CIRRELT

CIRRELT-2025-37

Profit Maximizing Network Design with Location Decisions Considering Incompatible Commodities and Demand Selection

Ehsan Mirzaei
Teodor Gabriel Crainic
Walter Rei
Ehsan Nikbakhsh
Ali Husseinzadeh Kashan

November 2025

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

Profit Maximizing Network Design with Location Decisions Considering Incompatible Commodities and Demand Selection

Ehsan Mirzaei[1,3], Teodor Gabriel Crainic[1,2*], Walter Rei[1,2], Ehsan Nikbakhsh[3], Ali Husseinzadeh Kashan[3]

1. Interuniversity Research Centre on Enterprise Networks Logistics and Transportation (CIRRELT)
2. School of Management, Université du Québec à Montréal
3. Faculty of Industrial and Systems Engineering, Tarbiat Modares University

Abstract: This study focuses on a profit maximization variant of the multi-commodity network design problem with location decisions and the flow of incompatible commodities. A novelty of the problem lies in the restrictions imposed on the commodity flows. Specifically, it is assumed (for either safety or security reasons) that each commodity is associated with a specified interdiction list, identifying the other commodities that cannot be shipped in conjunction with it. We thus consider a setting in which demand selection decisions (establishing which commodities to serve) must be made simultaneously with both the network design and flow decisions. The latter decisions aim to establish a network that supports the execution of commodity flows while enforcing the relevant interdictions. The problem, hence, is to select a subset of available nodes for activation, design arcs from a set of potential connections, and determine which demands to serve, to maximize total profit. Additionally, in this research, if commodities are not shipped directly from their origin to destination, there is no restriction on the number of transshipment locations they may visit. To solve this problem, a branch & Benders cut algorithm accelerated with various strategies is developed. An extensive set of computational experiments is conducted on a set of instances from the literature. Results indicate the efficiency of the proposed algorithm compared to a commercial solver for large-sized instances.

Keywords: Multi-commodity network design, Transshipment selection, Incompatible commodities, Demand selection, Branch & Benders cut

Acknowledgements: While working on the project, first author was a visiting researcher at the Université de Montréal, while the second author was Adjunct Professor, Department of Computer Science and Operations Research, Université de Montréal, and the third held the Canada Research Chair in Stochastic Optimization of Transport and Logistics Systems. We gratefully acknowledge the financial support provided by the Natural Sciences and Engineering Council of Canada (NSERC) through its Discovery Grant program, and the Fonds de recherche du Québec through their CIRRELT infrastructure grants.

Results and views expressed in this publication are the sole responsibility of the authors and do not necessarily reflect those of CIRRELT.

Les résultats et opinions contenus dans cette publication ne reflètent pas nécessairement la position du CIRRELT et n'engagent pas sa responsabilité.

---

* Corresponding author: teodorgabriel.crainic@cirrelt.net

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2025

1 Introduction

Network design (ND) problems are integral tools for managing complex systems that involve both a demand to be satisfied and a supply network that must be organized to fulfill that demand. In the multi-commodity network design (MCND) setting, the demand side is typically represented by a set of commodities, each specifying a given quantity to be moved from a point of origin to a point of destination. The supply side is represented by a network to be designed, enabling the efficient movement of these commodities. MCND problems are relevant across all levels of planning, from strategic decisions about network configuration to tactical decisions that guide the operations of the system. The objectives of these problems are flexible and can be determined based on requirements, including minimizing the overall cost of network design or maximizing profit, reliability, and service levels (Crainic and Hewitt, 2020; Crainic et al., 2021).
The complexity of the MCND problems is well established (Gendron et al., 1999); however, our study incorporates additional sources of combinatorial complexity arising from both the demand and supply sides. On the demand side, commodities are not only characterized by their origins and destinations but are also selected subject to explicit incompatibility constraints stemming from technical, regulatory, or safety considerations that prohibit certain commodities from being transported together. On the supply side, in addition to the conventional selection of arcs, our formulation explicitly incorporates the selection of transshipment nodes from a set of predefined, available locations within the network. This selection determines which existing nodes will serve as transshipment points for commodity flows. These interdependent decision layers substantially expand the solution space and increase the overall complexity of the problem. We refer to this integrated setting as the *profit-maximizing multiple incompatible-commodity network design problem with location decisions* (PMICND). The objective is to maximize profit by selecting a subset of commodities that respects incompatibility constraints, designing the network topology (nodes and arcs), and routing compatible flows through the network. Profit accounts for revenues from served commodities minus both fixed setup costs for opened facilities and arcs, and variable routing costs. Each commodity must be shipped either directly from origin to destination or via one or more open transshipment centers, with incompatible flows kept strictly separate.
This problem is motivated by multiple real-world applications. In *telecommunication network design*, companies determine the optimal placement of nodes (e.g., central offices, data centers) and the set of data streams to serve, while incompatibilities may arise from spectrum-sharing restrictions that forbid certain signals from being transmitted together (Voicu et al., 2018). In *transportation and logistics*, network design decisions are strategic and costly, making it essential to account for potential incompatibilities from the outset. These incompatibilities can stem from the nature of the goods, regulatory restrictions, or contractual agreements such as *Dedicated Shipment Agreements*, which require certain commodities to be shipped separately using exclusive resources. For example, fragile items like crystal glassware cannot be transported with heavy goods, and perishables

such as fruits and vegetables often have strict compatibility requirements. Compatibility tables, like those introduced by Lipton and Harvey (1977), help group commodities that can be safely consolidated. Other categories requiring separation include chemicals and food, pharmaceuticals and flammable materials, explosives and everyday goods, agricultural products and pesticides, and medical waste and general cargo.
In this paper, we introduce a new MCND variant that jointly optimizes commodity selection, routing, and transshipment facility selection under explicit incompatibility constraints. We develop a mixed-integer programming formulation for the problem and design a Benders decomposition algorithm with acceleration strategies tailored to the structure of the problem. The decomposition leverages the distinction between complicating variables (design and location decisions) and facilitating variables (flow assignments). We then conduct a computational study to compare decomposition variants and assess the value of explicitly modeling incompatibilities and transshipment selection.
The remainder of this paper is organized as follows. Section 2 provides a concise overview of the literature on different aspects of the current study, including location decisions in network design, profit-maximizing frameworks, and incompatibility of flow in network design. Sections 3 and 4 present the proposed problem setting and mathematical model, respectively. Section 5 describes the developed Benders decomposition algorithm and acceleration strategies. Section 6 discusses the best decomposition approach and provides numerical analyses on the performance of the proposed Benders decomposition algorithm.

2 Related literature

Multi-commodity network design problems involve deciding which nodes and arcs to design, and how to route multiple commodities through the network. Classical formulations assumed all commodities must be transported and aimed to minimize total cost. More recent work has begun to incorporate revenue and profit, allowing carriers to select a subset of demands to serve. By contrast, the treatment of commodity incompatibility has largely been confined to operational problems such as vehicle-routing, and remains scarcely explored in strategic network design. To clarify the state of the art, we organize the literature into three themes: location and network-design decisions, profit-oriented models, and studies that address incompatibility at any level.

2.1 Network design and location decisions

Network design problems may involve choosing nodes (facilities) and arcs to create a network that can route multiple commodities. Hub–location (HLP) and hub–network design (HND) models are among the most studied. In HLP, a subset of nodes is selected as hubs to consolidate flows; HND extends this by selecting inter-hub links. Classic HND

models assume economies of scale on hub–to–hub links and treat origin/destination nodes as fixed (access–level arcs are not designed) (Contreras, 2021). Some studies relax these assumptions by designing all, including the collection, transfer, and distribution arcs, and allowing flows through multiple hubs (Taherkhani and Alumur, 2019; Contreras, 2021). Yoon and Current (2008) showed that hub-network models can be seen as special cases of MCND when hub locations are fixed. Another class of MCND models, often called general network design (GND), do not distinguish hubs from customer nodes: any node may be activated and each arc incurs a fixed charge. GND involves the joint optimization of location, design, and non-trivial routing decisions, often accompanied by allocation choices (Contreras and Fernández, 2012). Variants of MCND consider transshipment network design (TND), and relay network design (RND). TND models integrate the selection of transshipment facilities with routing; early work by Current (1988); Current and Pirkul (1991) and Lien et al. (2011) used heuristics to locate transshipment points and determine flows. RND models decide where to place relay facilities (for driver or vehicle exchanges) in long-haul trucking or multimodal systems (Ali et al., 2002; Yıldız et al., 2018). These models underline that node selection decisions are strategic and interact strongly with subsequent routing.

2.2 Profit-maximizing network design

Traditional network design assumes fixed demand, but many carriers can choose which commodities or orders to serve. Profit-maximizing models incorporate revenue for serving a commodity and often allow the carrier to reject low-profit orders. In service network design (SND), Andersen and Christiansen (2009) considered a railway operator who selects the most profitable freight contracts given uncertain demand. In MCND, Zetina et al. (2019b) introduced a profit-oriented fixed-charge network design with elastic demand; they used a gravity model to relate demand to travel time and incorporated service commitments. Later, Huang (2021) and Bilegan et al. (2022) examined revenue management aspects in intermodal freight networks.

Within HND, profit-maximizing variants have attracted much attention. Alibeyg et al. (2014) formulated mixed-integer programs and Lagrangian relaxations for hub-network design with profits, allowing demand nodes to be assigned to at most two hubs. Taherkhani and Alumur (2019) removed this restriction and allowed flows through any number of hubs. Oliveira et al. (2022) developed a Benders decomposition for a profit-maximizing multiple-allocation HND, while Oliveira et al. (2023) proposed heuristics for a variant with incomplete hub networks. Cobeña et al. (2023) modeled demand elasticity in hub-line location problem using a gravity function and formulated mixed-integer non-linear and linear models, showing that designing a hub-line system under elastic demand is computationally challenging and that column generation can be effective.

Newer studies further integrate pricing and outsourcing decisions. Fernandez et al. (2024) studied a bi-level multi-commodity flow problem with outsourcing decisions. A leader

(carrier) chooses which commodities to serve on a hub–network, sets outsourcing fees for non-hub legs, and allocates carriers. Carriers (followers) accept or reject assignments to maximize their profits. The bi-level problem is reformulated as a single-level program, and the authors propose several MIP formulations (Fernandez et al., 2024). Although these models do not consider incompatibilities, they illustrate how profit-maximizing network design continues to evolve, adding new strategic decisions such as outsourcing fees.

2.3 Commodity incompatibility and conflict

Real networks often carry products that cannot be transported together due to safety, quality, or regulatory reasons. At the operational level, vehicle-routing problems with incompatible commodities model this by preventing incompatible pairs from sharing a vehicle or compartment. For example, Gu et al. (2023) surveyed multi-commodity VRPs and noted that explicit commodity modeling is necessary when aggregation leads to sub-optimal plans. At the network-design level, very few studies consider incompatibility. Şuvak et al. (2020) introduced the maximum flow problem with conflicts, where pairs of arcs cannot carry positive flow simultaneously, and proposed Benders-decomposition and branch-and-cut techniques. More recent studies, such as Montemanni and Smith (2025), consolidate these developments by reviewing exact and heuristic methods for the conflict-constrained maximum flow problem and highlighting applications in telecommunications and transportation. Importantly, this work centers on single-commodity flows; to the best of our knowledge, research on conflicting commodities in multi-commodity network design remains unexplored. By introducing conflict constraints into a profit-maximizing multi-commodity network design framework, our PMICND model addresses this gap. It allows for the strategic selection of commodities, the design of transshipment nodes and arcs, and the enforcement of incompatibility constraints across commodities, thereby extending conflict-constrained flow concepts to a multi-commodity setting and integrating them with profit-oriented network design.

2.4 Summary and positioning

Existing research provides a rich set of frameworks for network design, profit optimization, and managing incompatibilities, but these three streams rarely intersect. On the methodological side, most multi-commodity network design and hub-location models are formulated as mixed-integer programs with fixed-charge costs. A wide range of exact and heuristic algorithms have been proposed, including Lagrangian relaxations (Alibeyg et al., 2014), Benders decomposition (Şuvak et al., 2020; Oliveira et al., 2022) and column-generation procedures (Cobeña et al., 2023), which have proved effective for network design and related problems. However, these methods have been applied

either to profit-maximizing network design models that assume homogeneous commodities and ignore incompatibilities, or to conflict-constrained flow problems that focus on single-commodity flows in fixed networks. As a result, there is still no solution approach that simultaneously tackles node selection, commodity choice, and incompatibility constraints in a multi-commodity setting. The PMICND fills this gap by integrating all three dimensions.

