Model and Combinatorial Optimization Methods for Tactical Planning in Closed-Loop Supply Chains

Pierre Desport, Frédéric Lardeux, David Lesaint, Anne Liret, Carla Di Cairano-Gilfedder, Gilbert Owusu

Model and Combinatorial Optimization Methods for Tactical Planning in Closed-Loop Supply Chains

Pierre Desport
LERIA, Université d'Angers, France
Email: pierre.desport@univ-angers.fr

Frédéric Lardeux
LERIA, Université d'Angers, France
Email: frederic.lardeux@univ-angers.fr

David Lesaint
LERIA, Université d'Angers, France
Email: david.lesaint@univ-angers.fr

Anne Liret
BT France, France
Email: anne.liret@bt.com

Carla Di Cairano-Gilfedder
BT Group plc, UK
Email: carla.dicairano-gilfedder@bt.com

Gilbert Owusu
BT Group plc, UK
Email: gilbert.owusu@bt.com

***Abstract*—This paper proposes a constraint optimization model accepting different network topologies, site functions and distribution/transfer policies, applies it to a real case study or closed-loop supply chains in telecommunications services, and compares two approaches-a mixed-integer programming and a metaheuristics –to solve this problem. Its cost function is a linear combination of storage, transport, backorder and repair costs. We report experiments on pseudo random instances designed to evaluate plan quality and impact of cost weightings. Experiments validate approaches and compare our 2 methods. Finally, we discuss the possible extensions of the model to fit other specific cases and create interest among supply chain experts.**

I. Introduction

Reverse logistics and closed-loop supply chains (CLSC) have gained attention in recent years following growing concerns over the environmental and economic impact of products reaching end-of-life [3, 11]. Generally speaking, the aim is to recapture business value from used products through return, repair and disposal operations. CLSC extend traditional supply chains by coupling such operations with standard production and distribution operations. Figure 1 illustrates a CLSC that supports field service activities in the telecommunications domain. This CLSC distributes and recycles the many different parts (cables, network cards, IT equipment, etc.) used by engineers to carry out maintenance and repair activities on customer premises and telecommunications exchanges [2]. It is organized around a distribution center that replenishes field depots with healthy items (i.e., spare parts) which engineers then pick-up and use. Once their activities completed, engineers place faulty items in field depots which are then returned to the distribution center for repair following opposite routes in the supply chain.

Approaches to allocate inventory in CLSC differ depending on whether demand is stochastic or deterministic (see [6] for a comprehensive review). Field repair activities for instance are often ill-specified and the volume and type of healthy items required on each site cannot be forecast with certainty. Reactive and decentralized approaches prevail in such environments and inventory control models (economic order quantity, reorder point, order-up-to-level, ...) coupled with analytical or simulation methods are used to reduce service down-time and minimize inventory holding, transfer and repair costs [7, 9]. Conversely, service maintenance and provision activities which are scheduled in advance and involve replacing or installing specific parts allow the generation of accurate demand forecasts. In such settings, combinatorial optimization methods can be leveraged to design centralized planning systems that propose transfer and repair actions based on forecasts. We follow this approach and present a mixed integer programme (MIP) to address tactical distribution planning in CLSC under deterministic demand.

Given current inventory levels and a demand forecast over a discretized time horizon, the tactical distribution planning problem (TDPP) consists in generating a consistent and optimal plan of supply (i.e., transfer of healthy items), return (i.e., transfer of faulty items) and repair actions to be implemented on the different sites at the chosen times. Plans must primarily comply with the network routes and transfer schedules of the CLSC. These constraints define a space-time graph whose arcs encode all possible actions in terms of source site, destination site, start time and lead-time. A plan thus corresponds to the labelling of a subgraph, i.e., a choice of actions (arcs) and, for each action, the number of items at stake. Note that TDPP allows to model CLSC featuring multi-functional sites (i.e., demand fulfillment, repair capability, healthy/faulty inventory). It is thus applicable to a wide range of CLSC including multi-level networks with single inventory and repair warehouse centers [2], field depots equipped with remanufacturing capabilities [10] and warehouse centers meeting local demand [1].

Beyond topological and temporal constraints, plans are subject to flow constraints and a conventional inventory management policy. Flow constraints simply propagate planned actions on inventory levels by matching demand, backorders, and outgoing inventory with on-hand inventory and incoming inventory on each site. The inventory management policy is enforced on all sites and gives priority to the fulfilment of local demand. Specifically, the policy forbids backordering when inventory is available and forbids transferring or carrying inventory when orders are outstanding. Lastly, the objective function is a weighted sum of linear cost functions for storage,

transport, backordering and repair.

TDPP is closely related to the minimum convex network flow problem which is polynomial [8]. We show however that TDPP is NP-hard due to the dichotomic nature of the inventory management policy. The result is established by reducing the Subset Sum problem [4]. Alternatively to the MIP model, we propose a metaheuristics for TDPP and compare both approaches on a range of pseudo-random instances generated from a real case in the telecommunications domain. Results attest the superiority of the metaheuristics in terms of solution fitness (as measured by the objective function), run time and scalability.

The remainder of the paper is organised as follows. Section II presents the telecommunications case study and introduces the MIP model and computational complexity results for TDPP. Section III introduces the metaheuristics and its key components. Section IV reports experiments comparing the MIP and metaheuristics approaches. Section V discusses possible extensions to this work and Section VI concludes.

