CIRRELT
1971 – 2021

CIRRELT-2021-15

An Iterative Decomposition Algorithm for Flexible Two-Echelon Network Design

Martin P. Kidd
Maryam Darvish
Leandro C. Coelho
Bernard Gendron

April 2021

Document de travail également publié par la Faculté des sciences de l'administration de l'Université Laval, sous le numéro FSA-2021-002

Bureau de Montréal
Université de Montréal
C.P. 6128, succ. Centre-Ville
Montréal (Québec) H3C 3J7
Tél : 1-514-343-7575
Télécopie : 1-514-343-7121

Bureau de Québec
Université Laval,
2325, rue de la Terrasse
Pavillon Palasis-Prince, local 2415
Québec (Québec) G1V 0A6
Tél : 1-418-656-2073
Télécopie : 1-418-656-2624

An Iterative Decomposition Algorithm for Flexible Two-Echelon Network Design

Martin P. Kidd[1], Maryam Darvish[2,3], Leandro C. Coelho[2,3,*], Bernard Gendron[2,4]

1. Department of Technology, Management and Economics, Technical University of Denmark, Lyngby, Denmark
2. Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)
3. Department of Operations and Decision Systems, Université Laval, Québec, Canada
4. Department of Computer Science and Operations Research, Université de Montréal, Montréal

Abstract. This paper proposes a mixed integer programming model for a flexible two-echelon capacitated, multi-commodity, and multi-period network design problem. The model integrates several decisions of a supply chain and simultaneously plans production, inventory, location, and distribution. We consider a set of plants supplying intermediate facilities through which products are shipped directly to final customers. The model also includes real-world features of flexible delivery due dates and flexible location of the intermediate facilities. An iterative decomposition algorithm is proposed to solve this rich integrated problem. Three versions of the model are solved iteratively: two relaxation-based models and one restricted model. Solving the restricted model acts as a neighborhood search around the current best solution. Solutions obtained by the relaxation-based models guide the neighborhood search by identifying the binary variables to fix and at the same time promote diversity in the search. The results obtained by our computational experiments highlight the efficiency of the proposed method. Furthermore, managerial insights are presented on the utilization of flexibility in the obtained network design solutions, and how it relates to economies of scale.

Keywords: Integrated logistics, delivery due date, flexible location decision, distribution, decomposition algorithm.

Acknowledgements. This project was partly funded by the Natural Sciences and Engineering Council of Canada (NSERC) under grants 2020-00401, 2019-00094, and 2017-06054. This support is greatly acknowledged. We thank Serge Bisaillon with his help with an initial implementation of our algorithm.

---

* Corresponding author: Leandro.Coelho@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2021

1 Introduction

Supply chain network design (SCND) is an important component for efficient operations of supply chains. The topic is also among the most discussed ones in academia and practice. In order to design an efficient network, one needs to decide on the location, capacity of facilities, and flow of products to satisfy all customer demands at minimum cost [5, 10]. In many industrial applications of SCND, a multi-echelon supply network is considered, where products are first moved from central locations to a set of intermediate facilities and then, from these facilities, shipments to the final customers take place [16, 23].

SCND encompasses a broad spectrum of decisions, i.e., strategic, tactical, and operational, from all decision-making levels. Traditionally, facility location planning and network design problems have been treated as long term strategic decisions. However, in the modern business environment, owing to emerging technologies, boundaries between strategic and tactical decisions are fading [3]. This is mainly due to the vital role that agility and flexibility play in today's volatile and competitive business environment [28]. Especially for SCND, establishing a new facility, shutting down an existing one, or any capacity changes require capital investment on the one hand, while demand and distribution costs constantly change on the other hand. Therefore, flexibility to adjust to the changes in the environment, mainly the agility to satisfy short-term demand, has become an important aspect of modern SCND [18]. Flexibility in SCND is possible today thanks to technological advances and the use of shared warehouses, which have revolutionized supply chain network design and management [8]. Flexible or not, SCND deals with several decisions with mutual impacts. Therefore, it is advantageous to optimize them jointly. For example, synchronizing production and distribution has provided significant cost benefits for companies [22, 25, 30]. These advantages have accounted for the popularity of integrated supply chain optimization approaches. Several integrated supply chain optimization problems are already studied in the literature, where different models and solution algorithms are proposed. However, despite the proven efficiency in cost reduction of flexible and integrated supply chain planning, two concerns remain: (i) the challenges of incorporating real-world complexities [20], and (ii) the difficulty to solve large-size instances of these integrated multi-echelon problems [8].

This paper considers several important decisions faced by manufacturing companies: location, production, inventory allocation, and distribution. A company needs to decide when and how much to produce in each plant, how much inventory to keep and for how long, which ones and for how long use the potential intermediate facilities, and finally how much and when to deliver to each customer. We incorporate two specific real-world features into the problem setting. The first one is the customers' delivery due date [1], and the second one is the flexible network design [29].

We propose a decomposition method that solves this rich problem iteratively in three phases. Aiming to guide the search for good solutions, in the first phase, a relaxed version of the problem is solved. A set of guiding distribution centre (DC) selections is identified to be fixed. Another relaxed version is solved to get guiding customer assignments. Finally, using this information, a restricted model is solved. In order to obtain different and

improved solutions, the relaxed version is iteratively changed through the insertion of cuts to provide new guidance for variable fixing in the original model.

The main contributions of this paper are as follows. First, we incorporate several realistic features into our SCND model. We introduce a rich integrated problem with multiple products and multiple periods. Besides planning production, inventory, and distribution, we also optimize the timing for deliveries to final customers and leasing of the intermediate facilities. Second, we develop an efficient solution algorithm capable of solving large instances of the problem. Finally, we derive several important managerial insights for SCND based on the obtained results.

This paper is organized as follows. In Section 2, we provide an overview on the relevant literature. Section 3 presents the problem description and its mathematical formulation. We describe the proposed decomposition method in Section 4. The results of our computational experiments are presented in Section 5, followed by conclusions in Section 6.

2 Literature review

SCND and facility location have been broadly studied in the literature. More specifically, deterministic SCND problems have been the subject of interest since the multi-commodity network design problem introduced by Geoffrion and Graves [13]. During the previous decade, more realistic features have been considered in problem settings. Hence, the importance of including real-world issues into SCND and facility location problems are also emphasized in the literature [4]. Moreover, the dynamic facility location problem has been identified as an area of scarce research [28, 19]. The time dimension is what differentiates the static and dynamic versions of the problem. In the static version, the facility decisions are made at the beginning of the planning horizon and remain unchanged. In dynamic problems, on the contrary, in response to the changes in the decision environment (demand pattern, for example), one can revise the facility location decisions periodically [15, 27].

However, the idea of dynamic facility location planning is not new. To the best of our knowledge, Ballou [2] was the first to indicate that as the demand pattern changes over time, warehouse locations should be considered as a dynamic decision. A thorough review of dynamic facility location literature can be found in Owen and Daskin [24] and Farahani et al. [12]. The literature on dynamic facility location problems is scarce mainly because even the basic models of facility location are NP-hard. As discussed in Klose and Drexl [17], dynamic models are significantly more difficult than the static ones. Therefore, the literature usually treats location decisions as static and long-term or strategic in nature [6]. The problem we study in this paper lies in the category of deterministic dynamic facility location planning and therefore in what follows, we highlight some of the most recent and relevant contributions in this field.

Jena et al. [15] consider a multi-period and multi-product dynamic facility location problem. The problem is dynamic because it allows relocation of the facilities and temporary partial facility opening/closing. They propose a hybrid heuristic that solves a

Lagrangian relaxation followed by a restricted mixed-integer programming model. Their experiments show the importance of using a heuristic for large size instances. Silva et al. [27] apply three linear relaxation-based heuristics and an evolutionary heuristic and improve these solutions. An extension to cover a multi-commodity setting is studied in Jena et al. [14]. In order to solve this multi-commodity dynamic facility location problem, they propose Lagrangian heuristics based on the classical subgradient method and an aggregated bundle method. Zhi and Keskin [31] aim to find an efficient design for a production-distribution network with multiple products and stages. Their model considers direct shipment and the possibility of transshipment. They propose two metaheuristic approaches to solve the NP-hard production/distribution system design problem. All these studies share the underlying assumption that the facility location (reallocation) is a strategic level decision. This is mainly due to the considerable capital investment required for relocation [32], or closing a facility and opening a new one. However, as the decision environment changes, one would expect a more flexible approach to the facility location and supply network design problems. In what follows, we review some studies that consider a more flexible supply chain design.

Fahimnia et al. [11] introduce the facility location mobility concept in the context of blood supply chain disasters. They compare a static network design with a dynamic facility location. The authors show the reduced transportation cost as the main advantage of using a more flexible design. The idea of relocating intermediate facilities is also considered in Darvish and Coelho [7] and Darvish et al. [8]. They change the nature of location decisions to a tactical or even an operational level decision. Being inspired by the success of several on-demand warehouse providers, these authors introduce a flexible network design. Finally, Raghavan et al. [26] introduce the capacitated mobile facility location problem as an extension of its uncapacitated version proposed initially in Demaine et al. [9]. In this problem, the goal is to minimize facility movement costs and customer transportation costs. A set of existing facilities is defined, and the customers are dynamically assigned to these facilities.

3 Problem description and mathematical formulation

We now formally describe our supply chain network design problem in which we integrate decisions of production, dynamic facility location, inventory management, and distribution with due dates.

We consider a set of plants available over a finite time horizon (typically in days) producing multiple products. Most plants are assumed to be multi-functional but some products cannot be produced in certain plants. All plants have restrictions with respect to both production and storage capacities. Moreover, for each plant, fixed production setup and variable production costs are considered per product. Each plant owns a warehouse where the products are stored, and an inventory holding cost for each unit stored per period is incurred. All products are sent to DCs, where they are stored and shipped to final customers. A set of potential DCs is available for rent and the fixed rental fee is

paid per renting period (typically in months). DCs also charge an inventory holding cost per unit stored per period. Customers are offered a delivery due date that cannot be exceeded, but the producer can choose to deliver on any day between the demand release day and its due date. A direct delivery occurs, and the transportation cost is proportional to the distance traveled. Therefore, the producer can aggregate orders received from the same customer, in order to make fewer deliveries.

