Dynamic location of modular manufacturing facilities with relocation of individual modules

Andrew Allman, Qi Zhang

Department of Chemical Engineering and Materials Science, University of Minnesota, Minneapolis, MN 55455, USA

Abstract

Meeting highly variable product demands in a cost-efficient manner is an essential task for the chemical industry. Small-scale, modular, and mobile production units allow for a more agile response to spacial and temporal changes in demand while reducing the need of building new units. In this work, we present a generic mixed-integer linear programming (MILP) framework for determining optimal location and relocation of mobile production modules given time-varying demands. We introduce a new metric, the value of module mobility, to quantify the economic benefits of mobile production modules, and we demonstrate how it changes as a function of various economic parameters. Moreover, multiple different solution methods are developed to solve large instances of this dynamic modular and mobile facility location problem. First, we reformulate the original MILP by adding auxiliary variables which track the numbers of modules active at each site at any given time. This augmented formulation can be solved either directly using an off-the-shelf MILP solver, using the same solver but with priority branching on the auxiliary variables, or applying a branch-and-price algorithm. In the proposed branch-and-price algorithm, pricing subproblems for different time periods are solved separately and in parallel to generate new columns for the restricted master problem. Results from an extensive computational study show that solving the full-space augmented formulation is best when the number of time periods is small; however, the branch-and-price algorithm becomes superior for instances with a large number of time periods.

Keywords: Dynamic facility location, relocation, modular manufacturing, mobile facilities, branch-and-price

*Corresponding author
Email address: qizh@umn.edu (Qi Zhang)
1. Introduction

In today’s highly volatile markets, manufacturers face the increasing challenge of meeting distributed and fluctuating product demand. To remain competitive, companies have to be able to not only produce and distribute products at low cost but also respond to market changes quickly and cost-effectively. As a consequence, we are starting to see a paradigm shift toward smaller, more agile manufacturing systems, even in high-volume commodity industries that have traditionally profited from the economies of scale of large production facilities. Modular manufacturing systems can act as an enabling technology for distributed and agile manufacturing systems, specifically in the chemical industry (Bieringer et al., 2013; Baldea et al., 2017).

Construction of modular manufacturing systems considers standardized units, which we will refer to as modules, which are constructed off-site, transported to a production site, and then connected to other modules with minimal effort to form a working facility. Facilities can also be deployed in a modular fashion, with capacities increased or decreased incrementally by installing or disassembling additional modules. Both modular construction and deployment help to achieve a shorter time to market. The ease of assembly and disassembly also enables mobility of modules, particularly when individual modules are small in volume and can easily be loaded onto, for example, a tractor-trailer. These two key features of modular plants, modularity and mobility, can help substantially reduce financial risks and increase the flexibility in responding to demand changes (Bieringer et al., 2013; Baldea et al., 2017).

The benefits from modularity and mobility have to be assessed at the supply chain level since the impact will only be significant when we have multiple customer locations with changing (and possibly uncertain) demands. For this purpose, we introduce a new variant of the dynamic facility location problem (DFLP), which we refer to as the Dynamic Modular and Mobile Facility Location Problem (DMMFLP). Here, a facility consists of modules, and its production capacity depends on which modules and how many units of each module are at the facility location. Existing individual modules can be relocated, i.e. moved from one location to another, at given module- and origin-destination-specific costs. To the best of our knowledge, this is the first work that explicitly considers these types of movements and costs for individual modules in a DFLP. The objective of the DMMFLP is to decide where and when to locate and relocate modules over multiple time periods at the lowest cost while satisfying given demand. While primarily motivated by the modular manufacturing trend in the process industry, the DMMFLP and special cases thereof occur in a wide range of application areas, including the construction of temporary shelter and camp systems (Kilci et al., 2015), disaster response with mobile hospitals (Boonmee et al., 2017), and location planning for schools
The main contributions of this work are fourfold. First, we propose a mixed-integer linear programming (MILP) model for determining the optimal location and relocation of production modules within a given supply chain and a given planning horizon. Second, multiple solution strategies for solving this MILP are presented, including implementing priority branching on selected integer variables and applying a tailored branch-and-price algorithm. Third, the economic value of mobile production modules is established through a new metric and evaluated in a computational study. Finally, computational experiments on large instances of the DMMFLP are performed, analyzing the different solution strategies.

The remainder of this paper is organized as follows: In Section 2, we present a survey of related works, emphasizing the limitations of existing models with respect to the DMMFLP. Section 3 states the problem along with the associated notation, followed by Section 4, which introduces the proposed MILP model for the DMMFLP and presents mild assumptions under which the formulation can be further strengthened. Section 5 proposes multiple different solution methods which may improve the computational performance of solving the DMMFLP. In Section 6, a small illustrative example is provided which demonstrates the economic value of mobile and modular production facilities. The results of computational experiments on large instances of the DMMFLP are shown and discussed in Section 7. Finally, in Section 8, we provide concluding remarks and propose potential future extensions of this work.

2. Literature review

Most DFLPs are multiperiod extensions of classical location problems, allowing them to consider not only where but also when to locate a set of facilities. Early works in this area include Wesolowsky (1973), Cavalier and Sherali (1985), and Hakimi et al. (1999), which introduce the multiperiod extensions of the Weber problem, the minisum location problem, and the $p$-median problem, respectively.

Taking the time dimension into account allows dynamic problems to incorporate aspects that cannot be considered in static models, such as facility relocation (Wesolowsky and Truscott, 1975), phased capacity expansion (Luss, 1982), time-varying market conditions (Drezner and Wesolowsky, 1991), and inventory management (Melo et al., 2006). As a result, one can model a richer set of strategic decisions and more realistic problems with dynamic formulations (Owen and Daskin, 1998; Melo et al., 2009). Melo et al. (2006) propose such a DFLP formulation, which incorporates a large variety of features in strategic supply chain planning.
The two major considerations in facility location are the capacity of a facility and its location. In dynamic facility location, depending on the application, capacity changes (expansion and reduction) can be achieved by (1) opening new facilities, (2) expanding or contracting existing facilities, (3) closing existing facilities, (4) reopening closed facilities, and (5) relocating existing facilities. While many capacity planning models consider only the first two activities, relevant examples of models considering different activities are given in the following paragraphs.

Facility capacities can generally be modeled in two ways: as continuous variables or as sets of discrete capacity options (Luss, 1982). The latter is better suited for situations in which one can only choose among a discrete set of available facility types, especially if the number of facility types is relatively small (Shulman, 1991). In addition, discrete capacities allow specifying arbitrary facility-specific costs, e.g., to capture economies of scale, without introducing nonlinearities into the model. Several works consider facility location, closure, expansion, and contraction with discrete capacities (Antunes and Peeters, 2001; Troncoso and Garrido, 2005; Dias et al., 2007a). Wilhelm et al. (2013) developed a dynamic supply chain reconfiguration model that considers modular capacities for opening, closing, expanding and contracting multiple echelons of facilities with multiple products. Recently, Jena et al. (2015a, 2017) introduced a DFLP formulation with generalized modular capacities, which applies a cost matrix that accounts for the fact that the cost of adjusting capacity often depends not only on the size of adjustment but also on the current capacity level. Correia and Melo (2017) show that modular capacity adjustments in facility expansion and contraction can help to satisfy demands that are sensitive to lead times. In related work, modularized capacities have also been used in Alumur et al. (2016) for expanding capacity in multi-period hub network problems. However, to the best of our knowledge, existing DFLP models only consider modular capacities that depend on the number of available units but not also on the types, or functionalities, of units.