MCND and HND models decide which nodes and arcs to activate, but they typically require all commodities to be transported; profit-maximizing variants introduce commodity selection and pricing but ignore incompatibility constraints; and conflict-constrained flow models handle incompatibilities but assume a fixed network and omit profit considerations. The PMICND problem is particularly challenging and novel because it embeds demand selection directly into the network design process, rather than assuming a fixed set of commodities. Each commodity is characterized by its own origin–destination pair, demand volume, and incompatibility restrictions, which not only determine feasible flows but also fundamentally shape the network topology to be designed. This coupling between the demand side and the supply-side configuration (nodes and arcs) greatly expands the solution space beyond that of traditional network design, requiring simultaneous optimization over both the supply-side configuration, leading to a highly interdependent and combinatorially complex decision problem. The PMICND simultaneously selects transshipment nodes, chooses which commodities to carry based on profitability and incompatibility, and designs the network to route those commodities. Our review shows that no existing model addresses all these dimensions together, underscoring the novelty and importance of the PMICND framework. Therefore, the main contributions of the proposed PMICND model are stated as follows:

1. A profit-maximizing framework is developed to optimize commodity selection while accounting for incompatibilities in a multi-commodity setting where two incompatible commodities cannot share capacity on the arcs. Each selected commodity must be fully satisfied.
2. We expand the scope of MCND problems by considering the selection of nodes as transshipment points. There is a cost associated with selecting each transshipment point, and as a result, a specific capacity becomes available. Commodities can select to use transshipment points or not, or even split the flow between direct and indirect shipment.
3. An exact branch-and-Benders cut algorithm is developed to tackle the proposed problem, ensuring that the complex interactions between node selection, commodity routing, and incompatibility constraints can be managed computationally.

3 Problem setting

Consider $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ to be a directed graph, where $\mathcal{N}$ is the set of nodes and $\mathcal{A} \subseteq \mathcal{N} \times \mathcal{N}$ is the set of potential arcs. The PMICND problem consists of selecting a subset of arcs from the potential arcs $(i, j) \in \mathcal{A}$ and a subset of nodes from the fixed nodes $i \in \mathcal{N}$. For each arc $(i, j) \in \mathcal{A}$, $f_{ij} > 0$ is the fixed design cost whenever the arc is included in the design, giving access to a capacity of u_{ij}. For each node $i \in \mathcal{N}$, there is a fixed activation cost $f_i > 0$ considered if we allow the transition of flow through that node. When node j is activated as a transshipment node, it gives access to a capacity of u_j for transition. A maximum of $T \leq |\mathcal{N}|$ nodes may be selected as transshipment points.
Demands are represented by commodities. These commodities are defined on the nodes of the network. They are shown by $k \in \mathcal{K}$, with each commodity represented by an origin-destination (OD) pair $(O(k), D(k))$ and a volume d^k that represents the demand for commodity $k \in \mathcal{K}$. For every commodity $k \in \mathcal{K}$, the origin and destination are distinct nodes, i.e., $O(k) \neq D(k)$. Each commodity, if selected to be satisfied, comes with a revenue of $r^k d^k$. The unit flow cost for commodity $k \in \mathcal{K}$ on arc $(i, j) \in \mathcal{A}$ is $c_{ij}^k \geq 0$.
We specify that: (a) the origin and destination nodes of all commodities, $O(k)$ and $D(k)$, are always available, incur no activation cost, and do not count toward the T-limit; (b) only pure transshipment nodes are subject to activation and the limit T; (c) a node has no capacity limit unless it is activated as a transshipment node. In that case, it is assigned a maximum capacity u_j, which limits only the transit flows through node j, i.e., flows for which j is neither the origin nor the destination. Flows originating from or terminating at j are not subject to this limit; (d) flow of a commodity k can be split from $O(k)$ to $D(k)$.
The PMICND problem designs both the supply side and the demand side of the network simultaneously. On the demand side, it selects the most profitable commodities, each defined by its own OD pair, volume, and incompatibility restrictions. These incompatibility requirements directly affect feasible flows and, in turn, the supply-side configuration. This tight interdependence between commodity selection and network design expands the network design configurations beyond MCND models, resulting in a highly combinatorially complex decision problem. We define $\Lambda = \{\lambda_k : k \in \mathcal{K}\}$, where each $\lambda_k \subseteq \mathcal{K} \setminus \{k\}$ denotes the set of commodities incompatible with commodity k, meaning they cannot be flowed together. These incompatibility relations are non-transitive; that is, if a commodity k is incompatible with some $l \in \lambda_k$, and l is incompatible with another commodity $m \in \lambda_l$, this does not imply that $m \in \lambda_k$, unless explicitly specified. To avoid redundancy and keep each λ_k concise, we define only non-mutual incompatibility relations.
For each combination of selected demands on the demand side, there needs to be a feasible route that connects the origin and destination of selected commodities. This feasible route is achieved by the supply side of the network. On the supply side, the aim is to design the topological structure of the network given the selected demand with the least possible cost. The topological structure is achieved by designing the required arcs to be present in the network, and by selecting the proper intermediary nodes to be active as a

transshipment point.
Additionally, full-demand fulfillment is assumed. That is, if a commodity is accepted, it must be completely satisfied, and partial fulfillment is not allowed. This encourages the selection of the most profitable set of commodities while respecting incompatibility constraints. As a result, the decision process involves selecting the most profitable feasible combination of commodities, considering compatibility constraints. This leads to binary demand selection decisions determining which requests to accept and which to reject. For each unit of commodity $k \in \mathcal{K}$, a revenue r^k is earned if it is delivered to its designated destination $D(k)$.
The objective is to maximize the total revenue gained from satisfying the demands. This profit consists of: (1) the revenue r^k gained by serving an OD with d^k, (2) the sum of fixed costs regarding the selection of transshipment nodes in the network,(3) the sum of fixed costs regarding selecting the arcs to be included in the network, and (4) the transportation costs associated with moving the flow of all commodities.

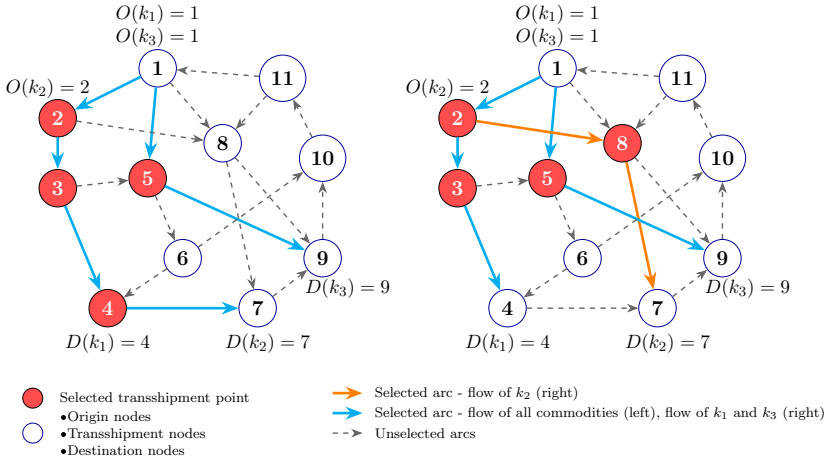


Figure 1: Impact of commodity incompatibility constraints on network flow paths. Prohibiting k_1 and k_2 from sharing arcs alters their routing while no transshipment cost or restriction is considered.

Figures 1 and 2 present an example involving three commodities (k_1, k_2, and k_3) with specified origins and destinations. Figure 1 illustrates the effect of considering commodity incompatibility, while Figure 2 shows the impact of jointly accounting for location decisions and commodity incompatibilities, assuming $f_\ell > f_{ij}$ and $f_\ell = f_l \quad \forall \ell, l \in \mathcal{N},\ \ell \neq l,\ (i,j) \in \mathcal{A}$. First, consider a case where the commodities face specific shipping restrictions and transshipment location decisions are not considered (Figure 1, left). In Figure 1, commodity k_1 cannot be shipped with k_2, and commodity k_3 has no restriction. As a result, k_1 and k_2 can not share capacity on the same arc and must follow separate paths, though they may still share the same node. Consequently, the paths for k_1 and k_2 change and become $k_1 : 1 \to 2 \to 3 \to 4$ and $k_2 : 2 \to 8 \to 7$ whereas the path for k_3 remains the same $k_3 : 1 \to 5 \to 9$. Now, in addition to incompatibility, if we consider the fixed

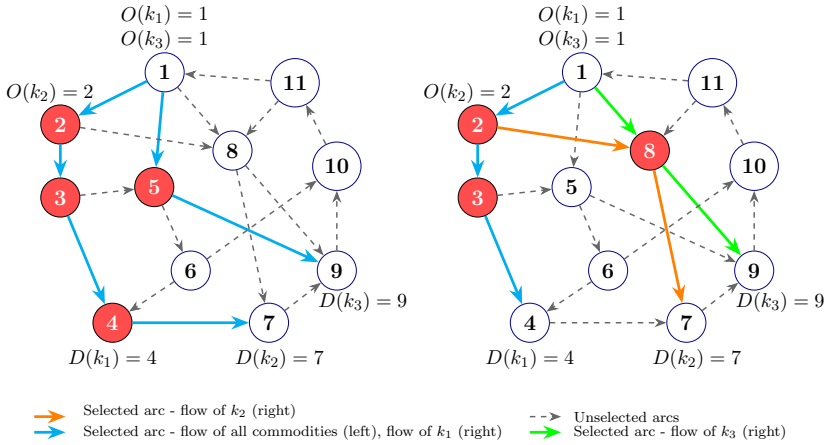


Figure 2: Joint impact of commodity incompatibility constraints and transshipment node selection. Simultaneously enforcing incompatibility restrictions and selecting cost-effective transshipment nodes modifies both network topology and flow patterns.

costs associated with the selection of transshipment points, the routes will be affected, as shown in Figure 2. It is no longer profitable to use four transshipment points (2,3,5,8), and therefore, both commodities k_2 and k_3 will share the same transshipment point (8), resulting in a network with a higher profit.
It should be noted that in some cases, shipping certain commodities may not be practical due to high number of incompatibility requirements and limited available capacity. As a result, maintaining network flow hinges on balancing the revenue from serving each commodity against the costs of designing the network.

4 Mathematical formulation

This section introduces the proposed formulation for the PMICND problem. The sets and parameters used in the model are outlined in Table (1).

The proposed PMICND model involves five decision variables, designing both the supply and demand side of the network. On the demand side, the demand selection decisions are modeled using the binary variable w^k, which takes the value one if commodity $k \in \mathcal{K}$ is selected to be fully satisfied; and zero otherwise, indicating that the demand is entirely rejected. On the supply side of the network, design and flow decisions will need to be decided. Given the demand selection, network topology is determined by the binary variable y_{ij}, indicating whether arc $(i, j) \in \mathcal{A}$ is included in the selected network. Additionally, the binary variable z_j denotes whether node $j \in \mathcal{N}$ is activated as a transshipment point. The continuous variable x_{ij}^k represents the flow of commodity $k \in \mathcal{K}$ on arc $(i, j) \in \mathcal{A}$. To enforce the incompatibility requirements specified by Λ,

Table 1: The nomenclature of the PMICND model

Symbol	**Description**
Sets	
$\mathcal{N}$	Set of all nodes \transshipment nodes $i, j \in \mathcal{N}$.
$\mathcal{A}$	Set of all arcs $(i, j) \in \mathcal{A}$.
$\mathcal{K}$	Set of all commodities $k \in \mathcal{K}$.
$\mathcal{A}_i^+ = \{j \in \mathcal{N} : (i, j) \in \mathcal{A}\}$	Set of outgoing arcs from node $i \in \mathcal{N}$.
$\mathcal{A}_i^- = \{j \in \mathcal{N} : (j, i) \in \mathcal{A}\}$	Set of incoming arcs to node $i \in \mathcal{N}$.
$\lambda_k \in \Lambda$	Set of incompatible commodities with commodity $k \in \mathcal{K}$.
Λ	Set of all incompatible sets: $\Lambda = \{\lambda_k : k \in \mathcal{K}\}$.
Parameters	
f_i	Fixed cost for selecting node $i \in \mathcal{N}$ as a transshipment node
f_{ij}	Fixed cost for designing a potential arc $(i, j) \in \mathcal{A}$
u_{ij}	Capacity of arc $(i, j) \in \mathcal{A}$.
c_{ij}^k	Variable cost of shipping one unit of commodity k on arc $(i, j) \in \mathcal{A}$
r^k	Revenue gained from serving one unit of commodity $k \in \mathcal{K}$.
d^k	Amount of demand for commodity $k \in \mathcal{K}$.
u_i	Capacity of transshipment node $i \in \mathcal{N}$.
T	Maximum number of transshipment points $T \leq \lvert\mathcal{N}\rvert$.
Decision variables	
w^k	A binary demand selection variable, equals one if commodity $k \in \mathcal{K}$ is selected to be served, 0 otherwise.
y_{ij}	A binary variable equals one if arc $(i, j) \in \mathcal{A}$ is selected, 0 otherwise.
z_j	A binary variable equals one if node $j \in \mathcal{N}$ is considered a transshipment point, 0 otherwise.
x_{ij}^k	A continuous variable representing the quantity of flow of commodity $k \in \mathcal{K}$ on arc $(i, j) \in \mathcal{A}$.
v_{ij}^k	A binary variable, indicating whether or not a part of commodity $k \in \mathcal{K}$ is sent via arc $(i, j) \in \mathcal{A}$.

the binary variable v_{ij}^k is introduced to indicate whether any portion of commodity k is transmitted via arc (i, j); it takes the value one when the arc is used for routing commodity k, and zero otherwise.