Formally, the problem is defined on a graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ where $\mathcal{N} = \{1, \ldots, n\}$ is the node set and $\mathcal{A} = \{(i, j) : i, j \in \mathcal{N}, i \neq j\}$ is the arc set. The node set $\mathcal{N}$ is partitioned into a plant set $\mathcal{N}_p$, a DC set $\mathcal{N}_d$, and a customer set $\mathcal{N}_c$, such that $\mathcal{N} = \mathcal{N}_p \cup \mathcal{N}_d \cup \mathcal{N}_c$. Let $\mathcal{P}$ be the set of P products, and $\mathcal{T}$ be the set of periods of the planning horizon of length H.

The inventory holding cost of product p at node $j \in \mathcal{N}_p \cup \mathcal{N}_d$ is denoted by h_{pj}, the unit shipping cost of product p from plant i to DC d by c_{pid}, and the unit shipping cost of product p from DC d to customer k by c'_{pdk}. Let also f_d be the fixed renting cost for DC d. Once rented, the DC remains open for the next g periods. We consider s_{pi} as the fixed setup cost per period for product p at plant i, v_{pi} the variable production cost of product p at plant i, and d^t_{pk} the demand of customer k for product p in period t. The demand occurring in period t can be fulfilled up to period $t + r$, as $t + r$ represents the delivery due date. For ease of representation, let D be a big number equal to the total demand for all products from all customers in all periods.

The objective is to minimize the total cost, including production, location, inventory, and distribution costs, considering that all orders have to be fulfilled until the promised delivery due date. Therefore, one needs to determine: the product(s) and quantities to be produced at each plant in each period; the DCs to be selected in each period, the inventory level of products at both plants and DCs; products sent from plants to DCs in each period; the period in which demand of each customer is satisfied, and products sent from DCs to customers in each period. Figure 1 shows a schematic of the problem.

We formulate the problem with the following binary variables: z^t_{pi} is equal to one if product p is produced at plant i in period t, and zero otherwise; y^t_d is equal to one if DC d is chosen to be rented in period t for the next g consecutive periods; w^t_d is equal to one to indicate whether DC d in period t is in its leasing period; x^t_{id} and $\hat{x}^t_{dk}$ indicate if there is any shipment from plant i to DC d in period t and from DC d to customer k in period t, respectively.

Quantities produced and shipped are defined as follows: $\hat{q}^{tt'}_{pdk}$ is the quantity of product p delivered from DC d to customer k in period t' to satisfy the demand of period t, with $t' \geq t$. a^t_{pi} is the quantity of product p produced at plant i in period t and q^t_{pid} is the quantity of product p delivered from plant i to DC d in period t. I^t_{pd} and $\hat{I}^t_{pi}$ indicate the quantity of product p held in inventory at the end of period t, at DC d and plant i, respectively. We assume that no inventory is available at the facilities at the beginning of the planing horizon.

In order to simplify the notation used in the formulation of the model, we introduce the following sets. First, let $\mathcal{R}^+_t$ denote the set of periods in which demand of period t can be satisfied, i.e., $\mathcal{R}^+_t = \{t' \in \mathcal{T} \mid t \leq t' \leq t + r\}$, and let $\mathcal{R}^-_t$ denote the set of periods

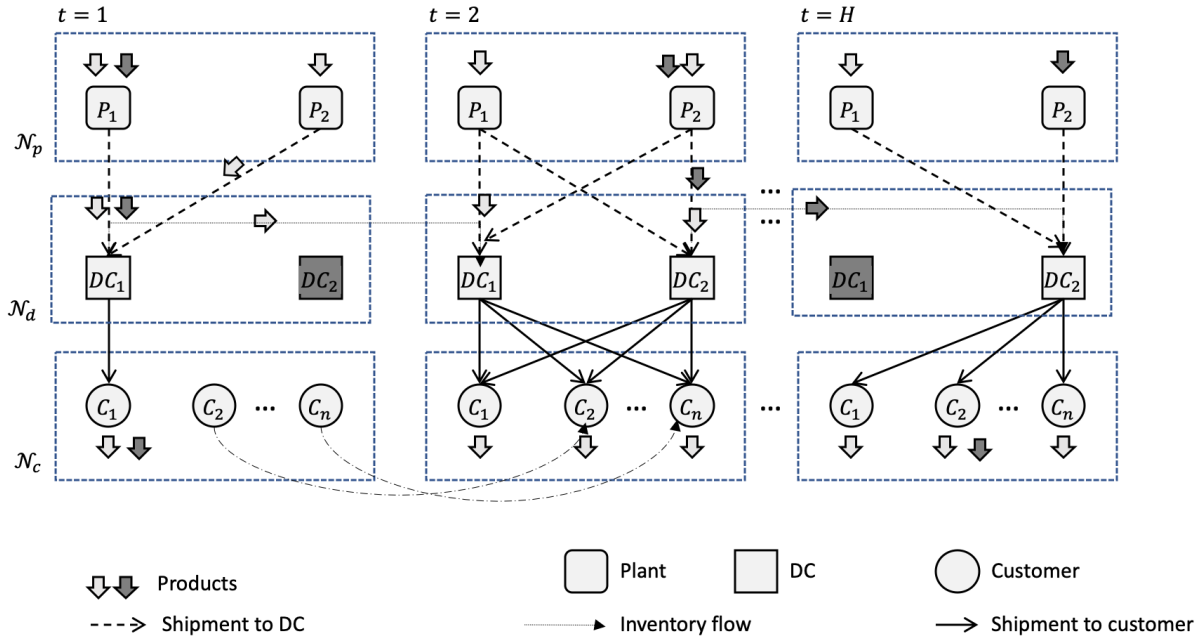

Figure 1: An example of the two-echelon supply chain network.

whose demand can be still be satisfied in period t, i.e., $\mathcal{R}_t^- = \{t' \in \mathcal{T} \mid t - r \leq t' \leq t\}$. Next, let $\mathcal{G}_t^+$ denote the set of periods covered by a lease started in period t, i.e., $\mathcal{G}_t^+ = \{t' \in \mathcal{T} \mid t \leq t' \leq t + g - 1\}$, and let $\mathcal{G}_t^-$ denote the set of periods in which a lease can start and then cover period t, i.e., $\mathcal{G}_t^- = \{t' \in \mathcal{T} \mid t - g + 1 \leq t' \leq t\}$.

Table 1 summarizes the notation used in the problem.

The problem is then formulated as follows:

$$\begin{aligned}\text{(M) } \min & \sum_{p\in\mathcal{P}}\sum_{i\in\mathcal{N}_p}\sum_{t\in\mathcal{T}} v_{pi}a_{pi}^t + \sum_{p\in\mathcal{P}}\sum_{i\in\mathcal{N}_p}\sum_{t\in\mathcal{T}} s_{pi}z_{pi}^t + \sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}_d}\sum_{t\in\mathcal{T}} h_{pd}I_{pd}^t + \\ & \sum_{p\in\mathcal{P}}\sum_{i\in\mathcal{N}_p}\sum_{t\in\mathcal{T}} h_{pi}\hat{I}_{pi}^t + \sum_{d\in\mathcal{N}_d}\sum_{t\in\mathcal{T}} f_d y_d^t + \sum_{i\in\mathcal{N}_p}\sum_{d\in\mathcal{N}_d}\sum_{t\in\mathcal{T}} c_{id}x_{id}^t + \sum_{d\in\mathcal{N}_d}\sum_{k\in\mathcal{N}_c}\sum_{t\in\mathcal{T}} c'_{dk}\hat{x}_{dk}^t\end{aligned} \tag{1}$$

subject to:

$$\sum_{p\in\mathcal{P}} a_{pi}^t \leq C_i \qquad i \in \mathcal{N}_p, t \in \mathcal{T} \tag{2}$$

$$a_{pi}^t \leq z_{pi}^t C_i \qquad i \in \mathcal{N}_p, t \in \mathcal{T}, p \in \mathcal{P} \tag{3}$$

$$\hat{I}_{pi}^t + \sum_{d\in\mathcal{N}_d} q_{pid}^t = \hat{I}_{pi}^{t-1} + a_{pi}^t \qquad p \in \mathcal{P}, i \in \mathcal{N}_p, t \in \mathcal{T} \tag{4}$$

$$\sum_{p\in\mathcal{P}} \hat{I}_{pi}^t \leq Q_i \qquad i \in \mathcal{N}_p, t \in \mathcal{T} \setminus \{H\} \tag{5}$$

$$I_{pd}^t + \sum_{k\in\mathcal{N}_c}\sum_{t'\in\mathcal{R}_t^-} \hat{q}_{pdk}^{t't} = I_{pd}^{t-1} + \sum_{i\in\mathcal{N}_p} q_{pid}^t \qquad p\in\mathcal{P}, d\in\mathcal{N}_d, t\in\mathcal{T} \tag{6}$$

$$\sum_{p\in\mathcal{P}} I_{pd}^t \le w_d^t Q_d \qquad d\in\mathcal{N}_d, t\in\mathcal{T} \tag{7}$$

$$\sum_{p\in\mathcal{P}} I_{pd}^t \le w_d^{t+1} Q_d \qquad d\in\mathcal{N}_d, t\in\mathcal{T}\setminus\{H\} \tag{8}$$

$$\sum_{t'\in\mathcal{G}_t^-} y_d^{t'} \ge w_d^t \qquad d\in\mathcal{N}_d, t\in\mathcal{T} \tag{9}$$

$$\sum_{t'\in\mathcal{G}_t^+} w_d^{t'} \ge \min(g, H-t) y_d^t \qquad d\in\mathcal{N}_d, t\in\mathcal{T} \tag{10}$$