In some situations, it may be beneficial to temporarily or permanently close facilities in order to avoid maintenance costs. Also, closing and reopening facilities are often considered if capacities are obtained through leasing contracts. Van Roy and Erlenkotter (1982) and Hinojosa et al. (2008) address the problem of dynamic supply chain redesign in which one needs to decide when and where to open new and close existing facilities. Chardaire et al. (1996) and Canel et al. (2001) consider the DFLP in which facilities can be closed and reopened multiple times over the course of the planning horizon. Dias et al. (2007b) further incorporate different costs for opening new and reopening existing facilities. Partial closing and reopening of facilities has been considered by Jena et al. (2015b, 2016).
Relocation of facilities becomes an important aspect when capacities are movable and there is significant economic benefit from relocating existing capacities compared to building new ones. The relocation of a single facility has been considered by Farahani et al. (2009) and Bastian and Volkmer (1992). Melachrinoudis and Min (2000) present a DFLP formulation for a single facility whose capacity can be split and relocated to multiple other locations. Melo et al. (2006) consider the multifacility case that allows capacity transfer from a fixed set of existing facilities to new facilities; here, capacities at new facilities cannot be reduced over time. Jena et al. (2015b, 2016) introduce a DFLP model that allows the relocation of multiple facilities with the following restrictions: a facility can only be relocated as a whole, the entire facility has to be closed before it can be relocated, and facilities cannot be merged at the same location. Tan and Barton (2015, 2016) and Gao and You (2017) address the DFLP with relocation in the context of exploiting stranded natural gas resources. In most cited studies, relocation costs are assumed to only depend on the amount of capacity moved, the time of relocation, and in a few cases also on the distance moved. Only Melo et al. (2006) consider relocation costs that depend on the origin and destination, which is important in certain applications, e.g., if the facilities to be moved are large and international borders have to be crossed. Note that in some relocation problems, it is assumed that the cost of purchasing new capacities is the same as the cost of relocating existing capacities (Wesolowsky and Truscott, 1975; Min, 1988), which often applies in situations in which capacities are leased. In that case, the problem reduces to a DFLP with facility closing and reopening, although some may still refer to it as a relocation problem. More recently, Becker et al. (2019) studied the potential benefits of modular production with respect to opening, closing, expanding, contracting, and relocating facilities, as well as considering that facilities can perform multiple different production processes; however, the study also only allows for movement of an entire facility. This work provides a generalization to the models presented in Becker et al. (2019) and Jena et al. (2015b) in that it explicitly considers mobility of the individual modules present at a facility, instead of a facility as a whole.

3. Problem statement

We consider a single product for which there is a set of customers with forecast demands over the course of a given planning horizon. As in most DFLPs, the objective is to place a number of manufacturing facilities such that all demands are met and the overall cost is minimized. The distinctive feature of the DMMFLP is that here, each facility is composed of a set of modules. The operating cost and production capacity of a facility depend on its modules, specifically which modules and how many units of each module are installed.
To illustrate the concept of modularity, let us consider a facility that requires two different modules, modules A1 and B1. Both modules have to be installed in order to produce the product. One can interpret it as module A1 producing an intermediate product that is required in module B1, which produces the final product. Hence, neither one of the two modules alone is able to produce any final product. Figure 1a shows a facility composed of one unit of module A1 and one unit of module B1, resulting in a production capacity of 10. One way to increase the operating capacity of this facility is through “numbering up”: installing twice as many units of each module, as depicted in Figure 1b, doubles the production capacity. Moreover, the facility represented in Figure 1b can operate in two different configurations, one in which only one unit of each module operates, and one in which both units of each module operate. Here, a facility configuration represents an operating state of the facility with nonzero capacity and is defined by the set of installed modules operating. Additionally, if a specific module can perform multiple different tasks, examples of which are prevalent in the chemical scheduling literature and include separators and batch units (Harjunkoski et al., 2014), this can also be reflected through different facility configurations.

Figure 1: (a) Facility composed of one unit of module A1 and one unit of module B1, resulting in a capacity of 10, (b) facility composed of two units of each module, resulting in a capacity of 20.

Another way to increase the capacity of a facility is through “scaling up”. Consider that modules may not only differ in functionality but also in capacity. Assume that in addition to modules A1 and B1, we also have the choice of building modules A2 and B2. While modules A1 and A2 have the same functionality, A2 provides twice the capacity as A1. The same applies to modules B1 and B2. Figure 2 shows four facilities that are composed of different sets of modules but all have the same production capacity. The larger modules A2 and B2 will typically have a relatively smaller capital cost due to economies of scale, but this is traded off with the benefits seen from the smaller modules, such as flexible operation, lower operating costs, cheaper movement between production sites, and shorter time to market (Baldea et al., 2017). As such, the capital cost decreases from facility I to facility IV; however, the level of flexibility decreases as well. For example, the setup of facility I allows phased capacity expansion or contraction, as well as operation in a lower-capacity configuration, i.e. only operating one unit
of each module instead of both, whereas facility IV only has one possible configuration.

![Figure 2: Four different facility setups with the same production capacity.](image)

We assume that modules are mobile, i.e. they can be transported from one location to another. This would typically be achieved by loading the skid or container that the module is placed in onto a tractor trailer or train car and moving it to a new production site, in the same manner in which the module is brought to the site from its production facility. We further assume that modules are constructed into a working facility onsite, and that time required to assemble or disassemble a module at a production site is much smaller than the length of time considered within a time period. The objective of the DMMFLP is then to locate, i.e. purchase and set up, new modules and relocate existing modules such that the overall cost is minimized while satisfying all demand, given a planning horizon and a demand forecast.

4. Mathematical formulation

In this section, we present the proposed MILP model for the DMMFLP and various means to improve the formulation.

4.1. Notation

To formulate the problem, we introduce the following notation.

Sets:
- \( \mathcal{I} \) set of customers
- \( \mathcal{J} \) set of candidate and existing facility locations
- \( \mathcal{J}_j \) set of locations to which a module can be moved from location \( j \), \( \mathcal{J}_j \subseteq \mathcal{J} \setminus \{j\} \)
- \( \mathcal{J}_i \) set of locations from which a module can be moved to location \( j \), \( \mathcal{J}_i \subseteq \mathcal{J} \setminus \{j\} \)
- \( \mathcal{L} \) set of facility configurations
- \( \mathcal{M} \) set of modules
- \( \mathcal{M}_l \) set of modules required for facility configuration \( l \)
- \( \mathcal{T} \) set of time periods
Parameters:

- $c_{ijlt}$ unit cost of serving demand at customer $i$ from a facility operating in configuration $l$ at location $j$ in time period $t$
- $d_{it}$ demand at customer $i$ in time period $t$
- $f_{jlt}$ cost of operating a facility in configuration $l$ at location $j$ in time period $t$
- $g_{jmt}$ cost of purchasing and setting up one unit of module $m$ at location $j$ at the beginning of time period $t$
- $h_{jj'mt}$ cost of relocating one unit of module $m$ from location $j$ to location $j' \neq j$ at the beginning of time period $t$
- $n_{lm}$ number of units of module $m$ required for configuration $l$
- $u_{jl}$ capacity of configuration $l$ at location $j$
- $v_{jm}^0$ initial number of existing units of module $m$ at location $j$
- $v_{jmt}^{\text{max}}$ maximum number of units of module $m$ that can be located at location $j$ in time period $t$

Decision variables:

- $w_{jj'mt}$ number of units of module $m$ moved from location $j$ to location $j' \neq j$ at the beginning of time period $t$
- $x_{ijlt}$ fraction of the demand at customer $i$ in time period $t$ that is served from a facility at location $j$ operating in configuration $l$
- $y_{jlt}$ equal to 1 if facility at location $j$ operates in configuration $l$ in time period $t$, 0 otherwise
- $z_{jmt}$ number of units of module $m$ purchased and set up at location $j$ at the beginning of time period $t$

Note that the variable operating cost $c$ includes both the cost incurred from operation of a production facility $\hat{c}$, as well as the cost of transporting the product from production site to customer $\bar{c}$, such that:

$$c_{ijlt} = \hat{c}_{jlt} + \bar{c}_{ijt}.$$  \hfill (1)

The installation cost $g$ includes module capital costs, as well as any costs incurred from the transportation of a module from the (assumed to be fixed and known) location where it was produced to the facility location and installation of the module into a facility. The relocation cost $h$ includes transportation costs for moving a module between different production sites, as well as the costs for disassembly and assembly of that module at the respective locations.
All indexed parameters and variables are scalars. The vector that represents the concatenation of all scalar elements across the Cartesian product of the corresponding index sets is denoted by the same symbol, but without the indices. For example, \( x \) denotes the vector that contains all \( x_{ijlt} \) with \((i, j, l, t) \in I \times J \times L \times T\).