II. Problem Description

In this section, we first provide background on CLSC by discussing a real case study in the telecommunications domain. We then introduce the Tactical Distribution Planning problem (TDPP) that applies to a wide variety of CLSC network topologies. We present a MIP model for TDPP and show it is NP-hard by reducing the Subset Sum problem.

A. A case study in telecommunications

TDPP is based on a case study in the telecommunications sector where large quantities and varieties of items are required to support field service activities. Planned maintenance and repair activities involve replacing faulty/used items on sites with healthy items. Faulty/used items can be repaired in order to be reused later on and the role of the CLSC in this context is to globally manage return, repair and supply operations. The CLSC relies on a classical design with a tree topology organized around a distribution center (see Figure 1).

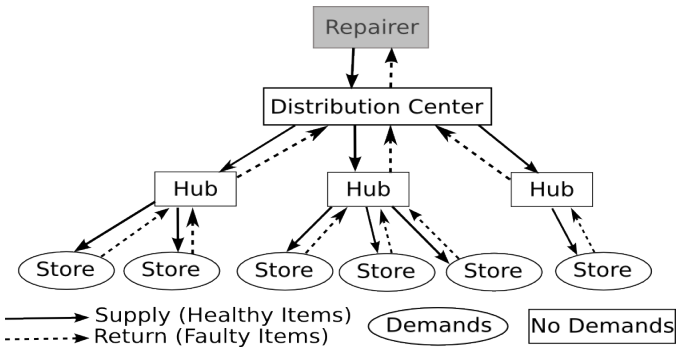

Fig. 1. A simple example of real case supply chain

Operationally, the distribution center (DC) is a inventory warehouse that replenishes stores (i.e., field depots) with healthy items through intermediate hubs (i.e., exchange points) and channels faulty items back to a repair center. Pick-up and delivery schedules are periodic with constant transportation lead times that are specific to each site. The same applies to the "repair loop" linking the distribution and repair centers. A demand is created for a healthy item whenever a faulty item is placed in a store. The demand is immediately satisfied if the store has items on-hand and prior on-site demands have been met. Otherwise, it is backordered and results in service down-time until healthy items are delivered. The faulty item is itself collected at the next pick-up date and returned to be stored at the DC. Once in the DC, it can be sent to the repair center to be fixed. The item is then available to be reinjected in the supply chain to meet another demand.

The management of the CLSC must fulfill conflicting objectives, namely, minimizing service down-time (i.e., costs of backordered or unsatisfied demands), inventory costs and costs relating to repairs and transfers (supplies of healty items and returns of faulty items). The current approach relies on inventory control policies to replenish stocks on each site. This decentralized reactive approach does not take advantage of demand forecasts when they are available. In the case of planned maintenance activities, such forecasts are precisely known (i.e., the items to replace are known for each scheduled activity) and a centralized proactive planning approach is then relevant. Equally, demand forecasts can be generated in the case of field repair activities based on historical data, seasonal trends, etc. Such forecasts are subject to uncertainties but a robust planning approach may still be considered as discussed subsequently.

We propose a periodic planning approach to determine return, repair and supply actions ahead of time based on a demand forecast. Figure 2 depicts the architecture of the proposed planning system. The principle is to generate a

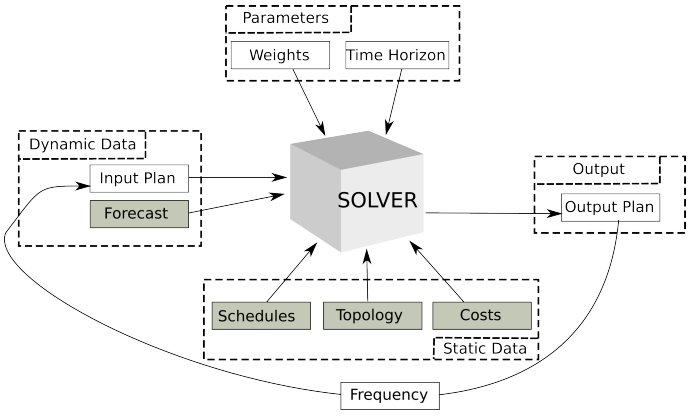

Fig. 2. The complete planning system

plan at regular time intervals. Each plan is based on current inventory levels and a demand forecast for the given time horizon. A static planning module is used to generate each plan by solving a combinatorial optimization problem. This module takes as input static data (CLSC topology, schedules, unit costs), dynamic data (current forecast, inventory levels,

...) and user-defined parameters (weights, time horizon). The optimization model is based on a weighted objective function allowing users to control the balance between the different costs (backorders, inventory, repair, transfer). The planning frequency is the other user-defined parameter. Altogether, these parameters provide the flexibility to implement robust distribution plans.

In the remainder, we restrict our attention to the static planning module and the corresponding tactical distribution planning problem (TDPP). We introduce a generic combinatorial optimization model for TDPP which is applicable to a wide variety of CLSC beyond this case study.

B. The Tactical Distribution Planning Problem

TDPP is based on a finite ordered set of time buckets T (the time horizon) and a finite set of sites L. The possible pairings of sites and time points ($\#L \times \#T$) define the nodes of a space-time graph G^* whose arcs represent the routes interconnecting sites through the time horizon. Specifically, G^* can be divided into 3 subgraphs G^h, G^f and G^r, where G^h models the supply routes for healthy items, G^f the return routes for faulty items, and G^r the repair routes. These subgraphs compile the topology and the transfer/repair schedules of the CLSC. Note that this model is generic and can accommodate any topology and schedule. Figure 3 is an example of space-time graph.