$$\sum_{d\in\mathcal{N}_d}\sum_{t'\in\mathcal{R}_t^+} \hat{q}_{pdk}^{tt'} = d_{pk}^t \qquad p\in\mathcal{P}, k\in\mathcal{N}_c, t\in\mathcal{T} \tag{11}$$

$$\sum_{p\in\mathcal{P}}\sum_{k\in\mathcal{N}_c}\sum_{t'\in\mathcal{R}_t^-} \hat{q}_{pdk}^{t't} \le Q_d w_d^t \qquad d\in\mathcal{N}_d, t\in\mathcal{T} \tag{12}$$

$$\sum_{d\in\mathcal{N}_d}\sum_{t'\in\mathcal{R}_t^+} \hat{q}_{pdk}^{tt'} \le \sum_{d\in\mathcal{N}_d}\sum_{t'\in\mathcal{R}_t^+} d_{pk}^t w_d^{t'} \qquad p\in\mathcal{P}, k\in\mathcal{N}_c, t\in\mathcal{T} \tag{13}$$

$$\sum_{d\in\mathcal{N}_d}\sum_{t\in\mathcal{T}} Q_d w_d^t \ge \sum_{p\in\mathcal{P}}\sum_{k\in\mathcal{N}_c}\sum_{t\in\mathcal{T}} d_{pk}^t \tag{14}$$

$$\sum_{i\in\mathcal{N}_p}\sum_{t\in\mathcal{T}} C_i z_{pi}^t \ge \sum_{k\in\mathcal{N}_c}\sum_{t\in\mathcal{T}} d_{pk}^t \qquad p\in\mathcal{P} \tag{15}$$

$$\sum_{p\in\mathcal{P}}\sum_{t'\in\mathcal{R}_t^-} \hat{q}_{pdk}^{t't} \le d_{pk}^{t'} \hat{x}_{dk}^t \qquad d\in\mathcal{N}_d, k\in\mathcal{N}_c, p\in\mathcal{P}, t\in\mathcal{T}, t'\in\mathcal{R}_t^- \tag{16}$$

$$\sum_{p\in\mathcal{P}} q_{pid}^t \le D x_{id}^t \qquad i\in\mathcal{N}_p, d\in\mathcal{N}_d, t\in\mathcal{T} \tag{17}$$

$$z_{pi}^t, w_d^t, y_d^t, x_{id}^t, \hat{x}_{dk}^t \in \{0,1\} \qquad p\in\mathcal{P}, i\in\mathcal{N}_p, d\in\mathcal{N}_d, k\in\mathcal{N}_c, t\in\mathcal{T} \tag{18}$$

$$\hat{I}_{pi}^t, a_{pi}^t, I_{pd}^t, q_{pid}^t, \hat{q}_{pdk}^{tt'} \ge 0 \qquad p\in\mathcal{P}, i\in\mathcal{N}_p, d\in\mathcal{N}_d, k\in\mathcal{N}_c, t, t'\in\mathcal{T}. \tag{19}$$

The objective function (1) minimizes the total costs of production (variable and setup), inventory costs (at the DCs and at the plants), rental costs of the DCs, and distribution costs (both from plants to DCs and from DCs customers). Constraints (2) guarantee that the production capacity at the plants is respected. Constraints (3) are production setup constraints. Constraints (4) balance the inventory flow at the plants. Constraints (5) guarantee that the inventory level at each plant does not exceed its storage capacity. Constraints (6) are the inventory flow balance at the DCs. Constraints (7) and (8) ensure that the inventory kept at the selected DCs does not exceed the capacities. Constraints (9) and (10) link leasing periods with the paying period of the lease: the fixed rental fee is paid once and then the DC remains open for g consecutive periods. Constraints (11) make sure that every demand is satisfied within r days. Constraints (12) and (13) condition the delivery from a DC to customers, to the DC being selected. With constraints (14) and (15), we make sure that enough DC inventory and

Table 1: Notation used in the model

Parameters			
h_{pj}	inventory holding cost of product p at node $j \in \mathcal{N}_p \cup \mathcal{N}_d$		
c_{id}	travel cost for each pair (i,d) where $i \in \mathcal{N}_p$ and $d \in \mathcal{N}_d$		
c'_{dk}	travel cost for each pair (d,k) where $d \in \mathcal{N}_d$ and $k \in \mathcal{N}_c$		
f_d	fixed renting cost for DC d		
s_{pi}	fixed setup cost for product p in plant i		
v_{pi}	variable production cost of product p in plant i		
d^t_{pk}	demand of customer k for product p in period t		
C_i	production capacity at plant i		
Q_i	inventory capacity at node i		
Sets			
$\mathcal{N}_c$	customers	$\mathcal{R}^+_t$	periods in which demand of period t can be filled
$\mathcal{N}_p$	plants	$\mathcal{R}^-_t$	periods whose demand can be filled in period t
$\mathcal{N}_d$	DCs	$\mathcal{G}^+_t$	periods covered by a lease starting in period t
$\mathcal{T}$	periods	$\mathcal{G}^-_t$	periods in which a lease can start and cover period t
$\mathcal{P}$	products		
Variables			
z^t_{pi}	equals to one if product p is produced at plant i in period t		
y^t_d	equals to one if DC d is chosen in period t to be used for g consecutive periods		
w^t_d	equals to one to indicate whether DC d is in lease in period t		
$\hat{q}^{tt'}_{pdk}$	quantity of product p delivered from DC d to customer k in period t' to satisfy the demand of period t, where $t \leq t' \leq max(t+r, T)$		
a^t_{pi}	quantity of product p produced at plant i in period t		
q^t_{pid}	quantity of product p delivered from plant i to DC d in period t		
I^t_{pd}	quantity of product p held in inventory at DC d at the end of period t		
$\hat{I}^t_{pj}$	quantity of product p held in inventory at plant j at the end of period t		
x^t_{id}	if there is any shipment from plant i to DC d in period t		
$\hat{x}^t_{dk}$	if there is any shipment from DC d to customer k in period t		

production capacities are considered to satisfy the total demand occurring in the whole planning horizon. Constraints (16) and (17) link the delivery quantities from plant/DC to DC/customers, to the direct shipment binary variables. Finally, constraints (18)–(19) define the domain and nature of the variables.

We now explain how specific features of our mathematical model make the problem inherently difficult to solve to optimality. Generally, the problem is complex as it is a combination of several NP-hard problems, including capacitated lot sizing and capacitated facility location problems. The proposed dynamic facility location model for a two-echelon supply chain network integrates several decisions, including production, inventory, location, and distribution. Simultaneously considering all these decisions in a single model makes the problem difficult to solve.

For each order placed in period $t' - r$, we have up to period t' to fulfill it. This will add to the size and consequently, the complexity of the problem we are solving. In order to reduce the size of the problem, note that we only define variables $\hat{q}^{tt'}_{pdj}$ for feasible combinations of t and t'. The problem still remains more difficult than its same-day delivery counterpart, as will be discussed in Section 5.5.

4 Solution algorithm

In this section, we introduce our proposed decomposition algorithm. The model presented in Section 3, hereafter referred to as model M, can be solved by state-of-the-art general purpose MIP solvers only for small instances. However, to solve large instances, a more powerful solution approach is required.

Most of the required numerical effort is related to the number of binary variables in the model. In our algorithm, we focus on the variables w^t_d and $\hat{x}^t_{dk}$. Variables w^t_d specify which DCs are open in which periods, and therefore constitute the main network design variables. These can be seen as important variables, since they define the structure of the network. The variables $\hat{x}^t_{dk}$ assign customers to DCs and are the most numerous, as they contain the index k that runs over all customers, which tend to be a large set in these kinds of models. Taking control of these variables thus allows us to effectively manage the complexity inherent in the model.

Our algorithm consists of three components, namely two relaxation-based models and one restricted model. In the restricted model, we fix (most) DC locations and customer assignments (i.e., variables w^t_d and $\hat{x}^t_{dk}$) based on the current best (i.e., incumbent) solution, and solve the restricted model to optimality. This can, therefore, be seen as a neighborhood search around the current best solution, and it returns a solution at least as good, and in most cases better than the incumbent. In order to decide which DCs and customer assignments to fix, two relaxation-based models are solved. In the first one, the only binary variables are the DC locations (w^t_d), and in the second one, the only binary variables are the customer assignments ($\hat{x}^t_{dk}$). To the first, a cut is added to force the solution in a direction away from the current best, which promotes diversity. In the second relaxation-based model, the DC locations from the first one are fixed. The DC locations and customer assignments obtained by these two models are then designated as a "guiding solution," and the restricted model continues to search "in between" the current best solution and the guiding solution. In what follows the models and their interactions are more formally outlined.

We define a *solution* as a mapping $\mathcal{X}$ such that $\mathcal{X}(w^t_d), \mathcal{X}(\hat{x}^t_{dk}) \in \{0, 1\}$ specify value assignments to the binary variables w^t_d and $\hat{x}^t_{dk}$, respectively. Moreover, let $|\mathcal{X}| = \sum_{d \in \mathcal{N}_d, t \in \mathcal{T}} \mathcal{X}(w^t_d)$

denote the number of w_d^t variables assigned to 1 in $\mathcal{X}$.