4.2. MILP formulation

Based on the definitions stated in the notation section, we formulate the following MILP model, which we refer to as the Modular and Mobile Facilities (MMF) formulation:

\[
\text{(MMF) minimize } \sum_{i \in I} \sum_{j \in J} \left( \sum_{l \in L} d_{it} x_{ijlt} + \sum_{l \in L} f_{jlt} y_{jlt} \right) + \sum_{m \in M} g_{jmt} z_{jmt} + \sum_{j' \in \bar{J}} u_{j'jmt} w_{j'jmt} \right) \\
\text{subject to } \sum_{j \in J} \sum_{l \in L} x_{ijlt} = 1 \quad \forall i \in I, t \in T \hspace{1cm} (3) \\
\sum_{i \in I} d_{it} x_{ijlt} \leq u_{jlt} y_{jlt} \quad \forall j \in J, l \in L, t \in T \hspace{1cm} (4) \\
n_{lm} y_{jlt} \leq v_{j}^{0} + \sum_{i=1}^{t} \left( z_{jmt} - \sum_{j' \in \bar{J}} w_{j'jmt} + \sum_{j' \in \bar{J}} w_{j'jmt} \right) \leq v_{jmt}^{\max} \hspace{1cm} \forall j \in J, l \in L, m \in M, t \in T \hspace{1cm} (5) \\
\sum_{l \in L} y_{jlt} \leq 1 \quad \forall j \in J, t \in T \hspace{1cm} (6) \\
x_{ijlt} \geq 0 \quad \forall i \in I, j \in J, l \in L, t \in T \hspace{1cm} (7) \\
y_{jlt} \in \{0, 1\} \quad \forall j \in J, l \in L, t \in T \hspace{1cm} (8) \\
w_{j'jmt} \in \mathbb{Z}_{+} \quad \forall j \in J, j' \in \bar{J}, m \in M, t \in T \hspace{1cm} (9) \\
z_{jmt} \in \mathbb{Z}_{+} \quad \forall j \in J, m \in M, t \in T. \hspace{1cm} (10)
\]

According to (2), the objective is to minimize the total cost of allocating demand, operating available facilities, purchasing and setting up modules, and relocating modules. Constraints (3) enforce that all customer demands are met. Constraints (4) are the production capacity constraints. Constraints (5) ensure that operation in a particular facility configuration is only possible when all required modules are available at the given location, and impose upper bounds on the numbers of modules at the same location. Constraints (6) state that at each location, only one facility configuration can be selected in each time period. If the fixed costs of closing a facility are zero or negligible, then
this constraint also allows for temporarily closing facilities during a time period, which occurs when all $y_{jlt}$ at a fixed $j$ and $t$ are decided to be zero. Finally, constraints (7)-(10) state the fraction of demand satisfied by a production facility is a nonnegative continuous variable, the choice of configuration is denoted by binary variables, the number of modules moved between locations is a nonnegative integer variable, and the number of modules built is a nonnegative integer variable, respectively.

Note that this formulation inherently also considers the location and relocation of entire facilities, which can be broken down into the location and relocation of its constituent modules. However, because individual modules can also be moved, this formulation is more general than previous DFLP frameworks that incorporate relocation. Note also that this formulation considers that some production locations $j$ are candidate locations (i.e. $v_{j,m}^0 = 0 \forall m \in M$), while others may already have existing facilities (i.e. some $v^0 \neq 0$). This provides another extension from the traditional DFLP, which typically assumes that no existing facilities are present at the beginning of the time horizon.

4.3. Valid inequalities

In addition to the model given above, we add the following set of strong inequalities proposed by Gendron and Crainic (1995):

$$x_{ijlt} \leq y_{jlt} \quad \forall i \in I, j \in J, l \in L, t \in T,$$

which states that demand $i$ cannot be serviced by production site $j$ operating in configuration $l$ unless the configuration is active.

4.4. Assumptions and resulting inequalities

In the following, we state mild assumptions that hold true in most practical instances, and apply these assumptions to derive inequalities that cut off integer feasible but suboptimal solutions.

Assumption 1. The cost of purchasing and setting up one unit of a module $m$ at a given location $j$ does not increase over time, i.e.

$$g_{jmt} \geq g_{jmt'} \quad \forall j \in J, m \in M, t, t' \in T, t < t',$$

which reflects the common notion of the time value of money ubiquitous in process design (Seider et al., 2016).

Assumption 2. The cost of purchasing and setting up one unit of a module $m$ at a location $j$ is no greater than purchasing and setting it up at another location $j'$ and
moving it to location $j$ in the same time period, i.e. for any $j \in \mathcal{J}$, $j' \in \hat{\mathcal{J}}_j$, and $m \in \mathcal{M}$,

$$g_{jmt} \leq g_{j'mt} + h_{j'jmt} \quad \forall t \in \mathcal{T}. \quad (13)$$

**Assumption 3.** For every module, the triangular inequality holds for its relocation costs with respect to all time periods, i.e. for any $j \in \mathcal{J}$, $j' \in \hat{\mathcal{J}}_j$, $j'' \in \hat{\mathcal{J}}_{j''}$, and $m \in \mathcal{M}$,

$$h_{jj'mt} + h_{jj''mt} \geq h_{jj''mt} \quad \forall t, t', t'' \in \mathcal{T}, t \leq t' \leq t''. \quad (14)$$

**Assumption 4.** The number of units of a module $m$ initially available at a given location $j$ is no greater than what is required for the facility configuration with the highest required number of units of module $m$, i.e.

$$v^0_{jm} \leq \max_{l \in \mathcal{L}_m} \{n_{lm}\} \quad \forall j \in \mathcal{J}, m \in \mathcal{M}, \quad (15)$$

where $\mathcal{L}_m$ denotes the set of facility configurations requiring module $m$.

**Proposition 1.** Let $v_{jmt}$ denote the number of units of module $m$ available at location $j$ in time period $t$, i.e.

$$v_{jmt} = v^0_{jm} + \sum_{t'=1}^t \left( z_{jmt'} - \sum_{j' \in \mathcal{J}_j} w_{j'jmt'} + \sum_{j' \in \hat{\mathcal{J}}_j} w_{j'jmt'} \right). \quad (16)$$

If Assumptions 1–4 apply, the following inequalities hold at the optimal solution of (MMF):

$$v_{jmt} \leq \max_{l \in \mathcal{L}_m} \{n_{lm}\} \quad \forall j \in \mathcal{J}, m \in \mathcal{M}, t \in \mathcal{T}. \quad (17)$$

**Proof.** This statement can be proved by contradiction, assuming that $v_{jmt} = \max_{l \in \mathcal{L}_m} \{n_{lm}\} + 1$ for some $j$, $m$, and $t$. Now consider the following two cases:

- **Case 1:** The state at $t$ remains, i.e. $v_{jmt'} = \max_{l \in \mathcal{L}_m} \{n_{lm}\} + 1$ for $t' = t + 1, \ldots, |\mathcal{T}|$. This case is clearly suboptimal since having $\max_{l \in \mathcal{L}_m} \{n_{lm}\} + 1$ units of module $m$ requires that at some $t'' \leq t$, module capital or transportation costs were incurred in installing this new unit per Assumption 4. However, this extra unit does not activate additional facility configurations compared to having $\max_{l \in \mathcal{L}_m} \{n_{lm}\}$ units, meaning no reduction of operating costs are possible to offset these added costs, and the total cost is necessarily greater than that where $v_{jmt} = \max_{l \in \mathcal{L}_m} \{n_{lm}\}$.

- **Case 2:** For some $t' > t$, $v_{jmt'} = \max_{l \in \mathcal{L}_m} \{n_{lm}\}$, i.e. one unit of module $m$ is
relocated at the beginning of time period \( t' \), incurring a relocation cost of \( h_{j'jt'} \), where \( j' \) denotes the location that the unit is moved to. The total cost of increasing \( v_{jmt} \) by 1 is either one of the following:

(a) \( g_{jmt'} + h_{j'jt'} \) if the unit was purchased and set up at location \( j \) at some \( t'' \leq t \). Considering Assumptions 1 and 2, this cost cannot be lower than the alternative of directly purchasing a unit of module \( m \) and setting it up at location \( j' \) at \( t' \):

\[
g_{j'jt'} \leq g_{jmt'} + h_{j'jt'}.
\]  
(18)

(b) \( h_{j'jt'} + h_{j'jt'} \) if the unit was moved from some location \( j'' \) to location \( j \) at some \( t'' \leq t \). As stated in Assumption 3, this cost cannot be lower than the alternative of relocating the unit directly from \( j'' \) to \( j' \) at \( t' \):

\[
h_{j'jt'} \leq h_{j'jt'} + h_{j'jt'}.
\]  
(19)

Thus, case 2 is also suboptimal.

This analysis can be easily extended to \( v_{jmt} = \max_{l \in L_m} \{ n_{lm} \} + n \) with \( n > 1 \). Hence, \( v_{jmt} > \max_{l \in L_m} \{ n_{lm} \} \) for any \( j, m, \) and \( t \) is shown to be suboptimal. Consequently, constraints (17) have to hold.

In the following corollaries, we assume that Assumptions 1–4 apply.

**Corollary 1.** As a direct consequence of Proposition 1, \( v_{jmt}^{\max} \) in constraints (5) can be replaced by \( \min \{ v_{jmt}^{\max}, \max_{l \in L_m} \{ n_{lm} \} \} \).