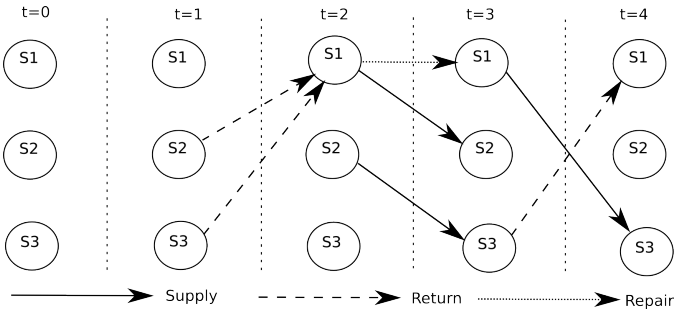

Fig. 3. An example of space-time graph

It defines the structure of the problem which the sets of constants, variables, constraints and cost terms are based upon. Constants and variables are typically $\#L$-$\#T$ integer matrices domains that denote numbers of items or cost values. We introduce the following notation x_z^y where x is the name of the constant (or variable), y the state (i.e., healthy, faulty, healthy stock,-...), z the space-time parameters (i.e., locations, time buckets). Table I presents the constants and the variables of the problem. Constants include the demand forecast d that defines the demand level for each site and time bucket of the time horizon, the unit costs μ^* (inventory , backorders, transfers, repairs), the user-defined costing weights ω^* and the initial state I. The initial state I is an assignment of all the variables at time bucket 0. The variables are separated in two categories: the decision variables (in bold style) and the auxiliary variables. The decision variables are related to actions and provide information on supply m^h, return m^f and repair m^r. They are associated with the arcs represented on the space-time graph such that if an arc does not exist then the corresponding value is set to 0. Auxiliary variables are directly computed from the decision variables. They not only describe the state supply chain for each time bucket and site but also detail the different costs for this particular state. Given a particular time bucket t and a particular site l, then $s^h_{l,t}$ (resp. $s^f_{l,t}$) details the inventory levels for healthy (resp. faulty) items and $b_{l,t}$ the pending backorders that are still to be met. Variables $p^h_{l,t}$ and $p^f_{l,t}$ explicit the domain of potential values. These variables represent respectively the balance between the amount of healthy items available to be sent out and the remaining backorders and the quantity of faulty items available to be sent out from location l at time bucket t.

TABLE I
CONSTANTS AND VARIABLES

Constants	
demand forecast	$d : L \times T \to \mathbb{N}$
inventory costs	$\mu^{s^h}, \mu^{s^f} : L \times T \to \mathbb{N}$
backorder costs	$\mu^b : L \to \mathbb{N}$
transfer costs	$\mu^{m^h}, \mu^{m^f} : L \times L \to \mathbb{N}$
repair costs	$\mu^{m^r} : L \to \mathbb{N}$
weights	$\omega^s, \omega^b, \omega^{m^{hf}}, \omega^{m^r} \in \mathbb{N}$
initial state	I
Variables	
inventory	$s^h, s^f : L \times T \to \mathbb{N}$
backorders	$b : L \times T \to \mathbb{N}$
healthy potential	$p^h : L \times T \to \mathbb{Z}$
faulty potential	$p^f : L \times T \to \mathbb{N}$
transfers	$m^h, m^f : L \times L \times T \times T \to \mathbb{N}$
repairs	$m^r : L \times T \times T \to \mathbb{N}$
costs	$c, c^s, c^b, c^{m^{hf}}, c^{m^r} \in \mathbb{N}$

We define a plan (solution of TDPP) as a set of supply, return and repair actions. It can be represented on the space-time graph as the set of edges labelled with a positive value. Labels on arcs may be viewed as item quantities being transferred or repaired between two sites. A plan precisely describes for each route between two sites l, l', from time bucket t to time bucket t' the quantity of healthy ($m^h_{l,l',t,t'}$) and faulty ($m^f_{l,l',t,t'}$) items transferred. Likewise, it also defines the quantity of items sent to repair at time bucket t from a particular site l and retrieved at time bucket t' ($m^r_{l,t,t'}$). Note that this operation is topologically restricted to link a site with itself.

We also introduce cost variables to measure the quality of a solution. They are all evaluated on the same financial scale to allow a comparison. The quality is defined through 4 main objectives: backorders (c^b), inventory (c^s), transfers ($c^{m^{hf}}$) and repairs (c^{m^r}).

a) Core of the problem:

$$\forall l \in L, t \in T, \qquad s^f_{l,t} = p^f_{l,t} - \sum_{t' \in T} m^r_{l,t,t'} - \sum_{\substack{l' \in L, \\ t' \in T}} m^f_{l,l',t,t'} \tag{1a}$$

$$\forall l \in L, t \in T, \qquad \sum_{t' \in T} m^r_{l,t,t'} + \sum_{\substack{l' \in L, \\ t' \in T}} m^f_{l,l',t,t'} \leq p^f_{l,t} \tag{1b}$$

$$\forall l \in L, t \in T \backslash \{0\}, \quad p^f_{l,t} = s^f_{l,t-1} + \sum_{\substack{l' \in L, \\ t' \in T}} m^f_{l',l,t',t} + d_{l,t} \tag{1c}$$