We first discuss the restricted model, $\bar{M}$, which takes as input two solutions $\mathcal{X}$ and $\mathcal{Y}$, of which only $\mathcal{X}$ is required to be feasible in M. As we discuss below, at each iteration of the algorithm, $\mathcal{X}$ represents the current incumbent solution and $\mathcal{Y}$ represents a "guiding" solution. The restricted model is then given by (1)–(19) together with the constraints below, and the solution produced by the model is denoted by $\bar{M}(\mathcal{X}, \mathcal{Y})$.

$$w_d^t = \mathcal{X}(w_d^t) \qquad d \in \mathcal{N}_d, t \in \mathcal{T}, \mathcal{X}(w_d^t) = \mathcal{Y}(w_d^t) \tag{20}$$

$$\hat{x}_{dk}^t = \mathcal{X}(\hat{x}_{dk}^t) \qquad d \in \mathcal{N}_d, k \in \mathcal{N}_c, t \in \mathcal{T}, \mathcal{X}(\hat{x}_{dk}^t) = \mathcal{Y}(\hat{x}_{dk}^t). \tag{21}$$

In other words, the restricted model fixes the variables w_d^t and $\hat{x}_{dk}^t$ that are assigned to the same value in both $\mathcal{X}$ and $\mathcal{Y}$. Solving $\bar{M}$ to find $\bar{M}(\mathcal{X}, \mathcal{Y})$ can therefore be seen as a neighborhood search around the incumbent solution $\mathcal{X}$, in the direction of $\mathcal{Y}$. More formally, the search actually occurs on the "line segment" between $\mathcal{X}$ and $\mathcal{Y}$ in the Hamming space, which is also true for most crossover operators in evolutionary algorithms; see Moraglio et al. [21] for details. Note that since $\mathcal{X}$ is in the neighborhood and by definition constitutes a feasible solution to M, $\bar{M}(\mathcal{X}, \mathcal{Y})$ is a feasible solution at least as good as $\mathcal{X}$.

The second model is denoted by $\underline{M}^w$, and takes as input a solution $\mathcal{X}$ and a real number $0 \leq \gamma \leq 1$. This model is given by (1)–(17) together with the constraints below, and the solution produced by the model is denoted by $\underline{M}^w(\mathcal{X}, \gamma)$.

$$\sum_{d \in \mathcal{N}_d, t \in \mathcal{T}: \mathcal{X}(w_d^t)=1} w_d^t \leq \lfloor \gamma |\mathcal{X}| \rfloor - 1 \tag{22}$$

$$\sum_{d \in \mathcal{N}_d, t \in \mathcal{T}: \mathcal{Y}(w_d^t)=1} w_d^t \leq \lfloor \gamma |\mathcal{Y}| \rfloor - 1 \tag{23}$$

$$w_d^t \in \{0, 1\} \qquad d \in \mathcal{N}_d, t \in \mathcal{T} \tag{24}$$

$$z_{pi}^t, y_d^t, x_{id}^t, \hat{x}_{dk}^t, \hat{I}_{pi}^t, a_{pi}^t, I_{pd}^t, q_{pid}^t, \hat{q}_{pdk}^{tt'} \geq 0 \qquad p \in \mathcal{P}, i \in \mathcal{N}_p, d \in \mathcal{N}_d, k \in \mathcal{N}_c, t, t' \in \mathcal{T}. \tag{25}$$

In other words, $\underline{M}^w$ relaxes all binary variables except for w_d^t, and among all variables that are assigned a value of 1 in the solution $\mathcal{X}$, (22) ensures that fewer than $\gamma \times 100\%$ are assigned a value of 1 by the model. The same is imposed for the previous guiding solution $\mathcal{Y}$ by (23). Note that the RHS is subtracted by 1 so that $\gamma = 1$ implies a cut ensuring that the same solution is not produced a second time. This model can therefore be seen as a constraint-directed search, where (22) directs the search away from a given solution (a diversification approach).

The third model is denoted by $\underline{M}^{\hat{x}}$, which takes as input a solution $\mathcal{X}$ given by (1)–(17) together with the constraints below. The solution produced by the model is denoted by $\underline{M}^{\hat{x}}(\mathcal{X})$.

$$w_d^t = \mathcal{X}(w_d^t) \quad d \in \mathcal{N}_d, t \in \mathcal{T} \tag{26}$$

$$\hat{x}_{dk}^t \in \{0, 1\} \quad d \in \mathcal{N}_d, k \in \mathcal{N}_c, t \in \mathcal{T} \tag{27}$$

$$z_{pi}^t, w_d^t, y_d^t, x_{id}^t, \hat{I}_{pi}^t, a_{pi}^t, I_{pd}^t, q_{pid}^t, \hat{q}_{pdk}^{tt'} \geq 0 \qquad p \in \mathcal{P}, i \in \mathcal{N}_p, d \in \mathcal{N}_d, k \in \mathcal{N}_c, t, t' \in \mathcal{T}. \tag{28}$$

In other words, this model relaxes all binary variables except for $\hat{x}_{dk}^t$ and fixes the variables w_d^t according to the solution $\mathcal{X}$.

Models $\underline{M}^w$ and $\underline{M}^{\hat{x}}$ are used in succession to produce a guiding solution $\mathcal{Y}$. Given the incumbent solution $\mathcal{X}$, this is done by first solving $\underline{M}^w$ in order to obtain a solution $\mathcal{Y}' =$

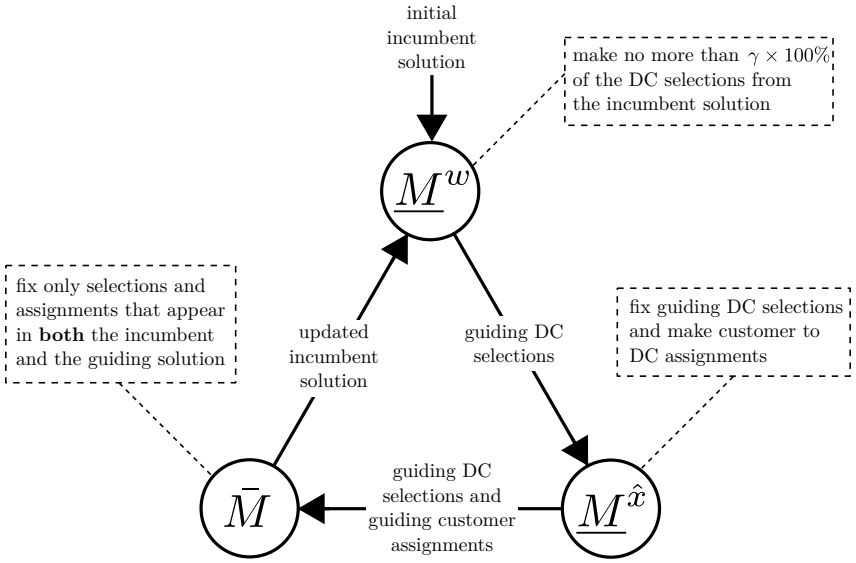

Figure 2: Graphical illustration of the proposed algorithm.

$\underline{M}^w(\mathcal{X}, \gamma)$ with different DC openings. Thereafter $\underline{M}^{\hat{x}}$ is solved to obtain the guiding solution $\mathcal{Y} = \underline{M}^{\hat{x}}(\mathcal{Y}')$ with customer assignments for the selected DCs. In other words, $\underline{M}^w$ changes the current solution and determines the DC locations to be fixed in each period, and given these fixed locations, $\underline{M}^{\hat{x}}$ assigns customers to DCs. The DCs and customer assignments that occur in both the incumbent $\mathcal{X}$ and the guiding solution $\mathcal{Y}$ are then fixed in the model $\bar{M}$, and by solving $\bar{M}$ a new incumbent is obtained in the direction of $\mathcal{Y}$ that is at least as good as $\mathcal{X}$. This process is illustrated in Figure 2.

If $\bar{M}(\mathcal{X}, \mathcal{Y}) = \mathcal{X}$ in any iteration (i.e., the incumbent did not improve), then the algorithm would be stuck in a local optimum. In this case, the parameter γ is decreased by 0.1 to force $\underline{M}^w$ to find a different guiding solution further away from $\mathcal{X}$. The parameter is reset after the incumbent is improved.

In order to generate an initial feasible solution, we first solve $\underline{M}^w$ with input $\mathcal{X}_0$, which is a solution that assigns only zeros to variables. Constraint (22) thus falls away from $\underline{M}^w$. Let $\mathcal{Y}' = \underline{M}^w(\mathcal{X}_0, \gamma)$ and then let $\mathcal{Y} = \underline{M}^{\hat{x}}(\mathcal{Y}')$. To attempt to find a feasible solution, $\bar{M}$ is solved to obtain $\bar{M}(\mathcal{Y}, \mathcal{Y})$, which is equivalent to fixing the partial solution $\mathcal{Y}$ in M. There is a risk that the resulting model is infeasible. In such a case, one option is to add a cut to $\underline{M}^w$ in order to obtain a different solution $\mathcal{Y}$ to fix. However, we did not explore this further as it never occurred using our instances.

5 Computational experiments

The formulation in Section 3 and the proposed algorithm in Section 4 are coded in C++ and solved using CPLEX 12.10 with default parameter settings apart from the time limit and number of threads. The computational experiments are conducted on the Central DTU HPC Cluster[1], using four threads per execution to match the performance of a standard desktop computer.

[1] `https://www.hpc.dtu.dk/?page_id=2520`

The performance of the proposed algorithm is demonstrated on a set of randomly generated instances. In order to compare methods, a time limit of one hour was imposed. Time limits were also imposed on the subproblems in each iteration of the proposed algorithm in order to be able to have some control over the number of iterations. We do this by first specifying a desired number of iterations NI. Note that this number will not necessarily correspond to the actual number of iterations, as discussed further below. The time limit imposed to each of the three subproblems in each iteration is then $3600/(3 \times NI)$ seconds. In particular, we will consider $NI = 5$ and $NI = 10$ resulting in subproblem time limits of 240 and 120 seconds, respectively. If each subproblem uses exactly its allocated time in each iteration, NI iterations will be performed. However, this might not always be the case. First of all, if a subproblem is solved to optimality before its time limit, then it will of course use less time. Such events might lead to more than NI iterations being performed. On the other hand, note that the proposed algorithm requires that each subproblem returns a feasible solution in each iteration, so if a subproblem does not find any feasible solution before its time limit is reached, it is allowed to continue and is stopped as soon as it finds the first feasible solution. In such cases, more time will be used, which might lead to fewer than NI iterations being performed. In any case, the proposed algorithm runs until the time limit of one hour is reached. Moreover, for none of our instances was CPLEX able to solve the full MIP to optimality within one hour. We thus do not directly report any runtimes.