**Corollary 2.** The integer variables in (MMF) can be bounded from above as follows:

\[
z_{jmt} \leq \max_{l \in L_m} \{ n_{lm} \} \quad \forall j \in J, m \in M, t \in T \tag{20}
\]

\[
w_{j'jt'} \leq \max_{l \in L_m} \{ n_{lm} \} \quad \forall j \in J, j' \in J_j, m \in M, t \in T. \tag{21}
\]

**Proof.** According to Proposition 1, for \( z_{jmt} > \max_{l \in L_m} \{ n_{lm} \} \), one requires \( w_{j'jt'} > 0 \) for some \( j' \in J_j \). However, this is suboptimal due to Assumption 2, hence (20). Similarly, \( w_{j'jt'} > \max_{l \in L_m} \{ n_{lm} \} \) is suboptimal due to Assumption 3, hence (21).

5. Solution methods

In this section, we propose an augmented formulation of the DMMFLP, along with a priority branching strategy and a tailored branch-and-price algorithm based on
Dantzig-Wolfe decomposition (Dantzig and Wolfe, 1960; Vanderbeck, 2000; Singh et al., 2009), all of which may be helpful in solving large instances of the DMMFLP.

5.1. DMMFLP augmented formulation

To improve computational performance and enable additional solution methods, we first augment (MMF) with additional variables and constraints as follows:

\[
\text{(MMF') minimize } \sum_{t \in T} \left[ \sum_{j \in J} \sum_{m \in M} \left( g_{jmt} z_{jmt} + \sum_{j' \in J} h_{jj'mt} w_{jj'mt} \right) + \sum_{j \in J} \sum_{l \in L} \left( \sum_{i \in I} c_{ijlt} d_{ilt} x_{ijlt} + f_{jlt} y_{jlt} \right) \right]
\]

subject to constraints (3), (4), (6)–(10)

\[
v_{jmt} = v_{0jm} + \sum_{t'=1}^t \left( z_{jmt'} - \sum_{j' \in J} w_{jj'mt'} + \sum_{j' \in J} w_{jj'jmt'} \right)
\]

\forall j \in J, m \in M, t \in T

\[
n_{lm} y_{ilt} \leq v_{jmt} \leq v_{\max jmt} \forall j \in J, l \in L, m \in M_l, t \in T
\]

\[
v_{jmt} \in \mathbb{Z}_+ \forall j \in J, m \in M, t \in T
\]

where we have introduced \(v_{jmt}\), the number of units of module \(m\) located at location \(j\) at the beginning of time \(t\), which is explicitly defined as an integer variable. The additional constraints (23) can be interpreted as conservation equations for the number of modules at a site at any time. Note that the valid inequality (11) and constraints resulting from Corollary 2 (equations (20)-(21)) can also be added to strengthen the formulation.

5.2. Priority branching

When solving problems with many integer variables using a branching scheme, as is done in state-of-the-art MILP solvers such as CPLEX as well as in the branch-and-price approach that will be discussed in Section 5.3, the branching strategy can have a significant impact on the computational performance. The most common approach to branching is to first branch on the variable which has the largest deviation from integrality. CPLEX itself uses an adaptive approach which uses both the aforementioned strategy along with a pseudo-shadow pricing approach, which estimates the cost of enforcing a branching constraint using dual variables (Nygreen, 1991), and a strong branching approach that selects variables by partially solving a number of subproblems to determine the most promising branch.
Inspired by similar reformulation and branching strategies applied in batch scheduling (Velez and Maravelias, 2013) and capacity planning (Flores-Quiroz et al., 2019), we attempt to improve the computational performance by giving higher priority to branching on the new \(v\)-variables. Branching on these auxiliary variables can quickly eliminate solutions with the same number of modules, which leads to the elimination of many symmetric solutions. Additionally, values of \(v\) at later times will be inherently dependent on the values at earlier times, since, for example, \(v_{t+1}\) will equal \(v_t\) plus contributions due to building or moving modules. Therefore, it is expected that any nonintegrality in \(v\) at early times will propagate through to later times, and that branching on \(v\) with smaller time indices can help prune part of the branch-and-bound tree such that branching on the corresponding \(v\)-variables with larger time indices does not need to be further considered. Thus, to solve (MMF') more efficiently, branching priorities are manually set to be higher for \(v\)-variables than all other discrete variables, and further increased with decreasing time index.

5.3. Branch-and-price algorithm

Notice that if constraints (23) were removed, (MMF') could be decomposed into \(|\mathcal{T}|\) independent subproblems, one for each time period. We propose to exploit this special structure of the problem using a branch-and-price algorithm. We first define for each \(t \in \mathcal{T}\) a set

\[
\mathcal{V}_t = \{ v_t : (3), (4), (6)-(8), (24), (25) \},
\]

where the constraints describing \(\mathcal{V}_t\) are written for the specific time period \(t\) and \(v_t \in \mathbb{Z}^{|\mathcal{J}| |\mathcal{M}|}\) is the vector containing all \(v_{jmt}\). Since all \(v_{jmt}\) are bounded integer variables, \(\mathcal{V}_t\) is a set of a finite number of discrete points. Thus, it can be rewritten as \(\mathcal{V}_t = \{ v^*_{tk} : k \in \mathcal{K}_t \} \) with \(\mathcal{K}_t\) denoting the full set of discrete points and \(v^*_{tk}\) being the vector associated with point \(k\). Any element of \(\mathcal{V}_t\) can then be expressed through

\[
v_{jmt} = \sum_{k \in \mathcal{K}_t} p_{tk} v^*_{jmtk} \quad \forall j \in \mathcal{J}, m \in \mathcal{M}
\]

(27) 

\[
\sum_{k \in \mathcal{K}_t} p_{tk} = 1
\]

(28) 

\[
p_{tk} \in \{0, 1\} \quad \forall k \in \mathcal{K}_t.
\]

(29) 

We can further assume that each \(v^*_{tk}\) has associated with it at least one optimal solution \((x_{tk}^*, y_{tk}^*)\) with respect to the cost function \(\zeta(x_t, y_t) = \sum_{j \in \mathcal{J}} \sum_{l \in \mathcal{L}} \left( \sum_{i \in \mathcal{I}} c_{ijlt} d_{it} x_{ijlt} + f_{jlt} y_{jlt} \right)\). Thus, we can attach a minimum cost \(\zeta^*_{tk}\) to each \(v^*_{tk}\).
Substituting the above expressions for $v_{jmt}$ and $\zeta_t$ into (MMF'), we arrive at the following Dantzig-Wolfe reformulation:

\begin{equation}
\text{(MP)} \quad \minimize \sum_{t \in T} \left[ \sum_{j \in J} \sum_{m \in M} \left( g_{jmt} z_{jmt} + \sum_{j' \in J_j} h_{j'jmt} w_{jj'mt} \right) \right] + \sum_{k \in K_t} p_{tk} \zeta_{tk}^*
\end{equation}

subject to constraints (9)-(10)

\begin{align}
\sum_{k \in K_t} p_{tk} v_{jmtk}^* & \leq v_{jm0}^* + \sum_{t' = 1}^t \left( z_{jmt'} - \sum_{j' \in J_j} w_{jj'mt'} + \sum_{j' \in \tilde{J}_j} w_{jj'jmt'} \right) \\
\sum_{k \in K_t} p_{tk} & = 1 \quad \forall t \in T \\
p_{tk} & \in \{0, 1\} \quad \forall t \in T, k \in K_t
\end{align}

Note that constraints (31) are formulated as inequalities, as incorporating inequalities instead of equalities in the master problem is well known to help stabilize dual prices in column generation algorithms (Vanderbeck, 2005). Since the cardinality of $K_t$ is likely to be huge, we solve (MP) using a branch-and-price procedure (Barnhart et al., 1998; Lübbecke and Desrosiers, 2005). We formulate a restricted master problem, (RMP), which is identical to (MP) except that instead of the full set of columns $K_t$, a subset $\bar{K}_t \subseteq K_t$ is considered. We also define the set of discrete points these columns correspond to as $\bar{V}_t = \{ v_{t_k}^* : k \in K_t \}$. At each node of the branch-and-bound tree, the linear relaxation of (MP), which we denote (MP-LP), is solved using dynamic column generation, i.e. by solving a sequence of (RMP-LP), the linear relaxation of (RMP), with updated column sets $\bar{K}_t$. The algorithm can significantly reduce the computational if $\bar{K}_t$ can be kept at a manageable size.

To identify new columns to be added to the restricted master problem, we solve the following pricing subproblem for each $t \in T$:

\begin{equation}
\text{(SP)}_t \quad \minimize \sum_{j \in J} \sum_{l \in L} \left( \sum_{i \in I} c_{ijlt} x_{ijlt} + f_{jlt} y_{jlt} \right) - \sum_{j \in J} \sum_{m \in M} \pi_{jmt} v_{jmt} - \mu_t
\end{equation}

subject to

\begin{align}
\sum_{j \in J} \sum_{l \in L} x_{ijlt} & = 1 \quad \forall i \in I \\
\sum_{i \in I} d_{ilt} x_{ijlt} & \leq u_{jlt} y_{jlt} \quad \forall j \in J, l \in L
\end{align}
which minimizes the reduced cost, and where \( \pi_{jmt} \) and \( \mu_t \) are the dual variables associated with constraints (31) and (32), respectively. Note that each pricing subproblem (SP\(_t\)) can be solved independently for all \( t \in \mathcal{T} \). As such, we apply a parallel implementation of the branch-and-price algorithm to reduce the solution time.