PMICND model

$$\max\ Z = \sum_{k\in\mathcal{K}} d^k r^k w^k - \sum_{(i,j)\in\mathcal{A}} \sum_{k\in\mathcal{K}} c_{ij}^k x_{ij}^k - \sum_{j\in\mathcal{N}} f_j z_j - \sum_{(i,j)\in\mathcal{A}} f_{ij} y_{ij} \quad (1)$$

$$s.t. \quad \sum_{j\in\mathcal{A}_i^+} x_{ij}^k - \sum_{j\in\mathcal{A}_i^-} x_{ji}^k = \begin{cases} d^k w^k, & i = O(k) \\ -d^k w^k, & i = D(k) \\ 0, & i \neq O(k), i \neq D(k) \end{cases} \quad \forall k \in \mathcal{K}, i \in \mathcal{N} \quad (2)$$

$$\sum_{k\in\mathcal{K}} x_{ij}^k \leq u_{ij} y_{ij} \quad \forall (i,j) \in \mathcal{A} \quad (3)$$

$$x_{ij}^k \leq u_{ij} v_{ij}^k \quad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K} \quad (4)$$

$$\sum_{k'\in\lambda_k} v_{ij}^{k'} \leq |\lambda_k|(1 - v_{ij}^k) \quad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K}, \lambda_k \in \Lambda \quad (5)$$

$$\sum_{\substack{k\in\mathcal{K}: \\ j\notin\{O(k),D(k)\}}} \sum_{i\in\mathcal{A}_j^-} x_{ij}^k \leq u_j z_j \quad \forall j \in \mathcal{N} \quad (6)$$

$$\sum_{j\in\mathcal{N}} z_j \leq T \quad (7)$$

$$z_j \in \{0,1\} \quad \forall j \in \mathcal{N} \quad (8)$$

$$y_{ij}, v_{ij}^k, w^k \in \{0,1\}, \quad x_{ij}^k \in \mathbb{R}_{\geq 0} \quad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K} \quad (9)$$

The objective function (1) maximizes the profit, consisting of four terms. The first term represents the total revenue gained from serving the selected demands. The second term computes the total flow cost, and the last two terms represent the fixed cost of arc and location selection, respectively. Equation (2) represents the flow conservation constraints, which are defined according to the selected commodities. Equation (3) presents the capacity linking constraint. This equation guarantees that the flow on each arc will not exceed the capacity of that arc. Equations (4) and (5) satisfy the incompatibility restrictions. The constraint (4) is a linking constraint that limits the flow to the capacity of the links. Constraint (5) ensures that if a commodity is selected, any incompatible commodities will not be assigned to the same capacity. Constraint (6) prohibits flow from a transshipment point unless it is opened. This equation indicates that a node can serve as a transshipment point for a commodity as long as it is neither the origin nor the destination of that commodity. They also help achieve the assumption of traversing as many transshipment nodes as needed before arriving at the destination. Equation (7) imposes a limit on the number of transshipment nodes in the network to at most T; together with (6), these constraints limit the number of hops in any path to a maximum of $T + 1$. Finally, equations (8) and (9) represent the domain of the decision variables.

5 Benders Decomposition Approach

In ND problems with incompatible flows, two network designs may appear identical while their associated flows differ significantly, posing a unique challenge. Obtaining not only the best network design but also the optimal flow for incompatible commodities is computationally demanding, especially for large instances where closing the optimality gap is difficult. This underscores the need for an efficient and exact algorithm capable of delivering high-quality solutions and closing the gap even on large-scale problems.
Benders decomposition algorithm (BD) is a classical decomposition technique that partitions the original problem into two interdependent components: a master problem (MP) and a sub-problem (SP) (Benders, 1962). The MP guides the solution process by proposing candidate solutions, while the SP evaluates these solutions and generates feedback to iteratively improve the MP's formulation. We apply the BD algorithm as an exact solution method for the proposed PMICND model. A critical question in the PMICND context is determining the appropriate partitioning of decisions between the MP and SP. Specifically, one must decide what information should remain in the MP to ensure effective guidance of the search, and what should be deferred to the SP to efficiently generate informative feedback (Section 5.1).
In the BD algorithm, the original model is projected onto a subspace, which is determined by the MP decision variables. Once the projection is made, it is converted into a dual form. As a result, an equivalent model is obtained that encompasses all the extreme points and rays of the dual polyhedron. The extreme rays represent feasibility conditions, while the extreme points indicate the projected cost associated with feasible projections (Rahmaniani et al., 2017). These are incorporated into the MP through feasibility and optimality cuts, respectively. The iterative process continues until a predefined convergence criterion is met. Although BD was originally developed for linear programming problems with continuous SP, it has been extended to SP with integer program settings as well (Laporte and Louveaux, 1993). When the SP includes integer variables, standard duality theory does not apply, and classical Benders cuts cannot be derived. In such cases, alternative theoretical tools or modified cut-generation techniques must be employed to preserve the algorithm's validity (Fontaine et al., 2021). Consequently, the overall framework of the decomposition depends on both the modeling structure and the continuous or discrete space of the SP.
The remainder of this section is as follows: we first discuss different approaches for decomposing the PMICND (Section 5.1), then we describe the overall algorithmic structure of our proposed Benders decomposition algorithm (Section 5.2), and finally, we present the specialized enhancements proposed to accelerate the proposed Benders decomposition algorithm (Section 5.3).

5.1 Decomposition strategies

In BD, the MP thus guides the search process, while the SP evaluates the resulting solutions and reacts to the MP's decisions. Therefore, a central consideration is how the complexity of the PMICND is distributed between the two problems. Assigning more variables to the MP can improve guidance by reducing information loss from relaxation, but at the cost of a harder MP to solve. However, restricting the SP to continuous variables simplifies cut generation via duality, yet risks producing an overly complex MP. Hence, an effective decomposition requires a careful partition of decision variables to balance computational burden, ensuring both strong guidance in the MP and tractable cut generation in the SP. In ND, topological design decisions are generally considered complicated due to their combinatorial impact (Crainic and Gendron, 2020). Since these decisions define the network's structural backbone and guide the routing process, they are typically assigned to the MP, while the flow variables are delegated to the SP. The combinatorics of the PMICND is compounded as it addresses both the supply-side decisions (i.e., y_{ij} and z_j) and the demand-side decisions (i.e., w^k and v_{ij}^k). It is natural to include the supply-side variables in the MP, as its solution directs the search process. However, it is the selection of commodities that ultimately drives the need for network design; no commodities can be distributed without first designing a feasible network. As a result, the general question posed here is: which decisions should be included in the MP to best guide the overall search process, while also supporting a computationally efficient solution approach? Therefore we propose four decomposition strategies to answer this question.

(DS_1): $y_{ij}, z_l, w^k, v_{ij}^k \rightarrow$	in MP	$x_{ij}^k \rightarrow$	in SP
(DS_2): $y_{ij}, z_l, w^k \rightarrow$	in MP	$x_{ij}^k, v_{ij}^k \rightarrow$	in SP
(DS_3): $y_{ij}, z_l \rightarrow$	in MP	$x_{ij}^k, v_{ij}^k, w^k \rightarrow$	in SP
(DS_4): $y_{ij}, w^k \rightarrow$	in MP	$x_{ij}^k, v_{ij}^k, z_l \rightarrow$	in SP

The DS_1 decomposition strategy follows the classical Benders approach by which discrete design variables are assigned to the MP (treated as complicating variables), while continuous variables are sent to the SP, enabling application of continuous duality. Decomposition (DS_2) is structured such that supply-side decisions are retained in the MP, while on the demand-side, only the selection of commodities is decided in the MP. This is justified by the fact that selected demands implicitly determine the structural requirements of the supply-side. Strategy (DS_3) decomposes the problem based on the supply-side and demand-side decisions. In this strategy the supply-side decisions are decided in MP, and demand-side decisions in SP. This approach is justified based on the facility location and network design models where location and design are the main components of structuring the network, therefore deciding on which arcs to route on and which transshipment locations to select are kept with the MP, whereas (DS_4) assumes that two strategic decisions of both the supply-side and demand-side (i.e., y_{ij} and w^k) are decided in MP. The last two strategies argue whether the transshipment decision should be decided in the MP or projected by the SP.

Given these decomposition frameworks, we adopt generalized Benders decomposition (GBD) when working with continuous SP, which yields generalized optimality cuts (GOC) and generalized feasibility cuts (GFC). When integrality constraints are retained in the SP (DS_2, DS_3, DS_4), the dual variables are not available. In such cases, we use the integer Benders decomposition (IBD) procedure, which generates integer optimality cuts (IOC) and integer feasibility cuts (IFC).
The decomposition and cut-generation approach in DS_1 differs from the other three strategies, while DS_2 aligns closely with DS_3 and DS_4. For clarity and brevity in presenting the acceleration strategies, we provide a detailed discussion of DS_1 and DS_2, with the remaining strategies deferred to Appendix A. In what follows, when referring to the MP solutions from one of the four decomposition strategies, the following notation will be used: $\bar{y}_{ij},\ \bar{z}_j,\ \bar{w}_k,$ and $\bar{v}^k_{ij}$, as applicable to the MP of each specific DS.

5.1.1 Strategy DS_1

Having all the complicating binary variables in the MP will result in the MP formulated as (5),(7),(8),(10)-(14), and the SP being a linear program formulated as (15)-(23). We use the generalized Benders decomposition (Geoffrion, 1972), as a result, a generalized optimality cut (12) is added to the MP for each $\gamma \in \Gamma_{DS_1}$ where Γ_{DS_1} is the set of all integer feasible solutions for which we have $Z^*_{SP_{DS_1}} < \overline{\theta}_{DS_1}$ in the DS_1 strategy, where $\overline{\theta}_{DS_1}$ is the approximation of the SP_{DS_1} objective function. On the other hand, a feasibility cut (13) is added to the MP for each $\omega \in \Omega_{DS_1}$ where Ω_{DS_1} is the set of all infeasible solutions found in the B&B tree in DS_1 strategy. The MP for DS_1 is formulated as follows:

$$\max\ Z_{MP_{DS_1}} = \sum_{k\in\mathcal{K}} r^k d^k w^k - \sum_{j\in\mathcal{N}} f_j z_j - \sum_{(i,j)\in\mathcal{A}} f_{ij} y_{ij} + \theta \tag{10}$$

$$s.t. \quad (5), (7), (8) \tag{11}$$

$$Optimality\ Cuts \quad \forall \gamma \in \Gamma_{DS_1} \tag{12}$$

$$Feasibility\ Cuts \quad \forall \omega \in \Omega_{DS_1} \tag{13}$$

$$y_{ij}, w^k, v^k_{ij} \in \{0,1\} \quad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K} \tag{14}$$

The SP for DS_1 is formulated as follows:

$$\max\ Z_{SP_{DS_1}} = -\sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}} c_{ij}^k x_{ij}^k \tag{15}$$

$$s.t.\quad (2),(3),(4),(5),(6) \tag{16}$$

$$y_{ij} = \overline{y}_{ij} \quad \forall (i,j)\in\mathcal{A} \tag{17}$$

$$w_k = \overline{w}_k \quad \forall k\in\mathcal{K} \tag{18}$$

$$z_j = \overline{z}_j \quad \forall j\in\mathcal{N} \tag{19}$$

$$v_{ij}^k = \overline{v}_j \quad \forall j\in\mathcal{N} \tag{20}$$

$$x_{ij}^k \in \mathbb{R}_{\geq 0} \quad \forall (i,j)\in\mathcal{A}, k\in\mathcal{K} \tag{21}$$

$$z_j \in [0,1] \quad \forall j\in\mathcal{N} \tag{22}$$

$$y_{ij}, w^k, v_{ij}^k \in [0,1] \quad \forall (i,j)\in\mathcal{A}, k\in\mathcal{K} \tag{23}$$

Let y_{ij}, w^k, v_{ij}^k, and z_j denote the copy-variables in subproblem SP_{DS_1}, and let $\hat{y}_{ij}$, $\hat{v}_{ij}^k$, $\hat{w}_k$, and $\hat{z}_j$ be their corresponding optimal values obtained by solving SP_{DS_1}. Constraints (17)-(20) are added as a result of GBD, and $\beta_{i,j}^1, \beta_k^2, \beta_j^3, and \beta_{ijk}^4$ represent their dual multipliers respectively. Therefore, the optimality cut for DS_1 is as follows:

$$\begin{aligned}\theta \leq &-\sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}} c_{ij}^k \overline{x}_{ij}^k + \sum_{(i,j)\in\mathcal{A}} \beta_{i,j}^1 (y_{i,j} - \hat{y}_{i,j}) + \sum_{k\in\mathcal{K}} \beta_k^2 (w_k - \hat{w}_k) \\ &+ \sum_{j\in\mathcal{N}} \beta_j^3 (z_j - \hat{z}_j) + \sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}} \beta_{ijk}^4 (v_{ij}^k - \hat{v}_{ij}^k)\end{aligned} \tag{24}$$