In some cases it occurred that either $\underline{M}^w$ or $\underline{M}^{\hat{x}}$ became infeasible. This happened in 37% of our instances, and in most cases it only happened later on during the search (on average after 2607 seconds). In such cases, we use the incumbent solution to warm start the full MIP and let it run for the remaining time.

In the remainder of this section, the details of instance generation are provided first and then the results obtained by the proposed algorithm are compared with the ones from the standard MIP solver (CPLEX 12.10). Finally, we present an analysis of the use of flexibility in the solutions found, and provide a number of managerial insights drawn from this analysis.

5.1 Generation of the instances

We perform our experiments on several randomly generated instances. A total of 243 instances are created as shown in Table 2. Our planning time unit is in days and we consider 30, 60, or 90 days in each planning horizon. The location of plants, DCs, and customers are generated using random coordinates. The coordinates of customers and DCs are randomly selected from $[300, 500]$, while the plants are distributed within $[0, 100]$. The idea is to locate plants farther away from DCs and customers, mainly to imitate the recent initiatives to keep production facilities far from city centers, where customers are located. In all our instances, we consider the leasing period (active period) for DCs (g) to be one month, meaning that from the day a DC is selected, it is available for 30 days. The delivery lead time or the maximum allowed time between the release and due dates (r) is set to five days. This means that within five days from the ordering day, products must be delivered to the customer. Finally, to mimic currently specialized production systems, plants are multi-functional but not able to produce all products. Therefore, for each product, we forbid its production in one randomly selected plant.

In the following sections, we denote an instance by $\mathcal{I} = (P, T, N_c, N_d, N_p)$. Note that three values are considered for each of these five parameters, and one instance per parameter setting leads to the $3^5 = 243$ combinations. As a benchmark for evaluating the performance of our

Table 2: Input parameter values

Name	Parameter	Values	Name	Parameter	Values
Products	P	$\{5, 10, 15\}$	Demand	d^t_{pk}	[5,15]
Periods	T	$\{30, 60, 90\}$	Plant setup cost	s_{pi}	[30,90]
Plants	N_p	$\{3, 4, 5\}$	Plant variable cost	v_{pi}	[5,15]
Plant coordinates	X_i, Y_i	[0,100]	Customer and DC coordinates	X_j, Y_j	[300,500]
DCs	N_d	$\{1, 2, 3\}$	DC active period	g	30 days
Customers	N_c	$\{50, 80, 100\}$	Delivery lead time	r	5
Shipping cost coefficient (plants-DC)	c_{pid}	(0, 1)	Shipping cost (DC-customers)	c'_{pdk}	[2,10]
Inventory holding cost plant	h_{pi}	[1,5]	Inventory holding cost DC	h_{pd}	[4, 10]
Production capacity plant	C_i	$0.5 \times D_{max}$	Inventory capacity	Q_j	$1.2 \times D_{max}$
			Fixed DC renting cost	f_d	$[2000, 5000]$

where D_{max} equals to the total demand of the peak day ($\max_t \sum_{i \in \mathcal{C}} d^t_{pk}$)

proposed approach in the following sections, we solve the full MIP presented in Section 3. In particular, this provides a lower bound for each instance, denoted by $LB_{\mathcal{I}}$. This is used in the subsequent sections to calculate an optimality gap, denoted by

$$\Gamma_{\mathcal{I}}(UB) = \frac{UB - LB_{\mathcal{I}}}{UB} \times 100\%$$

for a given upper bound UB for instance $\mathcal{I}$. Solving the full MIP also provides a benchmark upper bound, denoted by $UB_{\mathcal{I}}$, and a benchmark optimality gap, given by $\Gamma_{\mathcal{I}}(UB_{\mathcal{I}})$.

5.2 Parameter tuning

Two parameters are needed as input to the proposed algorithm, namely γ (representing the percentage limit of guiding DC selections that also occur in the incumbent) and NI, the desired number of iterations. For the parameter γ, the trade off is that a large value leads to guiding solutions close to incumbents, while a small value leads to guiding solutions further away. This parameter thus controls the size of the neighborhoods to be searched. Intuitively then, larger values should be preferred, as smaller values bring one closer to simply solving the full MIP. For the parameter NI, the trade off is that smaller values allow more time for subproblems to explore their respective solution spaces, while larger values allow for a more diverse exploration of the overall search space.

In order to explore the effects of these parameters, a subset of 32 of the largest instances were chosen and the parameter values $NI \in \{5, 10\}$ and $\gamma \in \{0.5, 0.75, 1.0\}$ were tested. For each instance $\mathcal{I}$ and parameter setting (NI, γ), an upper bound (best feasible solution) $UB^{NI,\gamma}_{\mathcal{I}}$ is produced, plus an optimality gap $\Gamma_{\mathcal{I}}\left(UB^{NI,\gamma}_{\mathcal{I}}\right)$. These gaps are shown in Table 3, and the smallest gap for each instance is shown in **boldface**. The final row of the table shows the number of instances for which each parameter setting produced the best feasible solution. Note that these numbers do not necessarily add up to 32 (the number of instances used), since for some instances the best solution was produced by more than one parameter setting.

As shown in the table, the parameter setting $(5, 1.0)$ is dominant, especially on larger instances. As the parameter NI is increased, poorer performance is observed, and also as the

Table 3: Optimality gaps for the different parameter settings of the proposed algorithm.

Instance	Parameter values (NI, γ)					
(P,T,N_c,N_d,N_p)	(5,1.0)	(5,0.75)	(5,0.5)	(10,1.0)	(10,0.75)	(10,0.5)
(10,60,80,2,4)	**4.20**	4.43	4.51	4.32	4.60	4.60
(10,60,80,2,5)	**4.06**	4.52	4.55	4.54	4.70	4.82
(10,60,80,3,4)	4.44	4.87	4.46	**4.41**	4.87	4.87
(10,60,80,3,5)	**6.18**	6.31	6.31	**6.18**	6.31	6.31
(10,60,100,2,4)	**2.64**	3.09	3.06	2.94	3.09	3.02
(10,60,100,2,5)	3.81	4.23	3.93	**3.58**	4.23	4.23
(10,60,100,3,4)	**4.33**	4.67	4.67	4.34	6.36	4.67
(10,60,100,3,5)	4.77	4.99	4.91	**4.65**	5.09	5.09
(10,90,80,2,4)	4.08	4.18	4.19	**4.06**	4.24	4.31
(10,90,80,2,5)	**4.56**	4.71	4.69	4.77	4.78	4.64
(10,90,80,3,4)	**4.21**	4.31	4.38	4.24	4.31	4.43
(10,90,80,3,5)	**5.69**	5.90	5.89	7.93	5.89	5.89
(10,90,100,2,4)	**3.23**	3.40	3.40	5.11	3.40	3.40
(10,90,100,2,5)	**3.09**	3.23	3.19	3.12	3.23	3.23
(10,90,100,3,4)	5.36	**5.26**	5.47	5.33	**5.26**	41.26
(10,90,100,3,5)	**6.08**	6.49	6.41	6.12	6.49	6.49
(15,60,80,2,4)	3.04	3.29	3.29	**2.87**	3.29	3.26
(15,60,80,2,5)	**3.92**	3.94	4.12	4.08	4.12	4.12
(15,60,80,3,4)	**4.40**	4.67	4.70	4.40	4.67	4.67
(15,60,80,3,5)	4.67	4.75	4.75	**4.67**	5.22	4.75
(15,60,100,2,4)	**2.36**	2.63	2.59	2.41	2.59	2.59
(15,60,100,2,5)	3.48	3.51	3.48	**3.32**	41.97	3.55
(15,60,100,3,4)	3.36	**3.33**	3.53	35.25	3.54	3.56
(15,60,100,3,5)	**2.81**	3.21	3.21	2.88	3.23	3.23
(15,90,80,2,4)	6.90	5.59	**5.39**	25.68	27.01	20.66
(15,90,80,2,5)	**4.47**	4.67	4.60	4.67	4.67	4.67
(15,90,80,3,4)	**2.90**	3.02	3.02	32.95	4.22	32.95
(15,90,80,3,5)	3.88	3.97	3.97	**3.86**	3.95	4.02
(15,90,100,2,4)	**3.08**	3.24	3.14	3.39	3.40	3.40
(15,90,100,2,5)	**3.43**	3.71	3.51	3.46	3.71	3.51
(15,90,100,3,4)	**2.56**	2.74	2.78	3.76	3.76	3.76
(15,90,100,3,5)	**4.22**	5.76	5.76	4.28	4.38	4.38
Best count	21	2	1	9	1	0

parameter γ is decreased. Since the parameter setting $(5, 1.0)$ performs the best overall, this setting is used in the next section to compare the proposed algorithm against solving the full MIP using CPLEX.

5.3 The proposed algorithm vs the MIP/CPLEX benchmark

We use the following metrics to evaluate the performance of the proposed algorithm using the parameter setting $(5, 1.0)$, henceforth referred to as $P(5, 1)$. First of all, recall that $UB_{\mathcal{I}}$ and $LB_{\mathcal{I}}$ denote the best upper and lower bound for instance $\mathcal{I}$ found by CPLEX when solving the full MIP formulation presented in Section 3. We will here further denote by $UB^*_{\mathcal{I}}$ the best upper bound found by $P(5, 1)$. Then we define the percentage improvement with respect to CPLEX as

$$\text{improve}_{\mathcal{I}} = \frac{UB_{\mathcal{I}} - UB^*_{\mathcal{I}}}{UB_{\mathcal{I}}}$$

and the reduction in the optimality gap with respect to CPLEX as

$$\text{reduce}_{\mathcal{I}} = \Gamma_{\mathcal{I}}\left(UB_{\mathcal{I}}\right) - \Gamma_{\mathcal{I}}\left(UB^{*}_{\mathcal{I}}\right).$$

Tables 4 and 5 show the results of using $P(5,1)$ and CPLEX to solve the 243 instances, which are grouped according to the number of DCs, plants and customers (N_d, N_p, N_c, respectively), and the numbers shown are thus all averages over the 9 instances in each class. In particular, Table 4 shows the absolute upper and lower bounds, while Table 5 shows the optimality gaps.