Equations (1a) to (1c) characterize faulty items management. They define the stock requirement constraint (i.e. an item can be transferred only if the stock is positive). Specifically, Equation (1c) represents the quantity of faulty items that can be returned from a particular site l on a particular time bucket t. They also describe the link between demands and faulty items. A faulty item is added to the stock when a demand occurs.

$$\forall l \in L, t \in T, \qquad s^h_{l,t} = max(0, p^h_{l,t} - \sum_{\substack{l' \in L, \\ t' \in T}} m^h_{l,l',t,t'}) \tag{2a}$$

$$\forall l \in L, t \in T, \qquad \sum_{\substack{l' \in L, \\ t' \in T}} m^h_{l,l',t,t'} \leq max(0, p^h_{l,t}) \tag{2b}$$

$$\forall l \in L, t \in T \backslash \{0\}, \quad p^h_{l,t} = s^h_{l,t-1} + \sum_{l' \in L, t' \in T} m^h_{l',l,t',t} + \sum_{t' \in T} m^r_{l,t',t} - b_{l,t-1} - d_{l,t} \tag{2c}$$

$$\forall l \in L, t \in T, \qquad b_{l,t} = \max(0, -p^h_{l,t}) \tag{2d}$$

Equations (2a) to (2d) detail healthy items management. Again, they define the stock requirement constraint. The potential value for a particular site at a particular time bucket is also defined. It is important to notice that these equations reflect a management rule that ensure the consumption of available items on sites if demands occur on the same day.

b) Fitness function:

$$c^s = \sum_{l \in L, t \in T} (s^h_{l,t} \times \mu_l^{s^h} + s^f_{l,t} \times \mu_l^{s^f}) \tag{3a}$$

$$c^b = \sum_{l \in L, t \in T} b_{l,t} \times \mu^b_l \tag{3b}$$

$$c^{m^{hf}} = \sum_{\substack{l,l' \in L, \\ t,t' \in T}} (m^h_{l,l',t,t'} \times \mu^{m^h}_{l,l'} + m^f_{l,l',t,t'} \times \mu^{m^f}_{l,l'}) \tag{3c}$$

$$c^{m^r} = \sum_{\substack{l \in L, \\ t,t' \in T}} (m^r_{l,t,t'} \times \mu_l^{m^r} l') \tag{3d}$$

$Global\ min$:

$$c = \omega^s \times c^s + \omega^b \times c^b + \omega^{m^{hf}} \times c^{m^{hf}} + \omega^{m^r} \times c^{m^r} \tag{3e}$$

These equations allow to compute the different costs of the supply chain. Note that Equation (3e) representing the global fitness is a weighted function that will allow to emulate different planning strategies by enforcing some of the objectives.

C. Reduction to the Subset Sum problem

Theorem 1. TDPP *is NP-Hard.*

Proof. We demonstrate that our problem is NP-Hard by reducing the Subset Sum Problem to the TDPP in polynomial time. The Subset Sum Problem is known to be NP-Complete [4] and it is defined by only one question: Given a finite multiset of x positive integers K $[y_1..y_x]$ and a positive integer B, is there a subset $K' \subseteq K$ such that $\sum_{k \in K'} y_k = B$? As we can check a solution in polynomial time, this reduction demonstrates that the TDPP is NP-Hard. This reduction is based on the implicit constraint induced by equations (2d), (2c), (2a) and (2b) that ensure the consumption of available items on sites if demands occur on the same day. It allows us to assure a non-separation of the outflow on the origin sites in order to have enough items in the supply chain to meet the demands in the stores.
We propose the following reduction:

1) The time horizon T is set to 3 time buckets.
2) Create 3 distinct sets of sites $L1$, $L2$ and $L3$ such that $L1 = [1..x]$, $L2 = [x+1..3x]$ and $L3 = [3x+1..3x+2]$. We define $L = L1 \bigcup L2 \bigcup L3$.
3) We set up the initial stock levels such that $\forall l \in L1,\ s^h_{l,0} = y_l + 1$.
4) We create the demand forecast such that $\forall l \in L2,\ b_{l,1} = 1$, $b_{3x+1,3} = B$ and $b_{3x+2,3} = \sum_{k=1}^{x} y_k - B$.
5) We define the associated costs such that $\forall l \in L2,\ \mu^b_l = 0$, $\mu^b_{3x+1} = 1$ and $\mu^b_{3x+2} = 1$.
6) $\forall l \in L1$, create 2 healthy transfers $m^h_{l,l+x,1,2}$ and $m^h_{l,l+2x,1,2}$.
7) $\forall l \in L1$, create 2 transfers $m^h_{l+x,3x+1,2,3}$, $m^h_{l+2x,3x+2,2,3}$.
8) All the unmentioned costs are assumed to be 0.

If the $fitness$ is equal to 0 (i.e. we can meet all the demands occurring on $l = 3x+1$ and $l = 3x+2$) then it exists a subset $K' \subseteq K$ such that $\sum_{k \in K'} y_k = B$. □

Example. Figure 4 shows an example of transformation with $K = [8; 18; 9; 2]$ and $B = 17$.