To generate the feasibility cut, we solve the following Feasibility Sub-Problem (FSP). FSP minimizes the amount of infeasibility caused by the choices made by the MP. Here, ϵ denotes a decision variable that captures the amount of deviation from feasibility to be minimized, and $\hat{\epsilon}$ represents its value. Let y_{ij}, w^k, v_{ij}^k, and z_j denote the copy-variables in subproblem FSP_{DS_1}, and let $\hat{y}_{ij}$, $\hat{v}_{ij}^k$, $\hat{w}_k$, and $\hat{z}_j$ be their corresponding optimal values

obtained by solving the following FSP_{DS_1}.

$$\min\ Z_{FSP_{DS_1}} = \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K}\\ i\notin O(k)}} (\epsilon^1_{ik} + \epsilon^2_{ik}) + \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K}\\ i\notin D(k)}} (\epsilon^3_{ik} + \epsilon^4_{ik}) + \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K}\\ i\notin O(k)\\ i\notin D(k)}} (\epsilon^5_{ik} + \epsilon^6_{ik})$$
$$+ \sum_{(i,j)\in\mathcal{A}} \epsilon^7_{ij} + \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}} (\epsilon^8_{ijk} + \epsilon^9_{ijk}) + \sum_{j\in\mathcal{N}} \epsilon^{10}_j \tag{25}$$

$$s.t. \quad \sum_{j\in\mathcal{A}^+_i} x^k_{ij} - \sum_{j\in\mathcal{A}^-_i} x^k_{ji} = \begin{cases} d^k w^k + \epsilon^1_{ik} - \epsilon^2_{ik}, & i = O(k) \\ -d^k w^k + \epsilon^3_{ik} - \epsilon^4_{ik}, & i = D(k) \\ 0 + \epsilon^5_{ik} - \epsilon^6_{ik}, & i \neq O(k), i \neq D(k) \end{cases} \quad \forall k \in \mathcal{K}, i \in \mathcal{N} \tag{26}$$

$$\sum_{k\in\mathcal{K}} x^k_{ij} \leq u_{ij} y_{ij} + \epsilon^7_{ij} \quad \forall (i,j) \in \mathcal{A} \tag{27}$$

$$x^k_{ij} \leq u_{ij} v^k_{ij} + \epsilon^8_{ijk} \quad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K} \tag{28}$$

$$\sum_{k'\in\lambda_k} v^{k'}_{ij} \leq |\lambda_k|(1 - v^k_{ij}) + \epsilon^9_{ijk} \quad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K}, \lambda_k \in \Lambda \tag{29}$$

$$\sum_{\substack{k\in\mathcal{K}:\\ \{O(k)\neq j, D(k)\neq j\}}} \sum_{\substack{i\in\mathcal{N}:\\ (i,j)\in\mathcal{A}}} x^k_{ij} \leq u_j z_j + \epsilon^{10}_j \quad \forall j \in \mathcal{N} \tag{30}$$

$$y_{ij} = \overline{y}_{ij} \quad \forall (i,j) \in \mathcal{A} \tag{31}$$

$$w_k = \overline{w}_k \quad \forall k \in \mathcal{K} \tag{32}$$

$$z_j = \overline{z}_j \quad \forall j \in \mathcal{N} \tag{33}$$

$$v_j = \overline{v}_j \quad \forall j \in \mathcal{N} \tag{34}$$

$$\epsilon^1_{ik}, \epsilon^2_{ik}, \epsilon^3_{ik}, \epsilon^4_{ik}, \epsilon^5_{ik}, \epsilon^6_{ik}, \epsilon^{10}_i \in \mathbb{R}_{\geq 0} \quad \forall k \in \mathcal{K}, i \in \mathcal{N} \tag{35}$$

$$\epsilon^7_{ij}, \epsilon^8_{ijk}, \epsilon^9_{ijk} \in \mathbb{R}_{\geq 0} \quad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K} \tag{36}$$

Consider $\hat{\beta}^1_{i,j}$, $\hat{\beta}^2_k$, $\hat{\beta}^3_j$ and $\hat{\beta}^4_{ijk}$ to be the dual multipliers of constraints (31)-(34) respectively. We can then generate the feasibility cut as follows:

$$0 \geq \sum_{i\in\mathcal{N}} \left(\sum_{\substack{k\in\mathcal{K}\\ i\neq O(k)}} (\hat{\epsilon}^1_{ik} + \hat{\epsilon}^2_{ik}) + \sum_{\substack{k\in\mathcal{K}\\ i\neq D(k)}} (\hat{\epsilon}^3_{ik} + \hat{\epsilon}^4_{ik}) + \sum_{\substack{k\in\mathcal{K}\\ i\neq O(k)\\ i\neq D(k)}} (\hat{\epsilon}^5_{ik} + \hat{\epsilon}^6_{ik}) \right) + \sum_{(i,j)\in\mathcal{A}} \hat{\epsilon}^7_{ij} + \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}} (\hat{\epsilon}^8_{ijk} + \hat{\epsilon}^9_{ijk})$$
$$+ \sum_{j\in\mathcal{N}} \hat{\epsilon}^{10}_j + \sum_{(i,j)\in\mathcal{A}} \hat{\beta}^1_{ij}(y_{ij} - \hat{y}_{ij}) + \sum_{k\in\mathcal{K}} \hat{\beta}^2_k(w_k - \hat{w}_k) + \sum_{j\in\mathcal{N}} \hat{\beta}^3_j(z_j - \hat{z}_j) + \sum_{(i,j)\in\mathcal{A}} \sum_{k\in\mathcal{K}} \hat{\beta}^4_{ijk}(v^k_{ij} - \hat{v}^k_{ij}) \tag{37}$$

5.1.2 Strategy DS_2

In this strategy, since v_{ij}^k carries no fixed cost, the MP is similar to the DS_1 formulation, omitting v_{ij}^k from the decision variables and retaining only constraints (7), (8). Since the SP's feasible space is discrete, an integer optimality cut (40) is added to the MP for each $\gamma \in \Gamma_{DS_2}$ where Γ_{DS_2} is the set of all integer feasible solutions for which we have $Z^*_{SP_{DS_2}} < \overline{\theta}_{DS_2}$ in the DS_2 strategy, where $\overline{\theta}_{DS_2}$ is the approximation of the SP_{D_2} objective function. On the other hand, a combinatorial cut (41) is added to the MP for each $\omega \in \Omega_{DS_2}$ where Ω_{DS_2} is the set of all infeasible solutions found in the B&B tree, in DS_2 strategy. The MP for DS_2 is formulated as follows:

$$\max\ Z_{MP_{DS_2}} = \sum_{k\in\mathcal{K}} r^k d^k w^k - \sum_{j\in\mathcal{N}} f_j z_j - \sum_{(i,j)\in\mathcal{A}} f_{ij} y_{ij} + \theta \tag{38}$$

$$s.t. \quad (7), (8) \tag{39}$$

$$Optimality\ Cuts \quad \forall \gamma \in \Gamma_{DS_2} \tag{40}$$

$$Feasibility\ Cuts \quad \forall \omega \in \Omega_{DS_2} \tag{41}$$

$$y_{ij}, w^k \in \{0,1\} \quad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K} \tag{42}$$

The SP for DS_2 is formulated as follows:

$$\max\ Z_{SP_{DS_2}} = -\sum_{(i,j)\in\mathcal{A}} \sum_{k\in\mathcal{K}} c_{ij}^k x_{ij}^k \tag{43}$$

$$s.t. \quad \sum_{j\in\mathcal{A}_i^+} x_{ij}^k - \sum_{j\in\mathcal{A}_i^-} x_{ji}^k = \begin{cases} d^k \overline{w}^k, & \forall k \in \mathcal{K}, i \in \mathcal{N}, i = O(k) \\ -d^k \overline{w}^k, & \forall k \in \mathcal{K}, i \in \mathcal{N}, i = D(k) \\ 0, & \forall k \in \mathcal{K}, i \in \mathcal{N}, i \neq O(k), i \neq D(k) \end{cases} \tag{44}$$

$$\sum_{k\in\mathcal{K}} x_{ij}^k \leq u_{ij} \overline{y}_{ij} \quad \forall (i,j) \in \mathcal{A} \tag{45}$$

$$x_{ij}^k \leq u_{ij} v_{ij}^k \quad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K} \tag{46}$$

$$\sum_{k'\in\lambda_k} v_{ij}^{k'} \leq |\lambda_k|(1 - v_{ij}^k) \quad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K}, \lambda_k \in \Lambda \tag{47}$$

$$\sum_{\substack{k\in\mathcal{K}: \\ \{O(k)\neq j, D(k)\neq j\}}} \sum_{\substack{i\in\mathcal{N}: \\ (i,j)\in\mathcal{A}}} x_{ij}^k \leq u_j \overline{z}_j \quad \forall j \in \mathcal{N} \tag{48}$$

$$v_{ij}^k \in \{0,1\},\ x_{ij}^k \in \mathbb{R}_{\geq 0} \quad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K} \tag{49}$$

The following optimality cut is added to the MP_{DS_2} once an integer feasible solution is found.

$$\theta \leq (Z_{\mathrm{SP}_{DS_2}} - UB_{\mathrm{SP}_{DS_2}})\Big(\sum_{(i,j)\in\eta_y} y_{ij} - \sum_{(i,j)\notin\eta_y} y_{ij} + \sum_{k\in\eta_w} w_k - \sum_{k\notin\eta_w} w_k + \sum_{j\in\eta_z} z_j - \sum_{j\notin\eta_z} z_j - (|\eta_y| + |\eta_w| + |\eta_z|)\Big) + Z_{\mathrm{SP}_{DS_2}} \tag{50}$$

In this context, $UB_{\mathrm{SP}_{DS_2}}$ denotes an upper bound on $Z_{\mathrm{SP}_{DS_2}}$. Furthermore, $\eta_y = \{(i,j) \in \mathcal{A} : y_{ij} = 1\}$ is the set of arcs where the variable y_{ij} takes the value 1 for that integer feasible solution. The sets η_w and η_z are defined similarly. In cases where SP_{DS_2} results in an infeasible solution, we add the following combinatorial cut to exclude the current infeasible solution from further investigation.

$$\sum_{(i,j)\in\eta_y} (1 - y_{i,j}) + \sum_{(i,j)\notin\eta_y} y_{i,j} + \sum_{k\in\eta_w} (1 - w_k) + \sum_{k\notin\eta_w} w_k + \sum_{j\in\eta_z} (1 - z_j) + \sum_{j\notin\eta_z} z_j \geq 1 \quad (51)$$

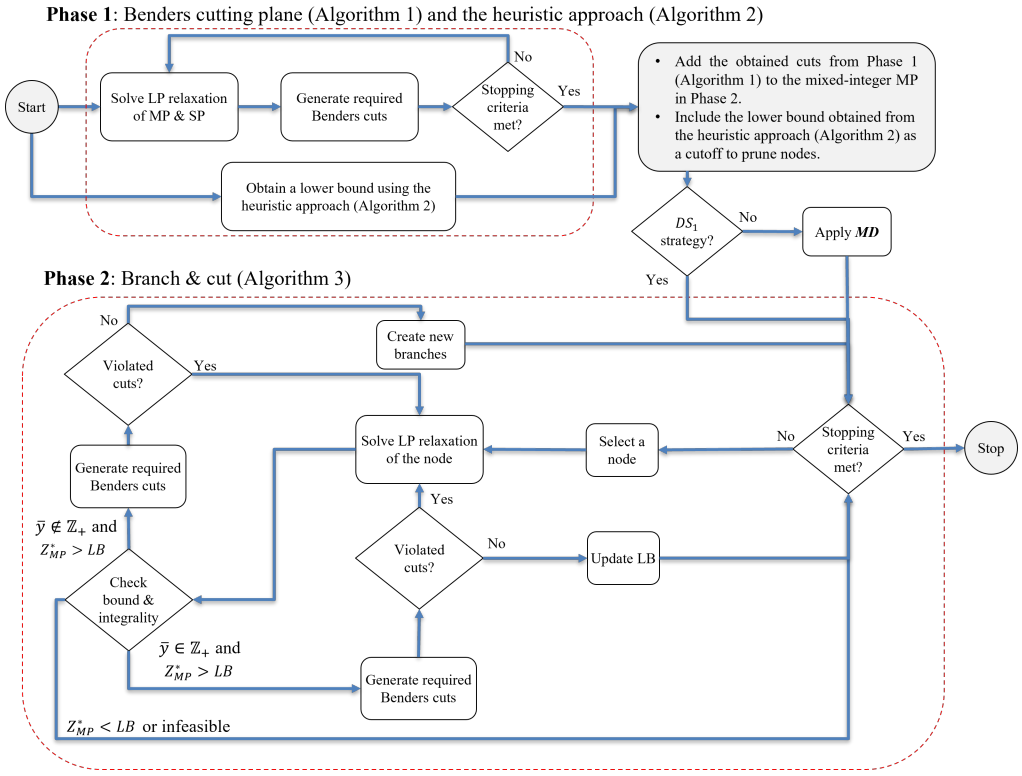
5.2 Algorithmic structure

In the original BD approach, the mixed-integer MP is solved to optimality during every iteration, in a cutting-plane fashion, a process that can be very demanding in terms of computational resources. Newer methods, however, tackle the MP only once and then integrate optimality cuts dynamically as the branch-and-bound (B&B) tree is explored. This technique, referred to as branch-and-Benders-cut (B&BC) or the Benders-based branch-and-cut algorithm, capitalizes on callback functions available in commercial solvers like Gurobi and CPLEX. We use the B&BC as it has shown to be efficient (Bodur and Luedtke, 2017). Figure (3) illustrates the proposed B&BC. The detailed pseudo-codes for the B&BC algorithm are provided in the appendix. In the first phase (Algorithm 1), we relax the integrality requirements of MP and solve the root node using a cutting-plane approach as discussed by (McDaniel and Devine, 1977). These cuts help tighten the linear relaxation in the subsequent phase. Additionally, we obtain a lower bound using a greedy heuristic approach (Algorithm 2) and add it as a cutoff to the subsequent phase. In the second phase (Algorithm 3), the integrality conditions are restored and the problem is solved in a branch-and-cut (B&C) fashion. This two-phase strategy enables Gurobi to inject additional or more robust cuts into the model, leveraging the enriched information from the initial relaxations. Whenever a cut is violated at a node, we add that cut to the MP using `lazycut`. When no further cuts can be added to a node, the algorithm moves to the next unexplored node. This process continues until the stopping criteria are met.