Table 4: Average upper and lower bounds produced by CPLEX and average upper bounds produced by the proposed algorithm, where the best average upper bound for each class of instances is given in **boldface**.

		$N_d = 1$			$N_d = 2$			$N_d = 3$		
		CPLEX		$P(5,1)$	CPLEX		$P(5,1)$	CPLEX		$P(5,1)$
N_p	N_c	UB	LB	UB	UB	LB	UB	UB	LB	UB
3	50	2,996,692	2,875,647	**2,981,243**	3,201,730	2,837,690	**2,952,498**	3,329,700	2,750,181	**2,883,954**
	80	4,734,640	4,454,237	**4,588,245**	4,919,571	4,389,591	**4,519,384**	5,316,741	4,413,939	**4,601,962**
	100	5,762,877	5,582,200	**5,742,055**	6,440,747	5,515,500	**5,680,130**	7,149,562	5,880,964	**6,088,986**
4	50	2,837,807	2,703,307	**2,820,168**	2,987,228	2,526,305	**2,654,020**	3,219,412	2,604,002	**2,774,303**
	80	4,414,253	4,187,685	**4,335,482**	5,052,711	4,118,645	**4,292,452**	5,180,275	4,133,123	**4,309,795**
	100	5,160,501	4,816,083	**4,944,812**	5,967,239	5,113,505	**5,281,916**	6,484,691	5,013,242	**5,227,570**
5	50	2,625,138	2,487,907	**2,603,366**	2,926,164	2,443,341	**2,599,180**	3,036,666	2,445,753	**2,623,518**
	80	4,220,030	3,948,452	**4,105,180**	4,497,164	3,703,494	**3,875,849**	5,125,534	3,836,083	**4,033,236**
	100	5,418,343	5,019,538	**5,152,513**	5,997,941	4,915,772	**5,100,697**	6,382,540	4,713,871	**4,953,109**
Average		4,241,142	4,008,340	**4,141,451**	4,665,610	3,951,538	**4,106,236**	5,025,013	3,976,795	**4,166,270**

The results from Table 4 show that our algorithm consistently provided better average solutions, as indicated by the boldface on all its solutions. Its solutions are then compared to the LB from CPLEX. From Table 5 it can be seen that the proposed algorithm $P(5,1)$ consistently shows gaps of around 4% (the mean is 4.05% and the standard deviation is 1.67%), while for CPLEX the gaps increase as the instances become larger (the mean is 12.4% and the standard deviation is 9.1%). Moreover, the proposed algorithm $P(5,1)$ provides better solutions than CPLEX in 235 out of 243 (97%) instances. In the cases where the proposed algorithm performs better, it is able to reduce the gap by 8.66% on average, the reduction being the largest for the largest class of instances, where the gap was reduced on average by 22.96%.

CPLEX outperforms $P(5,1)$ in only 8 instances. However, these 8 instances are only among the small ones, and most of them among instances with one DC only, where the instances are perhaps not yet too difficult for CPLEX to solve. At best, CPLEX reduces the gap by only 1.93% compared to $P(5,1)$, and by 0.79% on average. Compared to CPLEX, the proposed algorithm is thus competitive on smaller instances, while being capable of finding significantly better solutions on larger instances within the same time limit.

5.4 Iteration-level analysis of the proposed algorithm

In this section, the proposed algorithm is analyzed in more detail by showing results during the development of the iterations. First of all, Table 6 shows the amount of time spent by the

Table 5: Comparison of the results obtained by the proposed algorithm and CPLEX

N_d	N_p	N_c	average gap % proposed	average gap % CPLEX	% instances better than CPLEX	if better improve$_{\mathcal{I}}$	if better reduce$_{\mathcal{I}}$	if worse $-$improve$_{\mathcal{I}}$	if worse $-$reduce$_{\mathcal{I}}$
		50	3.17	3.81	100	0.65	0.63		
	3	80	2.76	4.42	89	1.99	1.93	0.55	0.53
		100	2.66	2.99	67	1.04	1.02	1.11	1.06
		50	4.00	4.72	89	0.94	0.90	0.73	0.70
1	4	80	3.05	4.43	89	1.76	1.71	1.21	1.16
		100	2.64	5.08	100	2.50	2.43		
		50	4.27	5.42	89	1.42	1.35	0.38	0.36
	5	80	3.35	4.99	89	1.97	1.90	0.43	0.40
		100	2.73	5.99	100	3.37	3.27		
		50	4.03	10.37	100	6.59	6.34		
	3	80	3.06	9.17	100	6.31	6.12		
		100	3.09	12.25	100	9.43	9.16		
		50	4.74	12.93	100	8.60	8.19		
2	4	80	3.93	16.07	100	12.69	12.14		
		100	3.39	12.71	100	9.65	9.33		
		50	5.87	13.29	100	7.92	7.42		
	5	80	4.48	14.52	100	10.54	10.05		
		100	3.58	14.14	100	10.94	10.55		
		50	4.81	14.98	100	10.75	10.17		
	3	80	4.17	16.69	100	13.10	12.52		
		100	3.74	16.83	100	13.63	13.09		
		50	6.09	17.73	100	12.37	11.64		
3	4	80	4.44	18.25	100	14.43	13.81		
		100	4.54	21.76	100	18.13	17.22		
		50	6.69	17.53	100	11.69	10.85		
	5	80	4.92	25.66	100	21.87	20.75		
		100	5.18	28.13	100	24.30	22.96		
Average			4.05	12.40	97	9.09	8.66	0.83	0.79

proposed algorithm in each of its components, namely $\underline{M}^w$, $\underline{M}^{\hat{x}}$, $\bar{M}$, and the warm-started full MIP if either $\underline{M}^w$ or $\underline{M}^{\hat{x}}$ became infeasible, denoted here by M^*.

Table 6: Time spent by the different components of the proposed algorithm.

		$N_d = 1$				$N_d = 2$				$N_d = 3$			
N_p	N_c	$\underline{M}^w$	$\underline{M}^{\hat{x}}$	$\bar{M}$	M^*	$\underline{M}^w$	$\underline{M}^{\hat{x}}$	$\bar{M}$	M^*	$\underline{M}^w$	$\underline{M}^{\hat{x}}$	$\bar{M}$	M^*
	50	467	1591	1072	471	1105	1422	918	154	1246	1331	418	604
3	80	700	1532	996	373	1340	1647	614	0	1490	1541	450	120
	100	932	1430	712	527	1166	1347	506	582	1394	1335	300	572
	50	419	1384	1509	288	1122	1435	905	138	1444	1373	502	281
4	80	811	1312	1138	339	1220	1528	823	29	1348	1441	522	289
	100	900	1412	911	376	1401	1375	511	313	1219	1344	451	587
	50	381	1416	1300	503	880	1284	839	597	1144	1359	879	218
5	80	802	1313	749	736	1136	1383	871	210	1241	1249	660	447
	100	778	1548	657	617	1410	1500	576	115	1273	1301	584	443
Average		688	1438	1005	470	1198	1436	729	237	1311	1364	529	396

As can be seen from the table, most of the time is usually spent on $\underline{M}^{\hat{x}}$, likely because $\hat{x}^t_{dk}$ constitutes the largest class of binary variables (due to a large number of customers). The division of time between $\underline{M}^w$ and $\bar{M}$ depends on the number of DCs in the instance, where for fewer DCs, $\bar{M}$ uses more time, and for $\underline{M}^w$ is the other way around (i.e., more time for instances with more DCs). This also makes sense, as the difficulty of $\underline{M}^w$ is driven by the number of w^t_d

variables, which depends on the number of DCs. Finally, on average, very little time is used by M^*. As already discussed, this component is only sometimes used by the algorithm.

Table 7 shows further results on iterations, namely the number of iterations used, the percentage of iterations where the incumbent solution was improved, the optimality gap of the initial solution and the optimality of the final solution (which corresponds to the same column in Table 5). The table shows that on average slightly more than the desired number of 5 iterations were performed, indicating that cases where subproblems were solved to optimality within the time limit of 240 seconds occurred more frequently than cases where a subproblem had difficulties in finding a feasible solution. The table also shows that, on average, the incumbent is improved in 89% of the iterations, indicating that the algorithm was seldomly stuck in local optima. Finally, the table shows that the first iteration was already able to improve the gap compared to CPLEX, but that further iterations were useful in reducing the gap even further, especially on the largest class of instances where the gap was reduced on average from 11.38% to 5.18% (while CPLEX showed an average gap of 28.13%).

Table 7: Detailed results on iterations performed.

			Iterations	% iterations	Gap (%)	
N_d	N_p	N_c	performed	w. improvement	Initial solution	Final solution
1	3	50	6.44	92	4.08	3.17
		80	6.22	93	4.06	2.76
		100	5.89	95	4.22	2.66
	4	50	5.67	98	4.89	4.00
		80	5.44	81	4.22	3.05
		100	5.78	92	4.09	2.64
	5	50	5.78	91	5.68	4.27
		80	5.33	83	4.21	3.35
		100	6.33	85	3.60	2.73
2	3	50	5.78	87	4.61	4.03
		80	6.67	92	3.74	3.06
		100	5.44	81	3.81	3.09
	4	50	5.89	93	5.52	4.74
		80	6.11	90	4.43	3.93
		100	5.67	92	4.77	3.39
	5	50	5.11	81	6.64	5.87
		80	5.44	84	5.06	4.48
		100	6.00	95	4.51	3.58
3	3	50	5.44	83	5.41	4.81
		80	6.11	98	4.78	4.17
		100	5.33	86	4.14	3.74
	4	50	5.56	84	6.62	6.09
		80	5.67	87	5.36	4.44
		100	5.44	90	5.05	4.54
	5	50	5.11	93	7.42	6.69
		80	5.11	84	5.59	4.92
		100	5.33	82	11.38	5.18
Average			5.71	89	5.11	4.05

5.5 Analysis of flexibility

In this section, we analyze the extent to which flexibility is utilized in the solutions found. Recall that there are essentially two types of flexibility in the model. Firstly, there is flexibility in the timing of the delivery of customer orders (*customer flexibility*), reflected by the parameter r,

which states that the demand of a customer for a certain product occurring in period t may be fulfilled up to r periods later. Secondly, there is flexibility in the structure of the network (*network flexibility*), since DCs need not be opened for the entire planning horizon. This is reflected in the parameter g, which states that a DC rented in period t will remain open until the end of period $t+g$, where g is typically less than the total number of periods in the planning horizon. In our instances, we assumed that $r = 5$ and $g = 30$, while the time horizon in the instances is either 30, 60 or 90 periods.