Convergence of column generation and branch-and-price algorithms is known to exhibit a tailing-off effect, converging to a near-optimal solution quickly but making much less progress per iteration as the optimality gap becomes smaller. To combat this issue, many different stabilization approaches have been proposed, most of which slightly modify the restricted master problem to give better control of the dual variables (Lübbecke and Desrosiers, 2005). For this application, (RMP-LP) is modified using the approach proposed by du Merle et al. (1999) as follows:

\begin{align*}
(RMPLPST) \quad \text{minimize} & \quad \sum_{t \in \mathcal{T}} \left[ \sum_{j \in \mathcal{J}} \sum_{m \in \mathcal{M}} \left( g_{jmt} z_{jmt} + \sum_{j' \in \mathcal{J}_j} h_{jj'mt} w_{jj'mt} ight) 
\quad + \delta^+_{jmt} s^+_{jmt} + \delta^-_{jmt} s^-_{jmt} \right] + \sum_{k \in \mathcal{K}_t} p_{tk} \hat{\xi}_{tk} \\
\text{subject to} & \quad \sum_{k \in \mathcal{K}_t} p_{tk} v^*_{jmtk} + s^+_{jmt} - s^-_{jmt} = \\
& \quad v^0_{jmt} + \sum_{t' = 1}^t \left( z_{jmt'} - \sum_{j' \in \bar{\mathcal{J}}_j} w_{jj'mt'} + \sum_{j' \in \check{\mathcal{J}}_j} w_{j'jmt'} \right) \quad \forall j \in \mathcal{J}, m \in \mathcal{M}, t \in \mathcal{T} \\
& \quad \sum_{k \in \mathcal{K}_t} p_{tk} = 1 \quad \forall t \in \mathcal{T} \\
& \quad 0 \leq s^+_{jmt} \leq \epsilon^+_{jmt} \quad \forall j \in \mathcal{J}, m \in \mathcal{M}, t \in \mathcal{T} \\
& \quad 0 \leq s^-_{jmt} \leq \epsilon^-_{jmt} \quad \forall j \in \mathcal{J}, m \in \mathcal{M}, t \in \mathcal{T} \\
& \quad 0 \leq p_{tk} \leq 1 \quad \forall t \in \mathcal{T}, k \in \mathcal{K}_t \\
& \quad w_{jj'mt} \geq 0 \quad \forall j \in \mathcal{J}, j' \in \check{\mathcal{J}}_j, m \in \mathcal{M}, t \in \mathcal{T}
\end{align*}
This formulation relaxes constraints (31) by introducing slack variables, $s^-_{jmt}$ and $s^+_{jmt}$, which allow deviation above or below the equality by as much as $\epsilon^-_{jmt}$ and $\epsilon^+_{jmt}$, respectively. These deviations are penalized in the objective through multiplication with penalty factors $\delta^-_{jmt}$ and $\delta^+_{jmt}$. By formulating the dual of (RMP-LP-ST), it can also be shown that $-\delta^-_{jmt}$ and $\delta^+_{jmt}$ provide lower and upper bounds, respectively, on the dual variables $\pi_{jmt}$, with deviations outside these bounds being penalized by $\epsilon^-_{jmt}$ and $\epsilon^+_{jmt}$.

At each iteration of the column generation algorithm, the lower dual bound is updated to be the average of the current dual lower bound and the dual variable returned by (RMP-LP-ST), i.e. $\delta^- \leftarrow (\delta^- + \pi)/2$. Additionally, if at a certain time $t$, no new column is found that gives a negative reduced cost, the primal lower bound is tightened, i.e. $\epsilon^- \leftarrow \epsilon^- / 2$. This ensures that (RMP-LP-ST) converges to (RMP-LP) as the number of iterations increases.

Since (RMP-LP-ST) is a relaxation of (RMP-LP), it does not provide an upper bound of the linear relaxation of (MMF'). Thus, we solve both problems at each iteration of the column generation algorithm: (RMP-LP-ST) is used to obtain values for the dual variables needed for the subproblems, while (RMP-LP) is used to obtain an upper bound. The function for processing a single branch-and-bound node is given in Algorithm 1, where $BC$ is a set of branching constraints, $UB$ is a global upper bound, $\bar{\text{LB}}$ and $\bar{\text{UB}}$ are lower and upper bounds, respectively, for the local column generation function, and gap $L_P$ and tol $L_P$ are the current and stopping relative optimality gaps, respectively, for solving the LP using column generation. The column initialization algorithm is given in Algorithm 2, where $LB$ is a global lower bound, $N$ is the number of nodes to process, $n$ is the current node being processed, $BC_n$ is a set of branching constraints for node $n$, and gap $L_P$ and tol $L_P$ are the current and stopping relative optimality gaps, respectively, for solving problem using branch-and-price. Note that the same branching procedure is applied in the branch-and-price algorithm as was presented in Section 5.2, with $v$-variables calculated by applying the $p$-variables determined from the solution of (RMP-LP) to equation (27). Because the $v$-variables do not explicitly appear in (RMP), branching is achieved by adding one of the following two constraints:

\[
\sum_{k \in K_n} p_{m \cdot t \cdot k} v^*_{j \cdot m \cdot t \cdot k} \geq \lceil \hat{v} \rceil \tag{50}
\]

\[
\sum_{k \in K_n} p_{m \cdot t \cdot k} v^*_{j \cdot m \cdot t \cdot k} \leq \lfloor \hat{v} \rfloor, \tag{51}
\]

where $\hat{v}$ is a noninteger value chosen based on the previous solution. In the unlikely
case where all \( v \)-variables are integer but the integrality gap has not closed, we then branch on the \( w \)-variable with largest deviation from integrality.

**Algorithm 1** Column generation function

```plaintext
function ColumnGen(UB, \( \bar{V} \), \( \delta^- \), \( \epsilon^- \), BC) do
    LB ← -∞, UB ← ∞, gap_{LP} ← ∞
    while gap_{LP} > tol_{LP} do
        Solve (RMP-LP) with constraints BC to obtain optimal value \( q^* \) and decisions \( p, w \)
        if (RMP-LP) infeasible then
            return -∞, ∞, \( \bar{V} \), \( \bar{p} \), \( \bar{w} \), \( \delta^- \), \( \epsilon^- \) and stop
        else
            UB ← \( q^* \), \( \bar{p} \) ← \( p \), \( \bar{w} \) ← \( w \)
            Solve (RMP-LP-ST) with constraints BC to obtain dual variables \( \pi, \mu \)
            for all \( t \in T \) do
                Solve (SP) \( t \) to obtain reduced cost \( r^*_t \), column \( v^*_t \), and decisions \( x^*_t, y^*_t \)
                if \( r^*_t < 0 \) and \( v^*_t \notin \bar{V}_t \) then
                    Add column \( v^*_t \) to \( \bar{V}_t \) with cost \( \zeta^*_t = \sum_{j \in J} \sum_{m \in M} \left( \sum_{i \in I} c_{ijlt} d_{it} x^*_{ijlt} + f_{jlt} y^*_{jlt} \right) \)
                else
                    \( \epsilon^- \) ← \( \epsilon^- / 2 \)
                end if
            end for
            LB ← \( \max \{ LB, UB + \sum_{t \in T} \min\{ r^*_t, 0 \} \} \), gap_{LP} ← \( \frac{UB - LB}{UB} \), \( \delta^- \) ← \( \delta^- + \pi / 2 \)
            if UB > LB then
                return -∞, ∞, \( \bar{V} \), \( \bar{p} \), \( \bar{w} \), \( \delta^- \), \( \epsilon^- \) and stop
            end if
        end if
    end while
    return LB, UB, \( \bar{V} \), \( \bar{p} \), \( \bar{w} \), \( \delta^- \), \( \epsilon^- \)
end function
```

6. Illustrative example

To analyze the benefits of mobile production modules, we define a new metric, the value of module mobility (VMM), as follows:

\[
VMM = q^*_{w=0} - q^* \tag{52}
\]

where \( q^* \) denotes the minimum cost obtained from solving the DMMFLP and \( q^*_{w=0} \) denotes the minimum cost from solving the same problem but without allowing modules to be relocated (that is, with \( w \) fixed to zero). As \( q^*_{w=0} \) is obtained from solving a restricted DMMFLP, the VMM is always nonnegative.
Algorithm 2 Branch-and-price algorithm