5.3 Accelerating Benders decomposition algorithm

Various acceleration and improvement approaches have been introduced for the Benders decomposition applied to network design problems (Costa, 2005; Rahmaniani et al., 2018; Zetina et al., 2019a). This section provides the selected methods to speed up the Benders decomposition algorithm for the PMICND problem. These methods include: (1) Modified (non-standard) decomposition (Gendron et al., 2016). (2) Using the Benders dual cuts to accelerate the convergence (Rahmaniani et al., 2020). (3) Strengthening the generalized

Figure 3: Flow chart of the proposed branch & Benders cut algorithm (B&BC).



Note. Z^*_{MP}, LB and $\overline{y}$ are the optimal objective value of the LPR, the best lower-bound, and an optimal solution to the problem, respectively

optimality cut (Papadakos, 2008). (4) Incorporating valid inequalities to strengthen the MP and SP formulations. (5) Warm starting the B&BC (McDaniel and Devine, 1977). (6) Initializing the B&B tree with a lower bound using a greedy heuristic approach.

5.3.1 Modified decomposition

Following the modified decomposition (MD) approach, also known as the nonstandard decomposition strategy (Rahmaniani et al., 2017), we strengthen the MP by explicitly incorporating a copy of the SP's linear relaxation directly within the MP formulation. This technique has been shown to significantly accelerate the Benders decomposition algorithm (Gendron et al., 2016), and we therefore adopt it to enhance the efficiency of our algorithm.

In practice, this is accomplished by duplicating the relevant decision variables and constraints from the SP and adding them as valid inequalities to the MP. This strategy has the advantage of partially restoring the linking constraints (between the complicating and facilitating decision variables, as defined by the variable partition) that are otherwise

omitted through the BD process, thereby mitigating the relaxation of the information associated with the SP that results from the decomposition. Moreover, this approach is grounded in the general idea of transferring information between the MP and the SP (in this case, from the latter to the former) as a means to enhance the Benders algorithm; see Hewitt and Rei (2024) for a general illustration of this concept in the context of stochastic programs. Additionally, the following valid inequality is included in the MP:

$$\theta \leq Z_{SP} \tag{52}$$

where Z_{SP} denotes the objective function of the SP, expressed in terms of the duplicated decision variables. It should be noted that this valid inequality complements the Benders optimality cuts, as both provide upper bounds on the value of the projected term in the MP's objective function (i.e., θ). It is important to note that this approach does not apply to DS_1.

5.3.2 Benders dual cuts

Benders dual cuts were first introduced by Rahmaniani et al. (2020). This approach proceeds by constructing a new MILP model for the SP, where a local copy of the MP variables is reintroduced into the new MILP sub-problem to project those in the MP with equality constraints. We call this reformulated problem the *sub-problem with projections* (SPP). This reformulation of the SP has also been used in previous studies to generate generalized Benders cuts (Geoffrion, 1972). Next, a Lagrangian duality is applied to the SPP to price out the equality constraints that link the local copies to the MP variables. We refer to this relaxed sub-problem as the *Lagrangian dual sub-problem* (SPL). By doing this relaxation, the local variables are no longer forced to equal the MP variables. The key advantage of this approach is that, by working with the Lagrangian dual SP, it becomes possible to lift the Benders cuts. This lifting can be achieved either by improving the SP value at the current Lagrangian multipliers through exploration of the original model's full decision space, or by directly exploring the space of Lagrangian multipliers through the Lagrangian dual itself. Rahmaniani et al. (2020) showed that for fractional solutions obtained from the MP, the optimality cuts generated from SPL dominate those generated from SPP. Moreover, for integer solutions of the MP, the cuts derived from SPL are as strong as those from SPP. The same logic applies to feasibility cuts: the *feasibility sub-problem* (SPF) is constructed to generate feasibility cuts, and its strengthened version, the *Lagrangian dual feasibility sub-problem* (SPFL), drives feasibility cuts via its Lagrangian duality.

We present the Benders dual cut generation approach for DS_2, and the same logic extends to the other decomposition strategies. It is important to note that since in DS_1, $SPP_{DS_1} = SP_{DS_1}$ and $SPF_{DS_1} = FSP_{DS_1}$, only SPL and SPFL need to be constructed. However, for DS_2, DS_3, and DS_4, all four sub-problems, namely SPP, SPL, SPF, and SPFL, are required.

Strategy DS_2:

We first build SPP to derive the generalized optimality cuts. SPP_{DS_2} is formulated as follows:

$$\text{Max} \quad Z_{SPP_{DS_2}} = -\sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}} c_{ij}^k x_{ij}^k \tag{53}$$

$$s.t.\ (2)-(7),\ (17)-(19),\ (21)-(23)$$

Assume that, $\hat{y}_{i,j}, \hat{w}_k$, $\hat{z}_j$ and $\hat{x}_{ij}^k$ are the values obtained from solving SPP_{DS_2} and also $\beta_{i,j}^1, \beta_k^2$, and β_j^3 are dual multipliers of constraints (17)-(19) respectively. Therefore, the generalized optimality cut for DS_2 is as follows:

$$\theta \leq -\sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}} c_{ij}^k \hat{x}_{ij}^k + \sum_{(i,j)\in\mathcal{A}} \beta_{i,j}^1 (y_{i,j} - \hat{y}_{i,j}) + \sum_{k\in\mathcal{K}} \beta_k^2 (w_k - \hat{w}_k) + \sum_{j\in\mathcal{N}} \beta_j^3 (z_j - \hat{z}_j) \tag{54}$$

After solving SPP_{DS_2} we can then solve the following SPL_{DS_2} to generate the strengthened optimality cut (SOC) as follows (56).

$$\begin{aligned}\text{Max} \quad Z_{SPL_{DS_2}} = &-\sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}} c_{ij}^k x_{ij}^k - \sum_{(i,j)\in\mathcal{A}} \beta_{i,j}^1 (y_{i,j} - \overline{y}_{i,j}) \\ &- \sum_{k\in\mathcal{K}} \beta_k^2 (w_k - \overline{w}_k) - \sum_{j\in\mathcal{N}} \beta_j^3 (z_j - \overline{z}_j)\end{aligned} \tag{55}$$

$$s.t. (2)-(9)$$

Assume that, $\tilde{y}_{i,j}, \tilde{w}_k$, $\tilde{z}_j$ and $\tilde{x}_{ij}^k$ are the values obtained from solving SPL_{DS_2}. Therefore, the SOC for DS_2 is as follows:

$$\theta \leq -\sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}} c_{ij}^k \tilde{x}_{ij}^k + \sum_{(i,j)\in\mathcal{A}} \beta_{i,j}^1 (y_{i,j} - \tilde{y}_{i,j}) + \sum_{k\in\mathcal{K}} \beta_k^2 (w_k - \tilde{w}_k) + \sum_{j\in\mathcal{N}} \beta_j^3 (z_j - \tilde{z}_j) \tag{56}$$

Similar to the optimality cuts, generalized and strengthened feasibility cuts can also be derived for DS_2, as presented in Appendix A.1. Consequently, each DS requires solving two subproblems to generate strengthened optimality cuts (SPP and SPL) and two subproblems to generate strengthened feasibility cuts (SPF and SPFL).

5.3.3 Pareto optimal cuts

Degeneracy in Benders decomposition occurs when the SP admits multiple optimal dual solutions, requiring the selection of the *deepest* optimality cut that no other cut can dominate. This is crucial for highly degenerate problems such as ND, where neglecting this step can degrade performance (Crainic et al., 2021). Magnanti and Wong (1981) proposed selecting dual variables for the deepest cut by solving an auxiliary SP (i.e. *Pareto SP* (PSP)) alongside the SP, thereby accelerating convergence. Choosing an appropriate core point enables the generation of Pareto-optimal cut (POC). Papadakos (2008) further demonstrated that this approach is independent of the SP's optimal value

and that any convex combination of distinct core points remains valid for producing POCs. In this study, we make use of the duals resulting in the deepest cut to generate the SOC. Instead of deriving the duality information from SPP, it is derived from PSP to generate stronger cuts. The core point is defined as $Cy_{ij}, Cw_k, Cz_j, Cv_{ij}^k$. For DS_2, we have:

$$\begin{cases} Cy_{ij} = (1-\zeta)Cy_{ij} + \zeta\overline{y}_{ij} \\ Cw_k = (1-\zeta)Cw_k + \zeta\overline{w}_k \\ Cz_j = (1-\zeta)Cz_j + \zeta\overline{z}_j \end{cases} \tag{57}$$

where Cy_{ij} and $\overline{y}_{ij}$ denote the core point and the current optimal MP solution, respectively, and $\zeta \in (0,1)$, typically set to 0.5. In PMICND, the presence of incompatible commodities and transshipment points complicates arbitrary core initialization, as it may render the PSP infeasible. To address this, initial core values are assigned 0.5 with respect to the decomposition approach and updated using (57). Since commodity selections may not always satisfy incompatibility constraints, Cw_k is obtained from SPN to ensure PSP feasibility.

5.3.4 Valid inequalities

We derive a series of valid inequalities to strengthen the MP and improve the lower bound, thereby enhancing the upper bound provided by the Benders decomposition algorithm. One of the reasons for the slow convergence of the BD algorithm is the lack of information regarding the network design in the MP. Thereby we introduce different sets of valid inequalities based on the required information for MP.

- **Network design based valid inequalities**

Given that the problem under investigation retains a fundamental MCND structure, we leverage inequalities originally proposed for the MCND to tighten the LPR and thereby accelerate the BD algorithm. Two families of valid inequalities are incorporated in our setting: (i) strong inequalities (SI) and (ii) cut-set inequalities (CI) (Rahmaniani et al., 2018; Crainic and Gendron, 2020). While the SI can be directly added without modification, the CI must be adapted to reflect the commodity-selection feature of our model. The first family, SI (58), directly links the activation of an arc to the flow that can be routed through it.

$$x_{ij}^k \leq b_{ij}^k y_{ij} \qquad \forall (i,j) \in \mathcal{A},\ k \in \mathcal{K} \tag{58}$$

Where $b_{ij}^k = \min\{d^k, u_{ij}\}$. This equation is added to the SP to improve the quality of its LPR and also the duals passed on to the MP, and as a result of MD, it is also added to the MP.
Additionally, the classical CI is adopted to account for the commodity selection feature of our model. For these selected commodities, sufficient capacity must be ensured across all

network partitions to support their flows. This cut accounts for both connectivity and arc capacities between different OD pairs, conditional on the selection of their corresponding demands. A cut is a division of the set $\mathcal{N}$ into two subsets ℓ and $\overline{\ell}$, where $\overline{\ell}$ is the complement of ℓ in $\mathcal{N}$, such that there is a positive net supply across ℓ. In other words, there exists at least one destination that is not in ℓ. A cut-set (ℓ,$\overline{\ell}$) includes the subset of arcs induced by the cut $(\ell, \overline{\ell}) = \{(i,j) \in \mathcal{A} : i \in \ell,\, j \in \overline{\ell}\}$.

$$\sum_{(i,j)\in(\ell,\overline{\ell})} u_{ij} y_{ij} \geq \sum_{k\in\mathcal{K}(\ell,\overline{\ell})} d^k w^k \qquad \forall\, \ell \subseteq \mathcal{N},\ \overline{\ell} \neq \emptyset \tag{59}$$

Where $\mathcal{K}(\ell, \overline{\ell}) = \{k \in \mathcal{K} : O(k) \in \ell, D(k) \in \overline{\ell}\}$. According to (Crainic and Gendron, 2020), adding these cuts to the MCND does not guarantee feasibility. However, these inequalities help reduce the non-optimal and infeasible regions, improving the lower bound in a reasonable time. The same applies to the PMICND problem. The challenge in using CI is dealing with the exponential number of cuts that need to be added to the MP at each node. We use a similar approach to Chouman et al. (2017); Rahmaniani et al. (2018) and generate cuts with the cardinality of ℓ being 1 and 2.