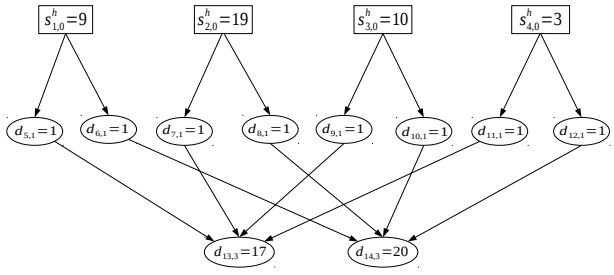

Fig. 4. Subset Sum to TDPP for $K = [8; 18; 9; 2]$ and $B = 17$

III. METAHEURISTICS

Firstly, we have proposed a Mixed Integer Programming (MIP) model to solve the TDPP problem. The MIP is directly derived from the equations presented in Section II. This exact method is working well but when the number of sites and the planning horizon grows up, it quickly has difficulty providing a solution in acceptable time (see Section IV).

In such case, metaheuristics are known to take up the slack [5]. They potentially allow us to find a very good solution in a reasonable time. We propose a metaheuristics denoted BIS (Best Improving Sequence). Classically, a metaheuristics is defined by: a solution, a neighborhood function, an evaluation function, and of course a choice of heuristics.

Search space and evaluation function

In our case, a solution is a plan which is a set of actions between adjacent sites in the space-time graph G^* realized in a given time horizon. The search space is defined by: $\mathcal{S} = 2^{\mathcal{M}}$ where $\mathcal{M}$ is the set of possible actions given by the topology. In order to compare each solution an evaluation function ($fitness : \mathcal{S} \rightarrow \mathbb{R}$) is defined using the cost function given by Equation (3e).

Neighborhood

The neighborhood function explores the solutions reachable by a $move$ ($move$: $\mathcal{N} : \mathcal{S} \rightarrow 2^{\mathcal{S}}$). A $move$ between 2 plans corresponds to a sequence of unitary actions (only one item is transferred or repaired each time) improving global objective function.

We introduce the notion of $sequence$. A sequence is a series of actions possibly empty. Actions included in a sequence are constrained temporarily and geographically such that given a sequence seq and 2 consecutive actions $m_{l1,l2,t1,t2}, m'_{l3,l4,t3,t4} \in seq$, we have $l2 = l3$ and $t2 \leq t3$. The type of action is also important. Supply and repair actions can only be followed by supply actions whereas a return can be followed either by another return or by a repair. For instance, a sequence could be compounded by a unique supply action or by a series of return, repair and supply actions. The maximum size of a sequence is $T - 1$ actions.

Local Search

A plan can be viewed as a set of sequences where all sequences are strongly interdependent. It is then computationally expensive to deduce the global impact of a sequence of actions. We then propose a method that evaluates locally the impact of a sequence. This method is denoted $improve(m, P, seq)$ with $m \in G^*$, a plan P and a sequence of actions seq.

It returns the maximum improvement value that a sequence initiated by seq and followed by the action m ($seq \bigcup m$) could bring to the plan P. Note that the number of potential sequences evaluated with this method can increase very quickly. The computation of the improvement value only takes into account backorder, inventory, repair and transfer costs. Figure 5 describes an example of $improve(m, P, seq)$. In that case, 5 sequences of actions are executable after m and $improve(m, P, seq)$ will return the maximum positive improvement value that one of these sequences could bring and 0 otherwise. The computation of the improvement value could seem expensive but this cost is in fact limited by to the usage of a data structure allowing to easily compute the improvement value of a sequence. This data structure is a delta matrix that details the improvement value of the different sequences based on actions. The matrix is created antichronologically and built around a propagation algorithm that consists in using subsequences to avoid the exploration of all the possible sequences. We also notice that even if the first computation of the matrix is costly, we only recompute it partially later on.

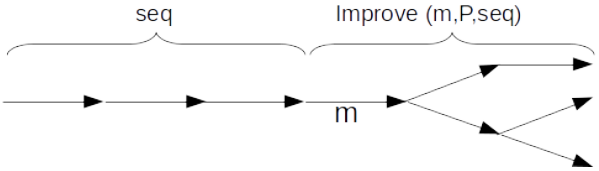

Fig. 5. Example of $improve(m, P, seq)$

This $improve$ function allows to define the graph $G|_{seq}$. Given a plan P and a sequence seq as this graph represents the doable actions (satisfying the stock constraints) $m \in G^*$ following the sequence seq and with $improve(m, P, seq) > 0$. For instance, if the last action m of a sequence seq is a supply action from site 1 to site 2 completed on time bucket 10 then $G|_{seq}$ include all the possible supply actions taking place from site 2 after time bucket 10 whose improvement value is positive. Algorithms 1 and 2 describe the process of selection and application of a sequence and Algorithm 3 presents the BIS method. We can notice that we use the best improvement selection (i.e. we select the action m such that $improve(m, P, seq)$ is maximum). It ensures us to only apply very efficient sequences of actions and provides good results in most of the cases. However, in some cases this selection will lead to a local optimum. It can be caused by different factors such as a complex topology, asynchronous schedules or stock positioning. Therefore, even if this selection process is good to drive the search, it has to be paired with a diversification process. This diversification process is represented by the α probability that allows us to explore the search space by selecting "less improving" sequences. This diversification probability is fixed to 1 in the first iteration of the BIS Algorithm and then decreases following a geometrical series. It allows to bifurcate at each step of the creation of the sequence such that with $\alpha = 0$, we can explore all the sequences seq of the search space initiated by m with $improve(m, P, seq) > 0$.