Set: $LB \leftarrow -\infty$, $UB \leftarrow \infty$, $N \leftarrow 1$, $n \leftarrow 1$, $\mathcal{BC}_1 \leftarrow \emptyset$

for all $j \in \mathcal{J}$, $m \in \mathcal{M}$, $t \in \mathcal{T}$ do
  Initialize $\delta^-_{jmt}$, $\delta^+_{jmt}$, $\epsilon^-_{jmt}$, $\epsilon^+_{jmt}$
  $\pi_{jmt} \leftarrow 0$
end for

for all $t \in \mathcal{T}$ do
  $\mu_t \leftarrow 0$, $\bar{V}_t \leftarrow \emptyset$
  Solve (SP) to obtain optimal value $r^*_t$ and column $v^*_t$
  Add column $v^*_t$ to $\bar{V}_t$ with column cost $r^*_t$
end for

while $n \leq N$ do
  $LB, q^*_{LP}, \bar{V}, \hat{p}, \bar{w}, \hat{\delta}^-, \hat{\epsilon}^- \leftarrow$ ColumnGen($UB, \bar{V}, \delta^-, \epsilon^-, \mathcal{BC}_n$)
  if $q^*_{LP} < \infty$ then
    Solve (RMP) to obtain optimal cost $q^*_{IP}$ and decisions $p$, $w$, $z$.
    $UB \leftarrow \min\{UB, q^*_{IP}\}$
    $\text{gap}_{IP} \leftarrow \frac{|UB - LB|}{LB}$
    if $\text{gap}_{IP} > \text{tol}_{IP}$ then
      $\bar{v}_{jmt} \leftarrow \sum_{k \in \mathcal{K}} \tilde{p}_{kt} v^*_{jmtk}$ \quad $\forall j \in \mathcal{J}$, $m \in \mathcal{M}$, $t \in \mathcal{T}$
      if noninteger element of $\bar{v}$ exists then
        $\hat{v} \leftarrow$ noninteger value of $\bar{v}$ with smallest time index
        $j^* \leftarrow$ index $j$ of $\hat{v}$, $m^* \leftarrow$ index $m$ of $\hat{v}$, $t^* \leftarrow$ index $t$ of $\hat{v}$
        $\mathcal{BC}_{N+1} \leftarrow \mathcal{BC}_n \cup \{v_{j^*m^*t^*} \geq \lceil \hat{v} \rceil\}$
        $\mathcal{BC}_{N+2} \leftarrow \mathcal{BC}_n \cup \{v_{j^*m^*t^*} \leq \lfloor \hat{v} \rfloor\}$
      else
        $\hat{w} \leftarrow$ with largest deviation from integrality
        $j^* \leftarrow$ index $j_1$ of $\hat{w}$, $j^*_2 \leftarrow$ index $j_2$ of $\hat{w}$, $m^* \leftarrow$ index $m$ of $\hat{w}$, $t^* \leftarrow$ index $t$ of $\hat{w}$
        $\mathcal{BC}_{N+1} \leftarrow \mathcal{BC}_n \cup \{w_{j^*2m^*t^*} \geq \lceil \hat{w} \rceil\}$
        $\mathcal{BC}_{N+2} \leftarrow \mathcal{BC}_n \cup \{w_{j^*2m^*t^*} \leq \lfloor \hat{w} \rfloor\}$
      end if
    end if
  end if
  $n \leftarrow n + 1$
end while

To determine the fundamental relationship between relevant economic parameters and the VMM, we develop a minimum working example case consisting of two nodes. Each node serves as both a candidate production site and customer location, and as such there is no transportation cost from moving product from a production site to its corresponding customer location. Production modules of two different sizes are allowed to be installed at a production site. The capital cost of installing a new large module is assumed to be related to that of the smaller module through the use of a capacity
The capacity exponent $\gamma$ typically takes values between 0.6 and 1 depending on the economies of scale present for the particular module (Seider et al., 2016). This equation assumes transportation and installation costs are negligible in comparison to capital costs.

A planning horizon of two time periods is considered. In the first time period, market 1 has a demand equal to the capacity of unit 1, $u_1$, while market 2 has a demand equal to the capacity of unit 2, $u_2$. In the second time period, these demands switch such that market 1 has a demand of $u_2$ and market 2 has a demand of $u_1$. The following assumptions are made, which will allow us to obtain an analytical form of the VMM:

(i) There are no fixed operating costs, i.e. $f = 0$.
(ii) Module relocation costs are proportional to the module capacity, such that $h = \bar{h}u$ with $\bar{h}$ being a given cost parameter.
(iii) Variable operating costs are independent of location, configuration, and time period, such that we have $\check{c} := \check{c}_{1t} = \check{c}_{2t}$ and $\check{c} := \check{c}_{12t} = \check{c}_{21t}$.
(iv) Building a new module is more expensive than transporting its capacity’s worth of goods between the two sites, i.e. $g_i > \bar{c}u_i$.
(v) The capacities of the two units differ by a factor of 2, such that $u_2 = 2u_1$.

Since the total demand is $u_1 + u_2$ in both time periods, assumption (iv) implies that the total installed capacity will always be equal to the total demand, as building additional capacity would be suboptimal. Then, due to the relationship between capital costs described earlier, the “immobile” optimal supply chain will always build one large and one small module and transport $u_2 - u_1$ of product between the two sites in one of the two time periods. Thus, the optimal cost of the immobile supply chain is

$$q_{w=0}^* = g_1 + g_2 + 2(u_1 + u_2)\check{c} + (u_2 - u_1)\bar{c}. \tag{54}$$

The mobile supply chain will choose between three possible strategies. The first strategy is the same as the optimal immobile supply chain. The second is one where one large and one small module are built, and both are moved between production sites after the first time period. The final option builds three small modules and moves one between production sites after the first time period. Hence, the optimal cost of the
mobile supply chain is
\[ q^* = 2(u_1 + u_2)\bar{c} + \min\{g_1 + g_2 + (u_2 - u_1)\bar{c}, g_1 + g_2 + (u_1 + u_2)\bar{h}, 3g_1 + u_1\bar{h}\}. \] (55)

Then, combining equations (54) and (55), as well as considering the relationship between the module capital costs, equation (53), gives the following analytical form of the VMM:
\[ VMM = \max\{0, u_1(\bar{c} - 3\bar{h}), (2\gamma - 2)g_1 + u_1(\bar{c} - \bar{h})\}. \] (56)

From (56), it is clear that the VMM depends on three parameters, the transportation cost \( \bar{c} \), the module relocation cost \( \bar{h} \), and the difference in capital cost of building a certain capacity with two modules of different sizes, \( (2 - 2\gamma)g_1 \). The impact of these parameters is shown in the heat maps in Figure 3, where \( u_1 = 50 \). One can see that the VMM increases with increasing \( \bar{c} \) and decreasing \( \bar{h} \). The shape of the contours is clearly dependent on the capital cost parameters. The red contour in each figure corresponds to when the optimal strategy of the mobile supply chain switches from the immobile strategy, on the left of the red contour, to one that involves moving modules, on the right of the red contour. This red contour is the maximum of two lines, one defined by \( 50(\bar{c} - 3\bar{h}) = 0 \), and the other defined by \( (2\gamma - 2)g_1 + 50(\bar{c} - \bar{h}) = 0 \). The “kink” in the red contour occurs where these two lines intersect, and thus is dependent on the capital cost parameters. It is apparent from the figures that when the difference in cost between building one large and two small modules is small, mobility is taken advantage of in more situations since it is cheaper to move one small module than a small and a large module.

![Figure 3: Heat maps for VMM with (a) \( g_1 = 8000 \), \( \gamma = 0.95 \); (b) \( g_1 = 10,000 \), \( \gamma = 0.8 \); and (c) \( g_1 = 12,000 \), \( \gamma = 0.7 \).](image-url)
7. Computational experiments

In this section, larger instances of the DMMFLP are solved. All models were implemented in Julia 0.6.4 using JuMP 0.18.5 (Dunning et al., 2017) and solved using CPLEX 12.8. All computations in this section were performed on the Mesabi cluster of the Minnesota Supercomputing Institute, a Linux cluster equipped with a set of 2.5 GHz Intel Haswell E5-2680v3 processors. The number of cores used is dependent on the number of time periods considered in the problem such that one core per time period is used (i.e. a 48-time-period problem uses 48 cores). Problems are solved to a 0.5% optimality gap or time out after 10,000 s, just under three hours.