- **Problem definition based valid inequalities**

Equations (60) and (61) represent the linking constraints between defining the demand-side and designing the supply-side. In the PMICND mode, this link was indirectly established using the flow decisions between the corresponding binary variables w^k and y_{ij}. Inclusion of these inequalities explicitly enforces direct connections between the corresponding binary variables. Equation (60) ensures that once a commodity is selected to be satisfied, at least one outgoing arc from the origin node must be opened. Equation (61) ensures that once a commodity is selected to be satisfied, at least one incoming arc toward the destination node must be opened.

$$w^k \leq \sum_{(i,j)\in\mathcal{A}|i=O(k)} y_{ij} \qquad \forall k \in \mathcal{K} \tag{60}$$

$$w^k \leq \sum_{(i,j)\in\mathcal{A}|j=D(k)} y_{ij} \qquad \forall k \in \mathcal{K} \tag{61}$$

Equations (62) and (63) establish the connection between transshipment-node selection and arc-opening decisions. In the PMICND model, this dependency was previously implied through the flow variables linking z_j and y_{ij}. Constraint (62) guarantees that whenever a candidate node is chosen as a transshipment point (and it is not the destination of commodity k), it must be reachable through at least one incoming arc. Conversely, constraint (63) ensures that when such a node is selected (and it is not the origin of

commodity k), there must be at least one outgoing arc available.

$$z_j \leq \sum_{i \in \mathcal{A}_j^-} y_{ij} \qquad \forall j \in \mathcal{N},\ k \in \mathcal{K} : j \neq D(k) \tag{62}$$

$$z_i \leq \sum_{j \in \mathcal{A}_i^+} y_{ij} \qquad \forall i \in \mathcal{N},\ k \in \mathcal{K} : i \neq O(k) \tag{63}$$

Considering the connection between v_{ij}^k with y_{ij}, we know that if a commodity k is selected to be sent through an arc $(i,j) \in \mathcal{A}$, then its corresponding arc and commodity should be selected. Since in DS_1, $v_{ij}^k \in \{0,1\}$ and in DS_2, DS_3, and DS_4, $v_{ij}^k \in [0,1]$, therefore, we have the following valid inequalities:

$$v_{ij}^k \leq y_{ij} \qquad \forall (i,j) \in \mathcal{A},\ k \in \mathcal{K} \tag{64}$$

$$v_{ij}^k \leq w^k \qquad \forall (i,j) \in \mathcal{A},\ k \in \mathcal{K} \tag{65}$$

As a result of incompatibilities and also the limited capacities on the arcs and nodes, choosing or neglecting a transshipment point is quite tricky. Therefore, it is best to know whether a transshipment will be used or not as soon as the demand is selected to be satisfied. Based on (6), whenever a commodity is selected for shipment, its flow can either (a) go directly from its origin to destination, (b) pass through at least one transshipment point, or (c) split between a direct route and at least one transshipment point. Hence, we can express:

$$w^k \leq \sum_{\substack{j \in \mathcal{N}: \\ j \neq O(k), j \neq D(k)}} z_j \ + \ y_{(O(k),D(k))} \qquad \forall k \in \mathcal{K} \tag{66}$$

In addition, we observe that considering the transshipment inequality (6) for the outgoing arcs, as illustrated in (67), as well as tightening the right-hand side (RHS) of the equations by replacing u_i with $\rho_i = \min\{u_i, \sum_{j \in \mathcal{A}_i^+} u_{ij}\} \quad \forall i \in \mathcal{N} : (i,j) \in \mathcal{A}$, significantly accelerates the algorithm.

$$\sum_{\substack{k \in \mathcal{K}: \\ O(k) \neq i,\, D(k) \neq i}} \sum_{j \in \mathcal{A}_i^+} x_{ij}^k \leq \rho_i z_i \qquad \forall i \in \mathcal{N} \tag{67}$$

Likewise for the incoming arcs we replace u_j with $\rho_j = \min\{u_j, \sum_{i \in \mathcal{A}_j^-} u_{ij}\} \quad \forall j \in \mathcal{N} : (i,j) \in \mathcal{A}$. Similarly, we tighten the RHS for (4) by replacing u_{ij} with b_{ij}^k, where $b_{ij}^k = \min\{d^k, u_{ij}\}$. These modifications

- **Maximal incompatible clique cuts**

A key challenge of the PMICND model lies in handling the large number of incompatible-commodity decisions across both the MP and SP. To strengthen constraint (5), the pairwise relations among incompatible commodities are introduced as

$$v_{ij}^{k'} \leq 1 - v_{ij}^k \qquad \forall (i,j) \in \mathcal{A},\ k \in \mathcal{K},\ \lambda_k \in \Lambda,\ k' \in \lambda_k \tag{68}$$

In DS_1, where $v_{ij}^k \in \{0,1\}$, these constraints improve the lower bound, while in the remaining decompositions they strengthen the upper bound. To further reinforce the formulation, *clique cuts* are employed. When more than two commodities are mutually incompatible, the number of pairwise relations in (68) increases rapidly; clique cuts consolidate these relations by enforcing only one active commodity per incompatible group, thereby strengthening the model and reducing computational effort (Atamtürk et al., 2000).
To derive the maximal clique cuts, we construct an undirected *conflict graph* $\mathcal{C} = (\mathcal{V}, \mathcal{E})$, where each vertex represents a commodity and an edge connects two commodities that cannot share the same arc. To ensure symmetry, the incompatibility relations are defined mutually as $\tilde{\lambda}_k \in \tilde{\Lambda}$. The incompatibility structure of $\mathcal{C}$ may contain several fully connected subgraphs, known as *cliques*. A clique is *maximal* if it is not a proper subset of any larger clique.

Proposition 1. *Let $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_l\}$ denote the set of maximal cliques identified in $\mathcal{C}$. For each arc $(i,j) \in \mathcal{A}$ and each maximal clique $\mathcal{C}_l \in \mathcal{C}$ of the conflict graph $\mathcal{C}$, the following inequality is valid for the PMICND model:*

$$\sum_{k \in \mathcal{C}_l} v_{ij}^k \leq 1 \qquad \forall (i,j) \in \mathcal{A},\ \mathcal{C}_l \in \mathcal{C} \tag{69}$$

Proof. Proof. See Appendix A.2. □

This inequality is added to the SP to improve the quality of the duals passed to the MP, and due to master–dual (MD) interactions, it is also included in the MP. In DS_1, where $v_{ij}^k \in \{0,1\}$, it strengthens the lower bound, while in DS_2, DS_3, and DS_4, where $v_{ij}^k \in [0,1]$, it tightens the linear programming relaxation (LPR).

- **Upper bounding function**

For any OD pair, the minimum flow cost can be determined by meeting the corresponding demand along its shortest path. Let $P_{O(k)D(k)}$ denote the shortest path from $O(k)$ to $D(k)$. Because the shortest path minimizes the routing cost for each commodity, the objective value of the SP is at least as large as the cost of this cheapest route. An important advantage of this upper-bounding function (UBF) is that it provides valuable information for the demand selection variables. Let $C^k_{P(O(k),D(k))}$ denote the flow cost of the shortest path for commodity k from its origin to its destination. Based on this, for each strategy, we derive the following cut, which can be added to the MP at each node.

$$\boldsymbol{DS_1 \& DS_2}: \quad \theta \leq -\sum_{k \in \mathcal{K}} C^k_{P(O(k),D(k))} d^k w^k \tag{70}$$

Although (52) is present in MP, UBF improves the upper bound. These UBFs are utilized in two ways. 1) They are added on the fly at each node after IOC & GOC. 2) They improve the strength of the IOC (i.e., (50)) at each integer node by deriving the information of UB_{SP} from UBF. The UBF for DS_3 and DS_4 are presented in A.3.

5.3.5 Warm-up strategy

The warm-start strategy (WS) presented is based on the original work of (McDaniel and Devine, 1977). We first solve the BD approach in a cutting plane manner using the relaxed version of the MP (& SP for DS_2, DS_3, and DS_4), and after the algorithm has converged, we add the integrality back to the MP. The cuts added in this phase (phase (I) are added as a warm start to the B&BC algorithm (phase (II)). Phase I uses the GBD approach, where the optimality and feasibility cuts are derived from SPP and SPF, respectively. In the warm-start, variables and constraints from MD are added to the MP only after completing Phase I. Adding MD to the MP creates a tight LPR, causing Phase I to often converge in a single iteration, effectively bypassing Phase I altogether. However, since including these cuts at the root node significantly improves convergence, MD is incorporated only after Phase I is completed. The warm-start phase is also restricted to a maximum of max_{iter} iterations, with early termination if no improvement is observed in α_0 consecutive iterations. Before adding the cuts to Phase II, we remove the top $\alpha_1\%$ of cuts with high slack values. The pseudocode for this warm-start procedure is shown in Algorithm 1.

5.3.6 Lower bounding with a compatible solution

A key challenge in the PMICND model lies in determining feasible itineraries for commodities while satisfying both capacity limitations and incompatibility restrictions. In particular, a commodity k may need to be split across multiple arcs to meet its incompatibility constraints, which, in turn, complicates the itineraries of all commodities in its incompatibility set λ_k. Consequently, deciding whether to include or exclude a commodity with a large $|\lambda_k|$ is critical, as its route may become excessively long or complex, rendering it inefficient to satisfy. Therefore, initializing the B&B tree with an integer feasible solution helps to fathom these nodes early and avoid unnecessary exploration.
We propose a two-phase greedy heuristic to tackle the PMICND. In the first phase, each commodity k that has no incompatibilities is routed along its shortest, capacity-respecting path. Should this path satisfy residual capacity requirements and yield a strictly positive profit, the flow is committed by opening any necessary arcs or transshipment points, updating residual capacities, and recording the selected itinerary. In the second phase, we address commodities that exhibit incompatibility relations. For each such commodity k, we generate up to three alternative itineraries (specifically, the

three shortest feasible paths ($e \in \mathcal{P}$) under the current network state). These candidate paths are evaluated according to two criteria: net profit $r^k d^k - \sum_{e \in \mathcal{P}} c_e\, d^k$ and number of incompatibility conflicts $|\lambda_k|$. Commodities are then ordered in a priority queue by descending profit and ascending conflict count. Iteratively, we extract the highest-priority commodity k, recompute its candidate paths to reflect any updates in opened infrastructure, and allocate the flow along the first path that simultaneously respects both capacity and incompatibility constraints. Upon completion of both phases, the heuristic returns a set of routed flows that adhere to all capacity and incompatibility requirements, together with the total profit of the resulting solution. The pseudo-code for this heuristic approach is presented in Algorithm 2.

6 Numerical results and analysis

In this section, various numerical tests are conducted to investigate the efficiency of the proposed B&BC algorithm. All algorithms are implemented in Python 3.12 and solved using Gurobi optimizer version 11. To ensure a fair comparison, Gurobi was restricted to a single thread. The experiments are conducted on a Linux server running at 2.67 GHz with 32 GB RAM. All variants of the B&BC are benchmarked against the PMICND model, which the Gurobi solver directly solves.

6.1 Data and experimental settings

Following the literature, a well-known test instance, R and C are used. These instances are commonly used for service and multi-commodity network design problems. The R instances are denoted as $rx.y$, where x represents the network size in terms of nodes, arcs, and commodities, and y represents the specifications related to the fixed and variable cost of the network design and the capacity of the arcs. Specifically we have used 135 R instances with x ranging from 04-18, and y from 1-9. For the C instances we have used all 31 of them. Besides the incompatibility set Λ, the rest of the parameters are presented in Table (2).
To generate the set of incompatibilities for each commodity λ_k, we assume that each commodity falls into one of the following categories: (I) no incompatibilities ($|\lambda_k| = 0$), (II) one or two incompatibilities, or (III) three or four incompatibilities. We use a zero-inflated distribution to populate the set $\lambda_k \in \Lambda$ accordingly. Let μ_1 denote the fraction of commodities with any incompatibilities, and let μ_2 denote the fraction of those commodities that have one or two incompatibilities. As a result, (I) $\lfloor (1-\mu_1)|K| \rfloor$ commodities have no incompatibility requirements, (II) $\lfloor \mu_1\mu_2|K| \rfloor$ have one or two incompatibilities, and (III) $\lfloor \mu_1(1-\mu_2)|K| \rfloor$ have three or four incompatibilities.