Table 8 shows the results on the utilization of the two types of flexibility in the solutions found by the proposed algorithm $P(5, 1)$, including a comparison to solutions found for the case where $r = 0$, i.e., where there is no customer flexibility. For $r = 0$, the full MIP was solved using CPLEX with a time limit of one hour. As we shall discuss, setting $r = 0$ results in a problem that is much easier to solve, and therefore we simply solve the full MIP here in order to obtain optimal solutions (to most instances).

Table 8: Results on the utilization of flexibility in the model solutions.

			Delayed delivery ($r = 5$)								No delay allowed	
			Periods of delay (%)						# of	% of periods in which	# of	% of periods in which
N_d	N_p	N_c	0	1	2	3	4	5	DCs used	a used DC is active	DCs used	a used DC is active
1	3	50	18	18	17	17	16	14	1.00	96	1.00	100
		80	18	18	17	17	16	14	1.00	95	1.00	100
		100	19	17	17	17	16	15	1.00	96	1.00	100
	4	50	18	18	17	17	16	13	1.00	96	1.00	100
		80	18	18	18	17	16	14	1.00	96	1.00	100
		100	17	18	18	17	16	14	1.00	94	1.00	100
	5	50	17	18	18	17	16	14	1.00	97	1.00	100
		80	17	17	18	17	16	14	1.00	96	1.00	100
		100	16	17	18	18	17	14	1.00	95	1.00	100
2	3	50	17	18	18	18	16	13	1.56	97	1.67	97
		80	16	18	18	18	17	14	1.89	93	1.89	100
		100	16	17	18	18	17	14	2.00	98	1.89	100
	4	50	16	18	18	18	17	13	1.56	96	1.67	100
		80	17	18	18	17	16	14	1.67	96	1.89	100
		100	16	17	18	18	17	14	2.00	99	1.89	100
	5	50	17	18	18	18	16	13	1.56	97	1.44	100
		80	17	18	18	18	16	13	1.78	97	1.67	100
		100	16	17	18	18	17	14	2.00	97	2.00	100
3	3	50	16	18	18	18	17	14	2.67	98	2.56	100
		80	16	18	18	18	16	14	2.89	99	2.89	100
		100	15	17	18	18	17	14	3.00	100	2.67	100
	4	50	16	17	18	18	17	14	3.00	99	2.22	100
		80	16	17	18	18	17	14	2.89	98	2.56	100
		100	15	17	18	18	17	14	3.00	99	2.78	100
	5	50	17	18	18	18	16	14	2.22	96	2.89	99
		80	16	18	18	18	16	14	2.67	96	2.78	100
		100	16	18	18	18	17	14	2.78	95	2.67	98
	Average		17	18	18	18	16	14	1.86	97	1.81	99.8

Firstly, the table shows for each feasible amount of delay in $\{0, \ldots, r = 5\}$ the average percentage of demand delayed by this number of periods. From this, we can see that customer flexibility is used to a large extent, in that each possible number of periods of delay is used roughly the same fraction of time. Secondly, the table shows the average number of DCs rented, and the average percentage of time the DCs are rented for. From this, we can see that network flexibility is used to a far lesser extent than customer flexibility, since any rented DC is usually used most of the time. This raises the question of how removing the possibility of customer flexibility will affect the usage of network flexibility. For this reason, the last two columns of the table show the same results where no delay is allowed (i.e., $r = 0$). Interestingly, when no customer flexibility is allowed, network flexibility is almost never used — in almost all instances

when a DC is rented, it is rented for the entire planning horizon if no customer flexibility is allowed. What this shows is that network flexibility is exploited only if customer flexibility is also allowed. A more diverse demand pattern would likely lead to different DC usage and increased exploitation of its flexibility.

Further comparison with the case of $r = 0$ is also shown in Table 9. Here, we compare the total network design cost between the cases of $r = 0$ and $r = 5$, together with the utilization of the DC capacities.

Table 9: Comparison of the cases $r = 0$ (no delay allowed) and $r = 5$ (delay allowed).

			Cost (millions)		% cost	DC utilization		No delay allowed	
N_d	N_p	N_c	No delay allowed	Delay allowed	reduction	No delay allowed	Delay allowed	Time (s)	Gap (%)
1	3	50	3.50	2.98	17	58	65	0.44	0
		80	5.35	4.59	16	61	68	0.67	0
		100	6.59	5.74	15	61	68	0.56	0
	4	50	3.28	2.82	16	64	71	0.56	0
		80	5.15	4.34	19	58	63	2.33	0
		100	5.88	4.94	18	55	61	1.11	0
	5	50	3.20	2.60	21	54	63	1.11	0
		80	4.86	4.11	18	56	62	1.22	0
		100	6.15	5.15	19	51	55	1.22	0
2	3	50	3.32	2.95	13	41	49	139.00	0
		80	5.14	4.52	14	32	36	140.78	0
		100	6.47	5.68	14	32	32	33.22	0
	4	50	3.08	2.65	16	36	49	130.56	0
		80	4.91	4.29	14	34	46	264.78	0
		100	6.02	5.28	14	32	32	183.33	0
	5	50	2.98	2.60	15	47	49	361.56	0
		80	4.48	3.88	15	37	38	140.67	0
		100	5.88	5.10	15	27	33	224.44	0
3	3	50	3.20	2.88	11	26	29	1905.33	0.25
		80	5.10	4.60	11	22	25	1969.78	0.45
		100	6.68	6.09	10	22	21	1860.44	0.23
	4	50	3.12	2.77	13	30	27	2594.22	0.51
		80	4.82	4.31	12	23	22	3259.78	0.37
		100	5.83	5.23	12	21	21	2831.89	0.48
	5	50	2.94	2.62	13	22	42	3029.33	1.17
		80	4.57	4.03	14	20	26	2924.44	0.75
		100	5.61	4.95	13	23	29	3533.11	0.54
Average			4.74	4.14	15	39	44	945.77	0.18

Note that increasing r relaxes the problem, so the solutions found obviously have lower costs. However, it is interesting to see the amount of savings possible by allowing customer flexibility. A cost-savings of around 15% on average can be seen, which can be significant if costs are high. Note that this cost savings is obtained even though for $r = 5$ the proposed algorithm does not solve the problem to optimality, and not all demands are delayed by five periods. The table also shows that the runtime needed to solve the problem to optimality for $r = 0$ is very often below one hour, and it can thus be concluded that the cost savings obtained through including customer flexibility comes at the cost of a problem that is much harder to solve. In addition to cost savings, the table also shows that utilization of the DC capacities goes up with an increase in customer flexibility. The benefits of customer flexibility therefore do not only lie in cost savings, but also in increased capacity utilization, which is often a key performance indicator used to evaluate the supply chain performance.

Finally, we analyze the extent to which economies of scale in transportation affect the utilization of flexibility. For this reason we consider an adapted version of the full MIP presented in Section 3, where the last two terms of the objective function are removed and replaced by

$$\sum_{p\in\mathcal{P}}\sum_{i\in\mathcal{N}_p}\sum_{d\in\mathcal{N}_d}\sum_{t\in\mathcal{T}} c_{id}q^t_{pid} + \sum_{p\in\mathcal{P}}\sum_{d\in\mathcal{N}_d}\sum_{k\in\mathcal{N}_c}\sum_{t\in\mathcal{T}}\sum_{t'\in\mathcal{R}^-_t} c'_{dk}\hat{q}^{t't}_{pdk},$$

and where constraints (16)–(17) are removed. The binary variables x_{id}^t and $\hat{x}_{dk}^t$ are thus completely removed from the model, and transportation costs are no longer incurred for each shipment independent of quantity, but it is incurred per unit shipped. This version of the full problem formulation is then solved using CPLEX with one hour time limit. Table 10 shows the extent to which customer flexibility is utilized (as in Table 8), together with the computation times and the fraction of instances for which optimal solutions where obtained within the time limit.

Table 10: Results for the case where there are no fixed transportation costs.

			Periods of delay (%)							
N_d	N_p	N_c	0	1	2	3	4	5	Time (s)	Gap (%)
1	3	50	33	20	15	12	11	10	1152	0
		80	46	16	12	10	9	8	892	0
		100	61	13	8	6	5	6	1271	0
	4	50	39	18	13	11	10	9	878	0
		80	43	17	12	10	9	8	776	0
		100	42	17	13	10	9	9	1485	0
	5	50	47	16	11	9	9	8	934	0
		80	45	17	12	10	8	7	947	0
		100	46	17	11	9	8	8	782	0
2	3	50	43	18	12	10	9	8	2300	0.03
		80	55	14	9	7	7	7	2295	11.05
		100	54	15	10	8	7	6	2359	0.01
	4	50	42	18	13	11	9	8	2408	0.02
		80	53	16	10	8	7	6	2301	0.01
		100	55	17	10	7	6	5	2118	11.05
	5	50	49	19	11	8	7	6	1975	0.03
		80	56	16	9	7	6	6	2193	11.05
		100	54	16	10	8	7	6	2518	11.05
3	3	50	55	16	10	8	6	5	2694	0.04
		80	62	13	8	6	6	5	2600	11.13
		100	50	16	12	8	7	7	2831	22.24
	4	50	61	15	7	6	5	5	2670	0.04
		80	53	14	10	9	8	7	2839	11.14
		100	58	15	9	7	6	5	2760	22.24
	5	50	43	19	12	10	9	7	2755	0.05
		80	59	14	9	7	6	5	2558	11.13
		100	62	13	8	6	5	5	2711	22.24
Average:			51	16	11	8	7	7	2000	5.35

As can be observed from Table 10, the extent to which customer flexibility was utilized in this case is considerably less than the case with fixed transportation costs (as shown in Table 8). Whereas in the latter case the number of periods of delay is almost equally spread between 0 and $r = 5$, in this case around half (51%) of the demand is delivered in the period it arises, as no economies of scale are present to justify delayed deliveries. This means that, when there are fixed transportation costs, customer flexibility is to a large extent used in order to exploit economies of scale in transportation. The table also shows that fixed transportation costs is another aspect of the model that contributes to its difficulty, since in many cases the optimal solution was found within one hour. However, the reduction in computation time is not as large as in the case where customer flexibility is removed (i.e., $r = 0$), indicating that customer flexibility contributes more to the difficulty of the problem.