7.1. Value of module mobility

Section 6 demonstrated the effect of supply chain economic parameters on the VMM. In this subsection, the effect of changing customer demands and locations is analyzed using a larger instance of the DMMFLP with 50 customers, 10 production sites, and 3 module sizes. Economic parameters are held constant at the values listed in Table 1, with the largest module being twice as large as the medium sized module, which itself is twice as large as the small module. In this table, $\Delta_{ab}$ denotes the distance between nodes $a$ and $b$, which can be either production site or customer nodes. Capital costs for larger modules are determined using equation (53), while, consistent with assumption 1, capital costs in time periods beyond the first are reduced by using a net present cost assuming a 10% discount rate per time period. All customer and production site locations are randomly placed on a 100x100 grid. Customer demands are randomly generated in a number of ways:

- Randomly generate a demand from the distribution $d = \mathcal{U}(0, \frac{|J| u_{\text{max}}}{|I|})$.
- Randomly generate a mean from the distribution $\mu = \mathcal{U}(0, \frac{|J| u_{\text{max}}}{|I|})$, then randomly generate a demand from the distribution $d = \mathcal{U}(0, 2\mu)$.
- Generate a mean as a function of customer location $\ell$ and time period $t$, such that $\mu(\ell, t) = \frac{|J| u_{\text{max}} (100 - |\ell_1 + 90(1-t) - 5|)}{100|J|}$, then randomly generate a demand from the distribution $d = \mathcal{U}(0, 2\mu)$.

We hypothesize that module mobility is most beneficial when large changes in demand occur in both space and time. To better quantify this, we introduce the “demand center of mass” (DCM) metric, which is a weighted average of customer locations $\ell_i$ based on their demands

$$\text{DCM}_t = \frac{\sum_{i \in I} d_{it} \ell_i}{\sum_{i \in I} d_{it}},$$ (57)
Table 1: Parameters used in large VMM study

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product transportation cost per unit distance ($\bar{c}<em>{ij}/\Delta</em>{ij}$)</td>
<td>60</td>
</tr>
<tr>
<td>Operating cost per unit capacity ($\bar{c}$)</td>
<td>5</td>
</tr>
<tr>
<td>Small module reference capital cost ($g_1$)</td>
<td>80,000</td>
</tr>
<tr>
<td>Capacity exponent for capital cost ($\gamma$)</td>
<td>0.9</td>
</tr>
<tr>
<td>Module transportation cost per unit distance ($\bar{h}<em>{ij'}/\Delta</em>{ij'}$)</td>
<td>5</td>
</tr>
</tbody>
</table>

Changes in the location of the DCM over time, then, should be related to the VMM. For example, the average movement of the DCM between consecutive time periods and the variance in the DCM are both plausible driving forces for module mobility. Figure 4 shows the solution of one instance of the DMMFLP with demands generated from a mean function of both location and time. The normalized VMM; that is, the VMM divided by the optimal cost for an immobile supply chain, has a value of 3.31% for this instance. Note how the DCM moves from left to right over time, promoting module mobility.

In Figure 5, we show the results from 60 instances of the DMMFLP with 4 time periods. Here, the normalized VMM is plotted against two different possible driving forces for mobility. The results show that variance in the DCM correlates slightly better with the VMM than does the average change in DCM between consecutive time periods. However, these results support the expected outcome that variability in demand in both space and time promotes module mobility within the supply chain.

7.2. Computational performance: scaling with time periods

In this subsection, the computational performance for solving large instances of the DMMFLP using four different solution approaches is analyzed. Namely, the following approaches are considered: the “full space” approach, which solves (MMF) directly using CPLEX, the “reformulation” approach, which solves (MMF’) directly using CPLEX, the “priority branching” approach, which solves (MMF’) directly using CPLEX while setting branching priorities as specified in section 5.2, and the “column generation” approach, which solves (MMF’) using the branch-and-price algorithm specified in Algorithms 1 and 2 with CPLEX used as the solver for all master and subproblems. Note that to ensure a fair comparison, CPLEX is called with 1 processor for subproblems in the column generation approach, and with $|T|$ processors in all other approaches. Stabilization parameters for column generation are initialized as $\delta^- = -10,000$ and $\epsilon^- = 0.01$ for all instances.

In order to analyze how each solution approach scales with problem size, a supply chain architecture with $|J| = 10$ production sites and $|I| = 50$ customers is considered.
Figure 4: Example supply chain result for instance with 4 time periods with normalized VMM of 3.31% for (a) $t = 1$, (b) $t = 2$, (c) $t = 3$, (d) $t = 4$. Arrow weights correspond to quantity of demand supplied to customer. Modules installed at production sites are indicated by differently sized squares, customers are indicated by circles, and the demand center of mass is indicated by a star.
Figure 5: Normalized value of modular mobility plotted against average change in demand center of mass in consecutive time periods and variance in demand center of mass.

Table 2: Distributions used for generating random problem instances.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Random Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Production and customer locations ($\ell_i$ or $\ell_j$)</td>
<td>$\mathcal{U}(0, 100), \mathcal{U}(0, 100)$</td>
</tr>
<tr>
<td>Distance between nodes $a$ and $b$ ($\Delta_{ab}$)</td>
<td>$\mathcal{U}(|\ell_a - \ell_b|_2, |\ell_a - \ell_b|_1)$</td>
</tr>
<tr>
<td>Product transportation cost per unit distance ($\bar{c}<em>{ij}/\Delta</em>{ij}$)</td>
<td>$\mathcal{U}(20, 60)$</td>
</tr>
<tr>
<td>Demand for customer $i$ in time $t$ ($d_{it}$)</td>
<td>$\mathcal{U}(\frac{\ell_{it}}{|\ell_i - \ell_j|_1})$</td>
</tr>
<tr>
<td>Operating cost per unit capacity ($\hat{c}$)</td>
<td>$\mathcal{U}(5, 20)$</td>
</tr>
<tr>
<td>Small module reference capital cost ($g_{j11}, g_{j31}$ in 2x2-module case)</td>
<td>$\mathcal{U}(10^4, 10^5)$</td>
</tr>
<tr>
<td>Capacity exponent for capital cost ($\gamma$)</td>
<td>$\mathcal{U}(0.7, 0.95)$</td>
</tr>
<tr>
<td>Small module relocation cost per unit distance ($h_{jj'1}/\Delta_{jj'}$,</td>
<td>$\mathcal{U}(50, 150)$</td>
</tr>
<tr>
<td>$h_{jj'3}/\Delta_{jj'}$ in 2x2-module case)</td>
<td></td>
</tr>
<tr>
<td>Factor relating relocation costs between modules of different sizes ($h_{jj'21}/h_{jj'11}$, $h_{jj'31}/h_{jj'21}$ in 3-module case, $h_{jj'34}/h_{jj'31}$ in 2x2-module case)</td>
<td>$\mathcal{U}(2, 5)$</td>
</tr>
</tbody>
</table>

To consider instances of different sizes, we vary the number of time periods, $|T|$. For each different value of $|T|$, 10 problem instances are randomly generated according to the distributions given in Table 2. Note that to ensure problem feasibility, the random demand generated is less than the total possible production capacity of all production sites. Also, the distance between different location nodes is calculated as a random value between the Euclidean and Manhattan distances, to capture the variability in real road networks. In all cases, the supply chain begins with no modules present anywhere, as well as capital and module transportation costs which are reduced in time periods beyond the first using a net present cost which assumes a 10% discount rate for each time period, such that Assumptions 1-4 presented in Section 4 hold.

To showcase the ability of our framework to handle modularity in both size and functionality, two different cases are considered. First is the “3-module case” where only
one type of module is available, and this module comes in 3 different sizes which can produce 50, 100, and 200 units per time period, respectively. In this case, \(|M| = 3\). Second is the “2x2-module case” where two modules have differing functionalities, both of which are required to produce a product and both of which come in two different sizes which support production of 50 and 100 units per time period, respectively. In this case, \(|M| = 4\). The 3-module case has 9 possible configurations, while the 2x2-module case has 7 possible configurations, as shown in Figure 6. From this figure, it is apparent that the maximum production per time period \(u_{\text{max}}\) is 200 units in the 3-module case and 100 units in the 2x2-module case. Note that in the 2x2 case, the maximum capacity is not 200 units because capacity is not additive among all modules when modules of different functionality are present, and is instead determined by the least available capacity constructed for any individual functionality.