Table 2: Values of the parameters

Description	Parameter	Value
Flow cost	c_{ij}^k	R& C Instance Data Set
Revenue for satisfying one unit of commodity	r_k	U$\sim(\lfloor 10 \times \min_{(i,j)\in\mathcal{A}}\{c_{ij}^k\}\rfloor, \lfloor 20 \times \max_{(i,j)\in\mathcal{A}}\{c_{ij}^k\}\rfloor)$
Capacity of arcs	u_{ij}	R& C Instance Data Set
Capacity of transshipment nodes	u_j	$U \sim \left(m(j)\min\{u_{ij} : i \in \mathcal{A}_j^+ \cup \mathcal{A}_j^-\},\ m(j)\max\{u_{ij} : i \in \mathcal{A}_j^+ \cup \mathcal{A}_j^-\}\right)$ $m(j) = \max\{\lvert\mathcal{A}_j^+\rvert, \lvert\mathcal{A}_j^-\rvert\}$
Arc design cost	f_{ij}	R& C Instance Data Set
Transshipment location cost	f_j	U$\sim(2 \times \min_{j\in\mathcal{N}\mid(i,j)\in\mathcal{A}}\{f_{ij}\}, 2 \times \max_{j\in\mathcal{N}\mid(i,j)\in\mathcal{A}}\{f_{ij}\})$
Demand	d^k	R& C Instance Data Set

6.2 Accounting for transshipment points and incompatibility in the design phase

When commodities are mutually incompatible, ignoring demand-side incompatibility requirements during supply-side design can lead to rejected demands due to flow constraints, reducing overall profit. To evaluate the impact of incorporating these requirements, we analyze two cases. In case P_1, incompatibility constraints are specified on the demand side but disregarded by the supply-side planner. In case P_2, both incompatibility constraints and transshipment restrictions (i.e., $\sum_{j\in\mathcal{N}} f_j z_j$, (6), and (7)) are ignored, reducing the problem to the basic profit-maximizing MCND. We compare these cases against the proposed PMICND model while keeping supply-side design fixed. We consider nine parameter settings: $\mu_1 \in \{0.2, 0.4, 0.8\}$ and $\mu_2 \in \{0.3, 0.5, 0.7\}$. Two transshipment scenarios are analyzed: (i) $T = |\mathcal{N}|$, where all nodes can act as transshipment points (Figures 4a, 4b), and (ii) $T = 0.3|\mathcal{N}|$, where only a subset of nodes serve as transshipment points (Figures 4c, 4d). Figures 4a and 4c illustrate case P_1, comparing the effects of considering versus neglecting incompatibility in the design phase for full and limited transshipment networks, respectively. For these analysis we use the r12 instance. Results indicate that selecting transshipment points from a limited set can hedge against the effects of ignoring commodity incompatibility, ensuring profit stability, particularly for networks with larger $|\mathcal{K}|$. Figures 4b and 4d show case P_2, where ignoring both incompatibility and transshipment requirements leads to even higher profit losses, particularly for networks with smaller $|\mathcal{K}|$, where neglecting these constraints can result in negative profits.

6.3 Evaluating decomposition approaches

This section presents a comparative analysis of the decomposition strategies. In this section, we consider $\mu_1 = 0.4$, $\mu_2 = 0.7$, and $T = 0.5|\mathcal{N}|$ for the analysis. The R instance data sets are used for this comparison. Table 3 summarizes the three primary performance metrics. Time (s) denotes the average solution time required by each solution approach. The optimality gap is reported as Gap $= \frac{\text{UB}-\text{LB}}{\text{UB}}$ where UB and LB are the best upper

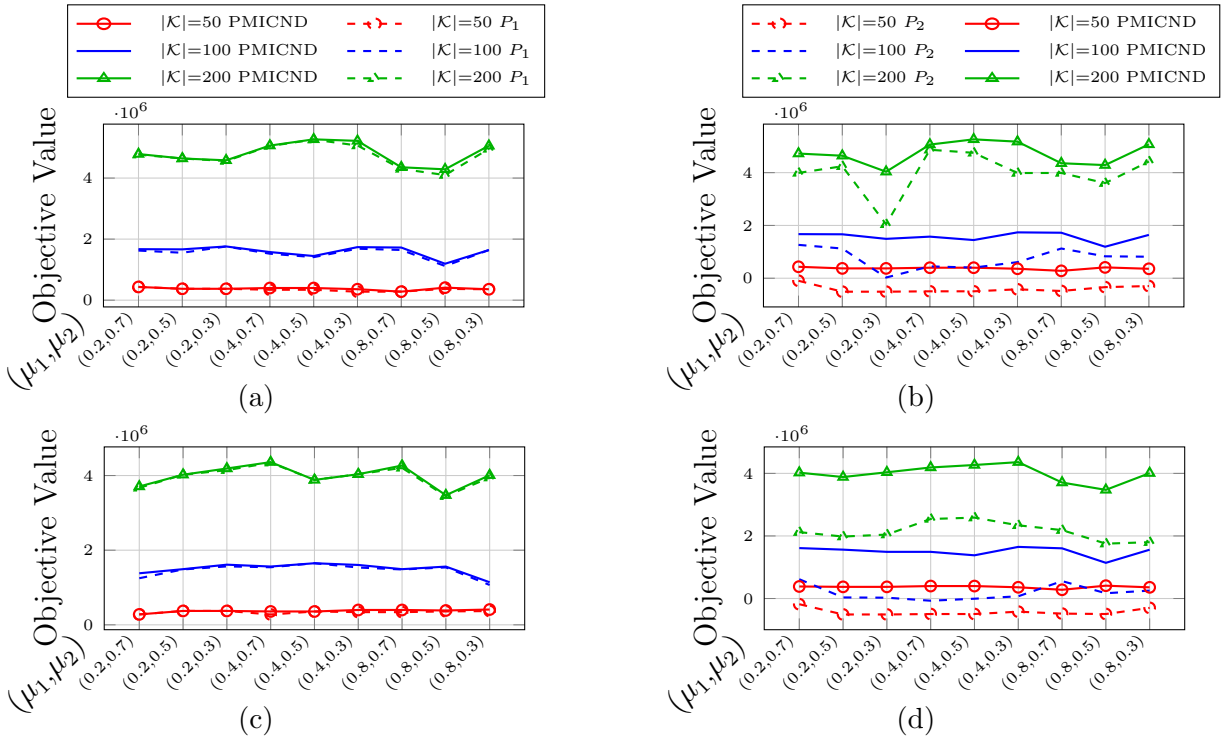


Figure 4: Comparing total profit in P_1 ((4a),(4c)) and P_2 ((4b),(4d)) vs $PMICND$ for different $|\mathcal{K}|$ and (μ_1, μ_2).

and lower bounds obtained. Finally, the dominance ratio D_{ratio} measures the fraction of instances on which B&BC performs at least as well as (or clearly outperforms) Gurobi. To formalize "clear dominance," we assign each instance to one of three categories: D_B for instances where B&BC clearly dominates Gurobi (in time or gap), D_G for instances where Gurobi clearly dominates B&BC, and D_N for instances with no clear dominance. An instance is classified as D_N if any of the following conditions hold: (i) both B&BC and Gurobi converge within 120 s (for the 7200 s limit) or 600 s (for the 18000 s limit); (ii) both solvers converge beyond these thresholds (120 s and 600 s) but their convergence times differ by less than 120 s; or (iii) neither solver converges within the time limit and the difference in their optimality gaps is less than 0.02%. The dominance ratio is then defined as $D_{\text{ratio}} = \frac{D_N+D_B}{D_N+D_B+D_G}$ reflecting the proportion of instances where B&BC is not outperformed by Gurobi. As evidenced by the results in Table 3, strategy DS_2 outperforms all other decomposition approaches with respect to solution time, optimality gap, and D_{ratio}. Therefore we consider this specific type of decomposition strategy for our analysis on the proposed B&BC algorithm.

Table 3: Analyzing decomposition strategies

R	Gurobi 11		DS_1			DS_2			DS_3			DS_4		
	Time(s)	Gap	Time(s)	Gap	D_{ratio}	Time(s)	Gap	D_{ratio}	Time(s)	Gap	D_{ratio}	Time(s)	Gap	D_{ratio}
r04	0.3	0.0%	6020.7	46.3%	0/9	0.8	0.0%	9/9	1671.9	0.1%	6/9	7200	14.7%	0/9
r05	3.1	0.0%	7200	100%	0/9	13.6	0.0%	9/9	2402.9	0.3%	6/9	7200	35.8%	0/9
r06	56.1	0.0%	7200	100%	0/9	35.6	0.0%	9/9	1740.6	1.2%	5/9	7200	28.6%	0/9
r07	1.1	0.0%	7200	82.3%	0/9	801.8	0.0%	8/9	811.5	0.0%	8/9	7200	14.2%	0/9
r08	5.8	0.0%	7200	92.8%	0/9	13.4	0.0%	9/9	864.5	0.3%	6/9	7200	38.6%	0/9
r09	75.1	0.0%	7200	90.3%	0/9	41.6	0.0%	9/9	1725.7	0.2%	5/9	7200	36.7%	0/9
r10	571.9	0.0%	7200	100%	0/9	1222.4	0.0%	6/9	2591.7	3.3%	5/9	7200	69.9%	0/9
r11	3066.4	0.7%	7200	100%	0/9	3744.2	0.4%	5/9	6171.9	3.5%	2/9	7200	91.3%	0/9
r12	4288.9	1.5%	7200	100%	0/9	5389.7	1.2%	4/9	6569.1	8.3%	1/9	7200	96.7%	0/9
r13	4428.5	0.3%	7200	100%	0/9	2911.1	0.1%	7/9	3522.5	1.8%	5/9	7200	67.9%	0/9
r14	5404.4	1.5%	7200	100%	0/9	4279.5	0.8%	6/9	4488.9	1.6%	5/9	7200	69.2%	0/9
r15	7050.4	3.3%	7200	100%	0/9	6617.4	3.1%	6/9	6717.7	5.1%	4/9	7200	77.9%	0/9
r16	4168.5	0.3%	7200	100%	0/9	1401.9	0.0%	8/9	2129.9	0.0%	7/9	7200	67.4%	0/9
r17	7200	1.9%	7200	100%	0/9	4906	0.6%	8/9	5349.6	0.7%	8/9	7200	67.7%	0/9
r18	7200	4.6%	7200	100%	0/9	6855.9	4.1%	7/9	7200	4.4%	6/9	7200	67.9%	0/9
Average	2901.8	0.9%	7121.4	94.1%	0.0%	2549.0	0.7%	81.5%	3597.2	2.1%	58.5%	7200	56.3%	0.0%

6.4 B&BC performance

This section evaluates the performance of the proposed B&BC algorithm, with particular emphasis on the impact of its acceleration strategies and on how problem characteristics influence its efficiency. First, we analyze the contribution of each acceleration strategy to determine their effectiveness in improving computational performance for a fixed $\mu_1 = 0.4$, $\mu_2 = 0.7$, and $T = 0.5|\mathcal{N}|$. Subsequently, we investigate the effect of varying values of T and the magnitude of Λ, on overall algorithmic performance, reflecting how problem complexity scales with these factors.

6.4.1 Evaluating acceleration strategies

To quantify the individual contributions of each acceleration strategy, we perform the analysis by systematically removing each component from B&BC and comparing the resulting performance with the full algorithm. Certain strategies prove critical, as their removal leads to significant performance degradation even on small instances, while the impact of others becomes apparent only when solving large-scale problems. To this end, we exclude MD (Section 5.3.1) and compare the outcomes with complete B&BC. In addition we also evaluate the performance of including (52) separately. These two prove especially important, even in smaller instances. Table 4 summarizes the results for small- and medium-sized R instances under a 3600-second time limit. Column #Sol indicates the number of instances yielding an integer-feasible solution. As demonstrated, omitting any of MD or (52) significantly reduces the algorithm's ability to find feasible solutions. Unlike MD, and (52), which show a noticeable impact even on small instances, the value

Table 4: Evaluating the effect of MD

R	*B&BC*			*B&BC*-(52)			*B&BC-MD*		
	Time(s)	Gap	*#Sol*	Time(s)	Gap	*#Sol*	Time(s)	Gap	*#Sol*
r04	0.8	0.0%	9/9	3213.3	12.4%	9/9	3600	100%	0/9
r05	65.9	0.0%	9/9	3600	17.4%	8/9	3600	100%	0/9
r06	866.5	0.1%	9/9	3600	8.1%	9/9	3600	100%	0/9
r07	401.4	0.0%	9/9	3600	57.4%	4/9	3600	100%	0/9
r08	15.9	0.0%	9/9	3600	24.9%	7/9	3600	100%	0/9
r09	485.2	0.0%	9/9	3600	8.4%	9/9	3600	100%	0/9
r10	1239.8	0.2%	9/9	3600	33.3%	7/9	3600	100%	0/9
Average	439.4	0.0%	100%	3488.2	26.5%	79.4%	3600	100%	0.0%

of remaining acceleration strategies become more evident on large-scale instances. We begin by examining the two key components of the B&BC algorithm: the warm start and the greedy heuristic. Table 5 presents the results of this analysis, focusing on two variations: (1) *B&BC-GH*, where the greedy heuristic is removed, preventing the use of its initial lower bound in the second phase, and (2) *B&BC-WS*, where the first phase is omitted. As illustrated, dropping either of these two components from the B&BC worsens the overall performance in terms of Time, Gap, and D_B.