Algorithm 1 Action Selection

Input: A Graph G, A Plan P, A Sequence seq, A Probability α

Output: An Action

With a probability α, return $argmax_{m \in G}(improve(m, P, seq))$; ▷ In case of equality, m is selected randomly amongst the best ones

With a probability $1 - \alpha$, Return $m \in G$;

Algorithm 2 Sequence Computation

Input: A Graph G, A Plan P, A Sequence seq, A Probability α

Output: A sequence

while $G|_{seq} \neq \emptyset$ **do**
    Select an action m by using Algorithm 1 with $G|_{seq}$, P, seq and α;
    Add m to seq;
end while
Return seq;

IV. Experimentations

In this section we present some experiments to prove the validity of our method. The results presented here are based on a particular type of instance but experiments also have been conducted on other topologies (hub and spoke, complete graph, etc.) with similar results.

A. Experimental Settings

Experiments are carried out on pseudo-random instances. In this configuration transfer costs are inclusive and do not depend on the quantity of items consequently they can be safely ignored. The objective is to assess the impact of the key features (demand distribution, stock levels, etc.) of problem instances on the fitness of the plan. To this end, but also to reduce bias in the analysis, we generate all instances using the same topology, time horizon, transfer schedules, lead-times, transition period, and demand pattern. Settings are chosen consistently with data emanating from our case study.

Algorithm 3 The BIS algorithm

Input: The constants of the problem, A number of iterations,

Output: A Plan

Set a probability α to 1
while Number of iterations is not reached **do**
    Create an empty Plan P;
    Create an empty sequence seq;
    while $G|_{seq} \neq \emptyset$ **do**
        Update seq by computing a sequence with Algorithm 2 with $G|_{seq}$, P, seq and α;
        Execute the sequence seq and add it to P;
        Empty seq;
    end while
    Evaluate P;
    α decreases following a geometrical series
end while
return the best Plan

The topology is the one presented in Figure 1. We have generated two sizes of instances. The supply network includes 1 distribution center, 9 (resp. 33) hubs and 15 (resp. 66) stores. Each hub deserves an exclusive set of stores. The DC serves all hubs and is the unique repair center. The time horizon is initialized to 60 day-buckets. Transfers between connected sites all take 1 day. To fit more precisely the real case, the repair time is set to 3 days. Sites also have identical pickup and delivery frequencies (every 5 days) with synchronous schedules (identical delivery and pickup days) hence transport cost is ignored. We introduce a "transition period" set to the first 7 days to coincide with the earliest delivery time for a store. Demands raised within this period are supposedly addressed by previous planning decisions. Hence, the results only depend on our model without any interferences from previous choices and allow us to devise confidently about the quality of our method. For this reason, instances have no demands within the transition period and ongoing transfers all complete within that time window. Similarly and consistently with the need to avoid side-effects induced by past decisions, instances have no residual faulty items.

Instances also share the same demand distribution pattern. Specifically, all stores have a single demand request that recurs every 5 days. However, demands may occur on different days for different sites. The actual variability amongst instances comes from the volume and allocation of healthy stocks which we generated using different schemes. As for stock allocation, healthy items are either all placed in the DC (scheme `DC`), all placed randomly in the stores (scheme `Stores`) or evenly distributed between the DC and the other stores (scheme `Mix`). For the last two schemes, allocation to stores is performed randomly using a uniform distribution law. In terms of volumes, the total number of items across all sites is either set to 100% of the total demand (scheme `High`), 50% (scheme `Med`) or 0% (scheme `Low`). Combining the two schemes yields seven classes of instances (stock allocation is irrelevant for scheme `Low`). The unit costs are configured to respect the hierarchy used in the real case. Solving the backorders is the ultimate goal of the supply chain; thus its cost is set to 1000 ; repairing an item costs 100, store an item in a store or in a hub costs 10 and store an item in a DC costs 1. The MIP model is implemented using CPLEX 12.6 and tests run on a *i5-3380M @ 2.90GHz* architecture. We allocate a maximum running time of 1 hour as the model is ultimately supposed to be running on a sliding window independently on many thousands of items. In case the maximum time is reached, we retrieve the lowest bound found by the model.

B. Experimental Results

Table II presents the results. The first two columns denote the schemes characterising each class of instances and the third column denotes the cost weighting used for the test. Weights are represented as a triplet $< \omega^b, \omega^{m^r}, \omega^s >$. Ex-

periments have been run on all combination of weightings. The remaining columns provide for the 2 sizes of instances information depending on the method used: *Mip-Fitness* is the best bound retrieved by the MIP in 1 hour, *Mip-Time* the run time of the MIP in milliseconds, *Meta-Fitness* the mean value of the metaheuristics on 100 iterations, *Meta-s.d.* the standard deviation of the metaheuristics, and *Meta - Time* the average run time of the metaheuristics in milliseconds. Note that if *Mip-Time* indicates $'-'$, it means that the MIP has reached the time limit. A bold value in the column *Meta-Fitness* means that either the MIP did not return a bound or that the bound returned is worse than the average value retrieved by the metaheuristics.