The results of this computational experiment are shown in Figure 7, where each data point indicates the average computation time for instances solved to 0.5% optimality gap within the time limit of \(10^4\) s. For the 2x2 module case, we consider instances with 4, 8, 12, 18, 24, 36, 48, 60, and 72 time periods, while for the 3-module case we consider the same set plus 84 and 96 time periods. Here, it is apparent that the full space approach is slowest in solving the problem for all but those with the fewest number of time periods, where the column generation approach does not provide any benefit. It is expected, however, that the solution time for the column approach scales more favorably with the number of time periods due to the parallel implementation of the
Table 3: Summary statistics for instances not solved to 0.5% optimality gap in $10^4$ s. (NS=Number of instances not solved, NF=Number of instances where no feasible solution found, Gap=average optimality gap in instances where feasible solution found)

| $|\mathcal{M}|$ | $|\mathcal{T}|$ | Full Space | Reformulation | Priority Branching | Column Generation |
|---|---|---|---|---|---|
| NS | NF | Gap (%) | NS | NF | Gap (%) | NS | NF | Gap (%) | NS | NF | Gap (%) |
| 3 | 4 | 1 0 1.10 | 0 0 - | 0 0 - | 1 0 0.51 |
| 8 | 5 | 0 1.64 | 0 0 - | 0 0 - | 0 0 - |
| 12 | 6 | 0 2.13 | 0 0 - | 0 0 - | 0 0 - |
| 18 | 8 | 0 1.47 | 1 0 0.53 | 1 0 0.55 | 0 0 - |
| 24 | 7 | 0 1.47 | 0 0 - | 0 0 - | 1 0 3.85 |
| 36 | 8 | 0 3.28 | 1 0 1.12 | 2 0 0.94 | 0 0 - |
| 48 | 6 | 0 2.82 | 1 1 - | 0 0 - | 1 0 11.26 |
| 60 | 8 | 0 2.56 | 2 2 - | 1 1 - | 0 0 - |
| 72 | 6 | 0 1.11 | 2 2 - | 1 1 - | 0 0 - |
| 84 | 6 | 0 1.66 | 2 2 - | 2 2 - | 0 0 - |
| 96 | 7 | 0 2.65 | 7 7 - | 5 5 - | 0 0 - |
| 4 | 4 | 2 0 0.80 | 0 0 - | 0 0 - | 0 0 - |
| 8 | 8 | 0 2.14 | 2 0 1.00 | 2 0 1.05 | 0 0 - |
| 12 | 9 | 0 1.49 | 0 0 - | 0 0 - | 1 0 0.64 |
| 18 | 8 | 0 3.23 | 2 0 1.40 | 2 0 1.62 | 0 0 - |
| 24 | 9 | 0 3.30 | 1 0 0.76 | 1 0 0.74 | 0 0 - |
| 36 | 8 | 0 3.62 | 2 0 1.01 | 0 0 - | 0 0 - |
| 48 | 9 | 0 3.17 | 0 0 - | 0 0 - | 0 0 - |
| 60 | 9 | 0 5.40 | 4 4 - | 3 0 1.63 | 2 0 .071 |
| 72 | 10 | 0 5.35 | 5 3 2.33 | 4 0 0.88 | 0 0 - |

algorithm. This is seen in both the 3-module and 2x2-module cases, where the solution time for the column generation algorithm remains approximately constant as time periods are added while all other approaches see increases in the computation time. In both cases, column generation is not competitive with other approaches for problems with small numbers of time periods, but becomes the fastest approach on average for problems with more than about 24-36 time periods. In addition, Table 3 gives statistics on the instances that were not solved to a 0.5% optimality gap after $10^4$ s. Here it is apparent that in problems with a greater number of time points, column generation is able to solve the largest number of instances, the full space approach solves the fewest number of instances, and the reformulation and priority branching approaches begin to have trouble finding even a feasible solution.

For problems with small numbers of time periods, the priority branching approach performs best on average. This conclusion does not hold for every instance of the problem, however; for example, in the 3-module case with 8 time periods, the priority branching approach achieves the shortest solution time in 50% of instances while the reformulation approach is fastest in the other 50%. This is likely due to the fact
that in some instances, CPLEX is able to detect the structure of the problem and apply branching rules that are more effective than our specified one. Overall, the difference in computation times between the reformulation and priority branching approaches are not significantly different, and both are good options for fast solutions of problem instances with small numbers of time periods.

7.3. Computational performance: performance curves

To analyze the general performances of the four different solution approaches to solving the DMMFLP, we randomly generate problem instances of different sizes in terms of |I|, |J|, and |T|, which are chosen to be between 2 and 50, 2 and 10, and 2 and 100, respectively. The total problem sizes considered vary over 4 orders of magnitude, ranging from 56-450,000 continuous variables, 28-45,000 binary variables, 36-44,000 nonbinary integer variables, and 148-492,000 constraints. We generate and solve 249 different instances of the 3-module case and 148 different instances in the 2x2-module case.

The results from this computational experiment are presented in the form of performance curves depicted in Figure 8, which show the fraction of all problem instances that each solution approach is able to solve as a function of the computation time budget, as well as a black “best method” curve which corresponds to the fraction of problems that can be solved by any of the four approaches within a given amount of time. A superior solution method will have a performance curve that is further up, indicating that this approach can solve more problems in a smaller amount of time. Figure 8 shows that among the four solution approaches, the column generation approach is most often the superior approach given this criterion for the 3-module case, whereas the column generation, reformulation, and priority branching all perform similarly (after about 100s of
computational budget) for instances from the 2x2-module case. It is also evident that no single solution approach is uniformly superior for every instance of the problem as evidenced by the significant space between the column generation curve and the black curve. In fact, even the full space approach, which is overall the worst performing approach, is able to solve a few problem instances faster than any of the other three approaches. In general, column generation tends to perform best in problems with larger number of time periods, while the reformulation and priority branching approaches perform better for problems with fewer time periods. This is evidenced in Figure 9, which breaks down all instances into the 139 instances where $|T| \leq 36$ and the 258 instances where $|T| > 36$. However, beyond considering the number of time periods, it is difficult to intuitively develop any strict guidelines to suggest which solution approach is best for a specific problem instance.

It is also important to consider the effect of the computational time budget when choosing which solution approach to use. Here, it is apparent that for computational budgets less than 100 s, both the reformulation and priority branching approaches are able to solve the most problems to a 0.5% gap. Beyond 100 s, the column generation approach is able to solve many more problems and is the better approach for large numbers of time periods. After the maximum budget of $10^4$ s, the priority branching approach is able to solve the most problems of any method. Overall, in all but one instance, at least one of the four approaches was able to solve the problem to a 0.5% gap within $10^4$ s. A summary of optimality gap results for problem instances not solved in $10^4$ s is shown in Table 4. Note that column generation is able to quickly (in the first iteration) find a feasible solution in every instance tested; however, there is at least one instance for each of the other methods in which they cannot find even a feasible
Figure 9: Performance curves for solving many different sized instances of the DMMFLP where (a) $|T| \leq 36$ and (b) $|T| > 36$.

Table 4: Summary statistics for instances not solved to 0.5% optimality gap in $10^4$ s

<table>
<thead>
<tr>
<th>Solution Method</th>
<th>Not Solved</th>
<th>No Feasible Found</th>
<th>Average Gap (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Space</td>
<td>145</td>
<td>1</td>
<td>1.65</td>
</tr>
<tr>
<td>Reformulation</td>
<td>14</td>
<td>11</td>
<td>8.8</td>
</tr>
<tr>
<td>Priority Branching</td>
<td>7</td>
<td>1</td>
<td>1.12</td>
</tr>
<tr>
<td>Column Generation</td>
<td>16</td>
<td>0</td>
<td>0.75</td>
</tr>
</tbody>
</table>

solution in $10^4$ s. Indeed, column generation provides the smallest average gap for problem instances not solved in $10^4$ s.

8. Conclusions and future work

In this paper, we introduced the dynamic modular and mobile facility location problem (DMMFLP), which aims to optimize the location and relocation of production modules that can be moved between different production sites, allowing for a supply chain that is more responsive to changing demand. Multiple solution methods were developed for the DMMFLP, including a reformulation of the original MILP formulation which allows for the use of priority branching on selected integer variables and a branch-and-price decomposition algorithm. In a comprehensive computational study, the branch-and-price algorithm exhibited superior computational performance for problem instances with large numbers of time periods, while solving the reformulated problem either directly or using priority branching was superior for instances with fewer time periods. Future work will consider uncertainties, e.g. in the customer demands, using a scenario-based stochastic programming approach. We anticipate that the branch-and-price algorithm will be crucial in solving instances of relevant sizes in
the stochastic case.

We also introduced two metrics, the value of module mobility (VMM) and the demand center of mass, to analyze the benefits from modular and mobile production facilities at the supply chain level. Computational studies show that the benefits are large when customer demands change significantly in both space and time. Moreover, the VMM increases with increasing capital cost, increasing product transportation cost, and decreasing module relocation cost.

Acknowledgments

The authors gratefully acknowledge financial support from the University of Minnesota and the Minnesota Supercomputing Institute (MSI) at the University of Minnesota for providing resources that contributed to the research results reported within this paper.

References


