzère-Pérès, S., and Feillet, D. (2015). A Two-Phase Iterative Heuristic Approach for the Production Routing Problem. *Transportation Science*, 49(4):784–795.

[Agnetis et al., 2014] Agnetis, A., Ali, M., and Fu, L.-l. (2014). Coordination of production and interstage batch delivery with outsourced distribution. *European Journal of Operational Research*, 238(1):130–142.

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

[Cakici et al., 2014] Cakici, E., Mason, S. J., Geismar, H. N., and Fowler, J. W. (2014). Scheduling parallel machines with single vehicle delivery. *Journal of Heuristics*, 20(5):511–537.

[Chang et al., 2014] Chang, Y.-C., Li, V. C., and Chiang, C.-J. (2014). An ant colony optimization heuristic for an integrated production and distribution scheduling problem. *Engineering Optimization*, 46(4):503–520.

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

[Cheng et al., 2019] Cheng, B., Leung, J. Y., Li, K., and Yang, S. (2019). Integrated optimization of material supplying, manufacturing, and product distribution: Models and fast algorithms. *European Journal of Operational Research*, 277(1):100–111.

[Cheng et al., 2015] Cheng, B.-Y., Leung, J. Y., and Li, K. (2015). Integrated scheduling of production and distribution to minimize total cost using an improved ant colony optimization method. *Computers & Industrial Engineering*, 83:217–225.

[Devapriya et al., 2017] Devapriya, P., Ferrell, W., and Geismar, N. (2017). Integrated production and distribution scheduling with a perishable product. *European Journal of Operational Research*, 259(3):906–916.

[Farahani et al., 2012] Farahani, P., Grunow, M., and Günther, H.-O. (2012). Integrated production and distribution planning for perishable food products. *Flexible Services and Manufacturing Journal*, 24(1):28–51.

[Geismar et al., 2008] Geismar, J. H., Laporte, G., Lei, L., and 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.

[Han et al., 2019] Han, D., Yang, Y., Wang, D., Cheng, T. C. E., and Yin, Y. (2019). Integrated production, inventory, and outbound distribution operations with fixed departure times in a three-stage supply chain. *Transportation Research Part E*, 125:334–347.

[Hassanzadeh et al., 2016] Hassanzadeh, A., Rasti-barzoki, M., and Khosroshahi, H. (2016). Two new meta-heuristics for a bi-objective supply chain scheduling problem in flow-shop environment. *Applied Soft Computing Journal*, 49:335–351.

[Holland, 1975] Holland, J. (1975). Adaptation in natural and artificial systems. *Ann Arbor: University of Michigan Press.*

[Jia et al., 2019] Jia, Z.-H., Zhuo, X.-X., Leung, J.-Y., and Li, K. (2019). Integrated production and transportation on parallel batch machines to minimize total weighted delivery time. *Computers and Operations Research*, 102:39–51.

[Kergosien et al., 2017] Kergosien, Y., Gendreau, M., and Billaut, J.-C. (2017). Benders decomposition based heuristic for a production and outbound distribution scheduling problem with strict delivery constraints. *European Journal of Operational Research*, 262(1):287–298.

[Koç and Sabuncuoglu, 2017] Koç, U. and Sabuncuoglu, I. (2017). Coordination of inbound and outbound transportation schedules with the production schedule. *Computers & Industrial Engineering*, 103:178–192.

[Lacomme et al., 2018] Lacomme, P., Moukrim, A., Quilliot, A., and Vinot, M. (2018). Supply chain optimisation with both production and transportation integration : multiple vehicles for a single perishable product. *International Journal of Production Research*, 56(12):4313–4336.

[Li et al., 2005] Li, C.-L., Vairaktarakis, G., and Lee, C.-Y. (2005). Machine scheduling with deliveries to multiple customer locations. *European Journal of Operational Research*, 164(1):39–51.

[Marandani and Ghomi, 2019] Marandani, F. and Ghomi, S. (2019). Integrated multi-factory production and distribution scheduling applying vehicle routing approach. *International Journal of Production Research*, 57(3):722–748.

[Moons et al., 2017] Moons, S., Ramaekers, K., Caris, A., and Arda, Y. (2017). Integrating production scheduling and vehicle routing decisions at the operational decision level: A review and discussion. *Computers & Industrial Engineering*, 104:224–245.

[Neves-Moreira et al., 2018] Neves-Moreira, F., Almada-Lobo, B., Cordeau, J.-F., Guimarães, L., and Jans, R. (2018). Solving a large multi-product production-routing problem with delivery time windows. *Omega*, 86:154–172.

[Ouenniche and Boctor, 2001] Ouenniche, J. and Boctor, F. F. (2001). The two-group heuristic to solve the multi-product, economic lot sizing and scheduling problem in flow shops. *European Journal of Operational Research*, 129:539–554.

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

[Rohmer et al., 2015] Rohmer, S., Brain, A., Morin, P.-A., and Billaut, J.-C. (2015). A two-agent model for production and outbound distribution scheduling. In *27th European Conference on Operational Research (EURO 2015)*, Glasgow.

[Santander-Mercado and Jubiz-Diaz, 2016] Santander-Mercado, A. and Jubiz-Diaz, M. (2016). The economic lot scheduling problem: a survey. *International Journal of Production Research*, 54(16):4973–4992.

[Scholz-Reiter et al., 2010] Scholz-Reiter, B., Frazzon, E. M., and Makuschewitz, T. (2010). Integrating manufacturing and logistic systems along global supply chains. *CIRP Journal of Manufacturing Science and Technology*, 2(3):216–223.

[Tavares-Neto and Seido, 2019] Tavares-Neto, R. F. and Seido, M. (2019). An Iterated Greedy approach to integrate production by multiple parallel machines and distribution by a single capacitated vehicle. *Swarm and Evolutionary Computation*, 44:612–621.

[Ullrich, 2013] Ullrich, C. A. (2013). Integrated machine scheduling and vehicle routing with time windows. *European Journal of Operational Research*, 227(1):152–165.

[Viergutz and Knust, 2014] Viergutz, C. and Knust, S. (2014). Integrated production and distribution scheduling with lifespan constraints. *Annals of Operations Research*, 213(1):293–318.

[Wang et al., 2019] Wang, D., Zhu, J., Wei, X., Cheng, T., and Yin, Y. (2019). Integrated production and multiple trips vehicle routing with time windows and uncertain travel times. *Computers and Operations Research*, 103:1–12.

[Wang et al., 2020] Wang, S., Wu, R., Chu, F., and Yu, J. (2020). Variable neighborhood search-based methods for integrated hybrid flow shop scheduling with distribution. *Soft computing*, 24:8917–8936.