As expected, initial stock levels strongly influence service levels. Scheme `High` dominates scheme `Med` when the weight on backorders is activated for any given distribution scheme and weighting. Likewise, `Med` dominates `Low`. Stock distribution also plays a key role. Comparing when the weight on backorders is activated again, distribution `DC` dominates `Mix` which dominates `Stores` for any given stock volume scheme and weighting. It highlights the importance of well locating items. This is particularly true in our case as healthy items cannot be sent back from the stores to the DC. Thus, bad positioning the items is a major risk and is highly depending on the quality of the forecast. When looking at the instances with weighting $\langle 1, 1, 1\rangle$ we notice that the results returned by the scheme `High` and the scheme `Med` are close. It gives us the hint that the optimum quantity needed to solve all the backorders and limit the inventory cost is between these two schemes. The distribution also affects the running time. For a particular scheme, distribution `DC` is slower than `Mix`, itself slower than `Stores`. Indeed, the more items are positioned in stores in the initial situation the less transfers you can perform. Hence, it indirectly cuts the search space and impacts the solving time.

We now compare the MIP with the metaheuristics. We first denote that the average value returned by the metaheuristics is equal or very close to the best bound retrieved by the MIP. We also point out that in all cases the metaheuristics reaches at least once the best bound value. The standard deviation remains low and demonstrates the stability of our method. The results validate the well-functioning of our metaheuristics on studied instances. In terms of scalability and running time, the metaheuristics appears to be the best method. In most of the cases, the metaheuristics is at least 100 times faster than the MIP model and consequently a complete run of 100 iterations still outperformed the MIP. We note that the bigger the instances are, the harder it is for the MIP to complete. Indeed, it reaches the time limit in 10 cases on instance 25_165 and in 18 cases on instance 100_726. We also remark that on instance 100_726 the MIP fails to return a bound in 3 cases and in 1 case returns a higher bound than the mean of the metaheuristics. It confirms that the MIP faces a scalability issue due to the increasing number of possible actions. On the contrary, the metaheuristics reaches very good quality solutions in a reasonable time.

V. Discussion

The TDPP is a generic problem and can easily be updated to include a large variety of extensions. Supply chain management problems are concrete problems that generally do not fit immediately a particular model as it is necessary to include some specific management rules. We sketch a non-exhaustive list of extensions that seems relevant in a supply chain management context.

The TDPP is limited to 3 basic actions (supply, return and repair) such that the number of items moving or installed in the supply chain is constant. Other actions can be added to the model to increase or decrease the quantity of items. For example, it is reasonable to include selling and purchasing actions to be able to meet a surplus of demands or oppositely to evacuate unused items. Equally, our model is currently restrained to maintenance or replacement operations. We can include other procedures such as installation (i.e. supply an item to a new site) and dismantling (i.e. the faulty items are not replaced). These procedures can be easily incorporated to our model by adding 2 matrices of constants.

Another common constraint in supply chain management problems is the capacity constraint. In the TDPP, capacity constraints could be applied on transfer, inventory and repair quantities. Matrices and inequations can be included to easily model these constraints but we intuitively understand that they add some complexity to the problem. The model may also be extended to model the notion of safety stock. It can be represented either with a strict capacity constraint or by adding an objective to the fitness function. The second method provides the possibility to prioritize objectives (i.e. meet the demands and if possible keep the stock level over the safety stock level).

Our current work includes the ability to implement particular management strategies by defining strict relation order between the weights such that a policy can be implemented thanks to the weights regardless the different costs. We are also investigating the impact of the frequency parameter to build some robust plans and manage uncertainties. Preliminary results show that a high frequency can handle uncertainties, especially with flexible topologies. However, we have to study the consistency of the following plans and determine when it is interesting to use a proactive or a reactive method.

VI. Conclusion

This work presented the Tactical Distribution Planning problem TDPP. We proved its complexity by reducing it to the NP-Complete Subset Sum problem. We proposed a generic and applicable model to problems featuring different topologies, schedules or business objectives. From this formulation, we introduced a MIP model and a dedicated metaheuristics (BIS) to solve the TDPP. We detailed a concrete case of this problem and tested the MIP model and the metaheuristics against instances strongly inspired from this case. Experiments demonstrate the scalability issue encountered by the MIP model and the quality of the metaheuristics on these instances. They show the importance of well positioning items to get a