5.6 Managerial insights

The results from the previous section show that customer flexibility is essential for exploiting both network flexibility and economies of scale in transportation. This makes also intuitive sense, since customer flexibility can be used to lump demand together in specific periods, leaving other periods free of demand. This, first of all, means that in certain periods DCs might not be needed, and costs savings and increased capacity utilization are possible through network flexibility where DCs are only rented when needed. Secondly, lumping together demand in the same periods allows for shipping larger quantities in the same period, leading to cost savings coming from economies of scale.

Network flexibility was exploited to a lesser extent compared to customer flexibility. This is likely due to the fact that demands can only be shifted in time by 5 periods, while DCs have to be rented for at least 30 periods. Thus it is not possible to "clear" an entire 30 period window of all demands so that no DC would be needed then. One could therefore expect that increasing the amount by which demands can be delayed while decreasing the lease period will lead to network flexibility being utilized to a larger extent.

The most important conclusion to take away from the above discussions is therefore that the two types of flexibility both depend on and reinforce each other. It is then important for flexible network design to work towards exploiting both simultaneously, and ensure that the windows of flexibility, which in this case was $r = 5$ and $g = 30$, to match more closely.

Finally, we have shown that customer flexibility in our model comes at the cost of an increase in the difficulty of solving the problem. With no customer flexibility, most of the instances we consider can be solved to optimality using an off-the-shelf solver in under an hour. On the other hand, including customer flexibility leads to the solver showing large gaps even after an hour. As we have shown, using off-the-shelf solvers in a more smart way by iterating through carefully constructed subproblems can lead to relatively low optimality gaps within the same time limit. This therefore allows to overcome the computational drawback that comes with customer flexibility in order to fully exploit its ability to create more efficient supply chain networks, both in terms of cost and capacity utilization.

6 Conclusions

In this paper, we have introduced a variant of two echelon production-distribution problems, in which the location of intermediate facilities can change during the planning horizon. The two features of this problem are flexibility with respect to delivery dates and regarding the location of intermediate facilities. Although these features make the problem more difficult to solve, they better portray the reality of the business environment. Furthermore, we have provided a formal mathematical formulation for this problem. Generally, the integrated optimization problems are very difficult to be solved, specially when real-world features are incorporated. In this paper, we have proposed a decomposition approach capable of solving large size instances of the problem. The results obtained by this method are compared with the ones from a general purpose MIP solver. We observe that for majority of the instances, specially the very large ones, our approach outperforms the MIP solver. Important insights with respect to the building blocks of our algorithms are derived, and managerial insights are presented on the interaction between the two types of flexibility, namely customer flexibility and network flexibility, and the economies of scale they allow.

References

[1] C. Archetti, O. Jabali, and M. G. Speranza. Multi-period vehicle routing problem with due dates. *Computers & Operations Research*, 61:122–134, 2015.

[2] R. H. Ballou. Dynamic warehouse location analysis. *Journal of Marketing Research*, 5(3): 271–276, 1968.

[3] Nadjib Brahimi, Ali Cheaitou, Pierre Cariou, and Dominique Feillet. An exact algorithm for the single liner service design problem with speed optimisation. *International Journal of Production Research*, page forthcoming, 2020.

[4] I. Contreras, J.-F. Cordeau, and G. Laporte. The dynamic uncapacitated hub location problem. *Transportation Science*, 45(1):18–32, 2011.

[5] J.-F. Cordeau, F. Pasin, and M. M. Solomon. An integrated model for logistics network design. *Annals of Operations Research*, 144(1):59–82, 2006.

[6] J. Current, S. Ratick, and C. ReVelle. Dynamic facility location when the total number of facilities is uncertain: A decision analysis approach. *European Journal of Operational Research*, 110(3):597–609, 1998.

[7] M. Darvish and L. C. Coelho. Sequential versus integrated optimization: Production, location, inventory control, and distribution. *European Journal of Operational Research*, 268:203–214, 2018.

[8] M. Darvish, C. Archetti, L. C. Coelho, and M. G. Speranza. Flexible two-echelon location routing problem. *European Journal of Operational Research*, 277(3):1124–1136, 2019.

[9] E. D. Demaine, M. Hajiaghayi, H. Mahini, A. S. Sayedi-Roshkhar, S. Oveisgharan, and M. Zadimoghaddam. Minimizing movement. *ACM Transactions on Algorithms (TALG)*, 5(3):1–30, 2009.

[10] M. Eskandarpour, P. Dejax, J. Miemczyk, and O. Péton. Sustainable supply chain network design: An optimization-oriented review. *Omega*, 54:11–32, 2015.

[11] B. Fahimnia, A. Jabbarzadeh, A. Ghavamifar, and M. Bell. Supply chain design for efficient and effective blood supply in disasters. *International Journal of Production Economics*, 183: 700–709, 2017.

[12] R. Z. Farahani, M. Abedian, and S. Sharahi. Dynamic facility location problem. In R. Z. Farahani and M. Hekmatfar, editors, *Facility Location*, pages 347–372. Physica-Verlag HD, Heidelberg, 2009.

[13] A. M. Geoffrion and G. W. Graves. Multicommodity distribution system design by Benders decomposition. *Management Science*, 20(5):822–844, 1974.

[14] S. D. Jena, J.-F. Cordeau, and B. Gendron. Lagrangian heuristics for large-scale dynamic facility location with generalized modular capacities. *INFORMS Journal on Computing*, 29(3):388–404, 2017.

[15] S.D. Jena, J.-F. Cordeau, and B. Gendron. Solving a dynamic facility location problem with partial closing and reopening. *Computers & Operations Research*, 67:143–154, 2016.

[16] A. Klose. An LP-based heuristic for two-stage capacitated facility location problems. *Journal of the Operational Research Society*, 50(2):157–166, 1999.

[17] A. Klose and A. Drexl. Facility location models for distribution system design. *European Journal of Operational Research*, 162(1):4–29, 2005.

[18] M. K. Lim, H.-Y. Mak, and Z.-J. M. Shen. Agility and proximity considerations in supply chain design. *Management Science*, 63(4):1026–1041, 2017.

[19] M. T. Melo, S. Nickel, and F. Saldanha da Gama. Dynamic multi-commodity capacitated facility location: a mathematical modeling framework for strategic supply chain planning. *Computers & Operations Research*, 33(1):181–208, 2006.

[20] M. T. Melo, S. Nickel, and F. Saldanha da Gama. Facility location and supply chain management–a review. *European Journal of Operational Research*, 196(2):401–412, 2009.

[21] Alberto Moraglio, Cecilia Di Chio, Julian Togelius, and Riccardo Poli. Geometric particle swarm optimization. *Journal of Artificial Evolution and Applications*, 2008, 2008.

[22] F. Neves-Moreira, B. Almada-Lobo, J.-F. Cordeau, L. Guimarães, and R. Jans. Solving a large multi-product production-routing problem with delivery time windows. *Omega*, 86: 154–172, 2019.

[23] A. I. Nikolopoulou, P. P. Repoussis, C. D. Tarantilis, and E. E. Zachariadis. Moving products between location pairs: Cross-docking versus direct-shipping. *European Journal of Operational Research*, 256(3):803–819, 2017.

[24] S. H. Owen and M. S. Daskin. Strategic facility location: A review. *European Journal of Operational Research*, 111(3):423–447, 1998.

[25] Y. Qiu, J. Qiao, and P. M. Pardalos. Optimal production, replenishment, delivery, routing and inventory management policies for products with perishable inventory. *Omega*, 82: 193–204, 2019.

[26] S. Raghavan, M. Sahin, and F. S. Salman. The capacitated mobile facility location problem. *European Journal of Operational Research*, 277(2):507–520, 2019.

[27] A. Silva, D. Aloise, L. C. Coelho, and C. Rocha. Heuristics for the dynamic facility location problem with modular capacities. *European Journal of Operational Research*, 290(2):435–452, 2020.

[28] M. G. Speranza. Trends in transportation and logistics. *European Journal of Operational Research*, 264(3):830–836, 2018.

[29] G. Van der Heide, P. Buijs, K. J. Roodbergen, and I. F. A. Vis. Dynamic shipments of inventories in shared warehouse and transportation networks. *Transportation Research Part E: Logistics and Transportation Review*, 118:240–257, 2018.

[30] X. Zheng, M. Yin, and Y. Zhang. Integrated optimization of location, inventory and routing in supply chain network design. *Transportation Research Part B: Methodological*, 121:1–20, 2019.

[31] J. Zhi and B. B. Keskin. A multi-product production/distribution system design problem with direct shipments and lateral transshipments. *Networks and Spatial Economics*, 18(4): 937–972, 2018.

[32] J. Zhi, B. B. Keskin, and S. H. Melouk. A multi-period dynamic location planning model for emergency response. *IIE Transactions on Healthcare Systems Engineering*, 5(4):211–224, 2015.