TABLE II
COMPARISON MIP VS METAHEURISTICS

			25_165					100_726				
		Weights	MIP		BIS			MIP		BIS		
			Fitness	Time	Fitness	s.d.	Time	Fitness	Time	Fitness	s.d.	Time
High	DC	<1,0,0>	0	4554	0	0	17	0	57000	0	0	220
		<0,1,0>	0	3205	0	0	3	-	-	**0**	0	42
		<0,0,1>	22725	4175	22726	6	8	99990	20500	99998	14	121
		<0,1,1>	23085	4555	23086	2	6	101574	23100	101581	13	80
		<1,1,0>	0	3562	2	14	15	0	32480	29	57	221
		<1,0,1>	25500	4554	25658	199	20	112200	-	113073	702	280
		<1,1,1>	25860	5870	25964	163	18	113784	-	114258	382	256
	Mix	<1,0,0>	0	2125	0	0	11	0	26220	0	0	147
		<0,1,0>	0	2085	0	0	3	0	10780	0	0	40
		<0,0,1>	31253	1182	31253	0	5	133862	2980	133869	50	70
		<0,1,1>	31613	1008	31613	1	3	135446	2840	135446	0	41
		<1,1,0>	100	3134	113	33	10	0	29500	12	35	143
		<1,0,1>	33423	2696	33573	193	14	143562	-	144348	587	191
		<1,1,1>	33880	4134	33959	169	12	145146	-	145508	389	164
	Stores	<1,0,0>	0	1072	0	0	5	27000	2730	27000	0	69
		<0,1,0>	0	820	0	0	3	0	2070	0	0	40
		<0,0,1>	58005	943	58005	0	6	256128	2030	256128	0	62
		<0,1,1>	58365	791	58365	0	3	257712	2030	257712	0	40
		<1,1,0>	1700	1329	1708	27	4	35600	3430	35625	48	71
		<1,0,1>	58565	1332	58585	59	6	285983	3510	286053	108	95
		<1,1,1>	60574	1423	60590	59	5	295909	3930	295995	137	78
Med	DC	<1,0,0>	0	5105	0	0	15	0	38000	0	0	221
		<0,1,0>	0	15161	0	0	3	-	-	**0**	0	40
		<0,0,1>	18326	7597	18327	2	8	80751	35410	80757	11	110
		<0,1,1>	18686	6863	18687	2	6	82335	31500	82343	18	82
		<1,1,0>	8300	-	8306	24	19	36300	-	36336	67	221
		<1,0,1>	21101	6321	21275	212	20	92961	-	93839	691	285
		<1,1,1>	29512	-	29650	192	19	129756	-	130237	379	252
	Mix	<1,0,0>	0	3120	0	0	13	0	25680	0	0	185
		<0,1,0>	0	9871	0	0	3	0	120500	0	0	40
		<0,0,1>	19407	1620	19408	6	6	88669	18250	88675	29	85
		<0,1,1>	19767	1569	19767	1	4	90253	14240	90272	55	54
		<1,1,0>	8300	-	8303	17	13	36300	-	36315	38	188
		<1,0,1>	22112	4543	22323	278	17	100339	-	101192	608	237
		<1,1,1>	30523	-	30623	130	15	-	-	**137643**	451	215
	Stores	<1,0,0>	31000	2542	31000	0	10	93000	11730	93000	0	148
		<0,1,0>	0	1536	0	0	3	0	4480	0	0	39
		<0,0,1>	25895	1261	25895	0	4	114648	6051	114648	0	63
		<0,1,1>	26255	1514	26255	0	3	116232	3900	116232	0	43
		<1,1,0>	39300	-	39303	17	12	129300	-	129310	36	148
		<1,0,1>	59100	3458	59141	78	13	217263	19240	217482	228	171
		<1,1,1>	67511	-	67564	81	12	254058	-	254342	281	163
Low	-	<1,0,0>	2625000	-	2625090	319	6	11550000	-	11550700	1923	94
		<0,1,0>	0	744	0	0	3	0	2000	0	0	40
		<0,0,1>	15045	1291	15045	0	4	66198	8410	66198	0	65
		<0,1,1>	15405	1408	15405	0	3	67782	8140	67782	0	40
		<1,1,0>	2635500	-	2635540	196	6	11603200	-	**11596400**	409	96
		<1,0,1>	2641620	-	2641710	289	7	11623128	-	11623500	1063	104
		<1,1,1>	2652165	-	2652160	2	7	11669526	-	11669900	1285	102

good proactive plan. We also pointed out the next steps of our work and presented a large variety of extensions that can be interesting for companies and supply chain experts.

REFERENCES

[1] K.D. Cattani, F.R. Jacobs, and J. Schoenfelder. Common inventory modelling assumptions that fall short: arborescent networks, poisson demand, and single echelon approximations. *Journal of Operations Management*, 29(5):488–499, 2011.

[2] P. Desport, F. Lardeux, and D. Lesaint. Tactical inventory planning in the telecommunications service industry : a case study. *Roadef Marseille France*, 2015.

[3] M. Fleischmann, J.M. Bloemhof-Ruwaard, R. Dekker, E. Van der Laan, J.A. Van Nunen, and L.N. Van Wassenhove. Quantitative models for reverse logistics: A review. *European journal of operational research*, 103(1):1–17, 1997.

[4] M.R. Garey and D.S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

[5] M. Gendreau and J-Y. Potvin. *Handbook of Metaheuristics*. Springer Publishing Company, Incorporated, 2nd

edition, 2010.

[6] K. Govindan, H. Soleimani, and D. Kannan. Reverse logistics and closed-loop supply chain: A comprehensive review to explore the future. *European Journal of Operational Research*, 240(3):603 – 626, 2015.

[7] V.D.R Guide, T.P. Harrison, and L.N. Van Wassenhove. The challenge of closed-loop supply chains. *Interfaces*, 33(6):3–6, 2003.

[8] G.M. Guisewite and P.M. Pardalos. Minimum concave-cost network flow problems: Applications, complexity, and algorithms. *Annals of Operations Research*, 25(1):75–99, 1990.

[9] A. Gupta, P.C. Tewari, and R.K. Garg. Inventory models and their selection parameters: a critical review. *Int. Journal of Intelligent Enterprise*, 2(1):1–20, 2013.

[10] K. Lieckens, P.J. Colen, and M. Lambrecht. Optimization of a stochastic remanufacturing network with an exchange option. *Decision Support Systems*, 54(4):1548–1557, 2013.

[11] J.R. Stock. *Development and Implementation of Reverse Logistics Programs*. Council of Logistics Management, 1992.