Table 5: Evaluating the effect of accelerations *GH* & *WS* on the proposed B&BC

Instance	$\|\mathcal{N}\|$	$\|\mathcal{A}\|$	$\|\mathcal{K}\|$	*B&BC*			*B&BC-WS*			*B&BC-GH*		
				Time(s)	Gap	D_B	Time(s)	Gap	D_B	Time(s)	Gap	D_B
r11	20	120	100	1102.7	0.0%	4/9	6287.3	0.3%	4/9	6584.0	0.2%	4/9
r12	20	120	200	4931.1	0.0%	3/9	9473.5	0.3%	3/9	8747.8	0.2%	3/9
r13	20	220	40	1272.7	0.0%	7/9	5193.1	0.0%	5/9	4593.1	0.0%	5/9
r14	20	220	100	6518.7	0.4%	8/9	6460.9	0.5%	6/9	7265.3	0.5%	5/9
r15	20	220	200	14065.2	0.6%	6/9	15050.5	1.3%	6/9	15150.1	1.1%	6/9
C37-40	20	230	200	18000	1.0%	4/4	17757.8	0.7%	4/4	18000	0.8%	4/4
C45-48	20	300	200	18000	0.8%	3/4	18000	0.8%	3/4	18000	0.7%	3/4
r16	20	314	40	633.0	0.0%	7/9	2573.9	0.0%	6/9	2689.4	0.0%	6/9
r17	20	318	100	7745.5	0.2%	9/9	9027.8	0.4%	8/9	9151.6	0.3%	8/9
r18	20	315	200	16514.1	2.8%	7/9	16775.8	3.5%	6/9	16724.3	2.7%	8/9
C49-52	20	520	100	15794.3	0.3%	4/4	16795.9	0.3%	3/4	15724.1	0.2%	4/4
C53-56	30	520	400	18000	1.1%	1/4	18000	1.0%	2/4	18000	0.9%	3/4
C57-60	30	700	100	15022.5	0.1%	4/4	15067.1	0.0%	4/4	14884.2	0.0%	4/4
C61-64	30	700	400	18000	0.9%	3/4	18000	1.0%	2/4	18000	1.0%	2/4
Average				11114.4	0.58%	72.9%	12461.8	0.74%	64.6%	12394.0	0.62%	67.7%

Since there are different classes of VI, we categorize them based on their definitions and further investigate the effect of each class by considering the following categories. a) VI_1, which consists of (60), (61), b) VI_2, which consists of (62), (63), c) VI_3, which consists of (64), (65), d) VI_4, which consists of (66), (67), e) network design based VIs (ND), f) maximal incompatible clique cuts (MIC), and g) upper bounding function (UBF). The results of this analysis are presented in the Appendix in Table (8). As illustrated, neglecting any of the VIs leads to a drop in the overall performance of the B&BC algorithm. Among these VIs, we consider ones to be more impactful if their absence leads to a higher Gap and a lower D_B, compared to when it is considered. In the event of a tie, we prioritize Gap as the deciding factor. Based on this criterion, for $T = 0.5|\mathcal{N}|$

the relative effectiveness of the VIs can be ranked from strongest to weakest as follows: $MIC > ND > VI_4 > VI_3 > VI_1 > UBF > VI_2$.
We further discuss the effect of adding cuts derived from the linear relaxation of SP in the B&C phase in more detail. As discussed in section 5.3.2 we have excluded the feasibility cuts derived from the linear relaxation of SP since it does not contribute much in acceleration. The effect of excluding these cuts on large-sized instances is presented in Table (6). To precisely evaluate the value of these cuts, we compare three cases: (1) excluding only feasibility cuts, which is the current approach ($B\&BC$), (2) excluding feasibility and optimality cuts ($B\&BC\text{-}OC\&FC$), (3) maintaining both feasibility and optimality cuts ($B\&BC{+}FC$), and the (4) effect of strengthening the SOC cuts by driving the duals from PSP instead of SPP ($B\&BC\text{-}PO$). By excluding the optimality cuts, we exclude the generalized, strengthened, and Pareto-optimal cuts, whereas for the feasibility cuts the generalized and strengthened cuts are excluded. Results indicate that keeping the optimality cuts in the B&BC improves D_B, whereas omitting feasibility cuts from the B&BC results in an overall better performance.

Table 6: Evaluating the effect of cuts derived from the LPR of SP.

Instance	$\|N\|$	$\|A\|$	$\|K\|$	*B&BC*			*B&BC-(OC&FC)*			*B&BC-PO*			*B&BC+FC*		
				Time	Gap	D_B	Time	Gap	D_B	Time	Gap	D_B	Time	Gap	D_B
r11	20	120	100	1102.7	0.0%	4/9	7384.2	0.2%	4/9	6634.0	0.2%	4/9	6536.9	0.2%	4/9
r12	20	120	200	4931.1	0.0%	3/9	10569.9	0.8%	2/9	9332.9	0.5%	2/9	9477.0	0.6%	2/9
r13	20	220	40	1272.7	0.0%	7/9	5073.2	0.0%	5/9	4825.6	0.0%	5/9	4928.4	0.0%	5/9
r14	20	220	100	6518.7	0.4%	8/9	8174.7	0.6%	5/9	7268.0	0.5%	6/9	7287.3	0.5%	6/9
r15	20	220	200	14065.2	0.6%	6/9	15467.4	1.8%	6/9	15094.0	1.6%	6/9	15222.2	1.6%	6/9
C37-40	20	230	200	18000	1.0%	4/4	18000	1.0%	4/4	18000	1.0%	4/4	18000	1.1%	4/4
C45-48	20	300	200	18000	0.8%	3/4	18000	1.0%	1/4	18000	0.6%	3/4	18000	0.8%	3/4
r16	20	314	40	633.0	0.0%	7/9	2980.3	0.0%	6/9	2762.6	0.0%	6/9	2745.5	0.0%	6/9
r17	20	318	100	7745.5	0.2%	9/9	9775.5	0.5%	6/9	9022.3	0.4%	8/9	9279.3	0.5%	7/9
r18	20	315	200	16514.1	2.8%	7/9	16686.6	3.3%	5/9	16953.6	3.3%	5/9	16728.4	3.3%	5/9
C49-52	30	520	100	15794.3	0.3%	4/4	15917.0	0.4%	4/4	14741.8	0.2%	4/4	15968.7	0.3%	4/4
C53-56	30	520	400	18000	1.1%	1/4	18000	1.1%	2/4	18000	1.0%	1/4	18000	1.1%	1/4
C57-60	30	700	100	15022.5	0.1%	4/4	15284.8	0.1%	4/4	15166.0	0.1%	4/4	15084.9	0.1%	4/4
C61-64	30	700	400	18000	0.9%	3/4	18000	0.9%	3/4	18000	0.9%	3/4	18000	0.9%	3/4
Average				11114.4	0.58%	72.9%	12808.3	0.84%	57.3%	12388.8	0.74%	63.5%	12518.6	0.79%	62.5%

The performance of the algorithm for small-, medium-, and large-scale instances is summarized in Table (7) in A.6. For small and medium-sized instances, the D_B ratio remains relatively low. However, as the problem size increases, the dominance of B&BC becomes more evident. In fact, for large-scale instances, B&BC strongly dominates Gurobi in 72.9% of the cases, and in 86.5% of the cases, Gurobi fails to strongly dominate the proposed B&BC.

6.4.2 Evaluating the effect of transshipment points

Constraint (7) in the PMICND formulation directly impacts route complexity by limiting the number of candidate transshipment points, and therefore the maximum number of hops for each commodity. When all network nodes are allowed to serve as transshipment points ($T = |\mathcal{N}|$), the model offers maximum flexibility in route design, whereas

restricting T to a smaller subset reduces this flexibility. We examine how variations in T affect the performance of the proposed B&BC algorithm. Four cases are considered for this analysis: $T = \{0.3|\mathcal{N}|, 0.5|\mathcal{N}|, 0.7|\mathcal{N}|, |\mathcal{N}|\}$. Performance is evaluated using three metrics: Time, Gap, and D_B, focusing on how the proposed acceleration strategies perform under these topological constraints. First, we examine the performance of WS and GH strategies, two components of the B&BC algorithm. Figure (5) underscores their complementary strength in enhancing algorithmic efficiency and solution quality as T increases. Each point on the plot represents the average results over 103 large-scale R & C instances. As illustrated, excluding either of WS or GH, increases the computational. This additional time, however, does not yield any improvement in the other performance measures. Excluding WS reduces D_B from an average of 72.9% to 64.6%, while excluding GH reduces it to 67.7%. Similarly, for the Gap metric, excluding WS increases the value from an average of 0.58% to 0.74%, whereas excluding GH results in a Gap of 0.62%.

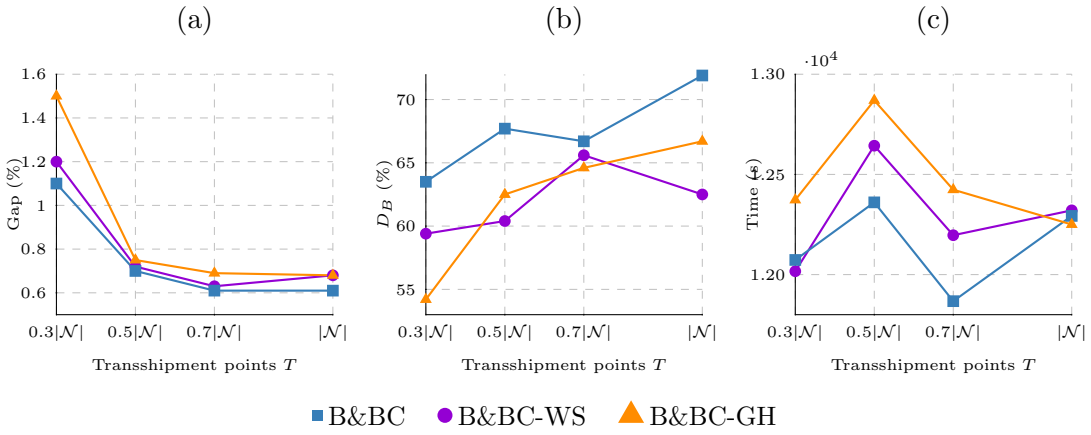


Figure 5: Analyzing the effect of WS and GH on the overall performance of the B&BC: (a) Gap, (b) D_B, and (c) Time (s) across different transshipment point configurations

Additionally, we evaluate the quality of lower-bound solutions produced by GH. Given that the R instances share similar network topology, they are used to assess the heuristic's performance under varying network characteristics, including the $\frac{f_{ij}}{c^k_{ij}}$ ratio and capacity tightness, represented as $(F/V, Cap)$. As illustrated in Figure (6), the y-axis represents $Gap_{GH} = \frac{MP_{OFV} - GH_{OFV}}{MP_{OFV}} \times 100$, where MP_{OFV} is the objective function value of the MP solved in 18000 sec time limit using B&BC, and GH_{OFV} is the objective function value obtained by the heuristic approach. The results show that on average this gap is approximately 29.74%. Considering that the heuristic approach (Algorithm 1) solves large-sized instances (R & C) in less than one second (approximately 0.2 sec on average), compared to an average of approximately 7991.6 seconds required by B&BC for the large R instances (r11–r18), this gap can be deemed acceptable. The quality of solution produced by Algorithm 1 shows a slight decline relative to the optimal solution as the $\frac{f_{ij}}{c^k_{ij}}$ ratio increases. A similar trend is observed when the network capacity becomes more constrained. However, in instance r12.9, where both $\frac{f_{ij}}{c^k_{ij}}$ and capacity tightness attain their

highest levels, Algorithm 1 finds the optimal solution for all values of T. Conversely, the worst performance occurs in instance r12.8, where the gap reaches 76.5% despite its $\frac{f_{ij}}{c_{ij}^k}$ ratio being lower than in r12.9.

Figure (7) illustrates the impact of incorporating cuts derived from the LPR of the SP, for networks with different sizes of T. As depicted, the addition of feasibility cuts ($B\&BC{+}FC$) significantly reduces the efficiency. The performance gap between $B\&BC$ and $B\&BC$-($OC\&FC$) represents the contribution of optimality cuts. The influence of these cuts becomes more pronounced with increasing values of T. As T increases, optimality cuts derived from the LPR of SP play a more critical role in enhancing the overall efficiency of the B&BC algorithm, specifically in D_B.

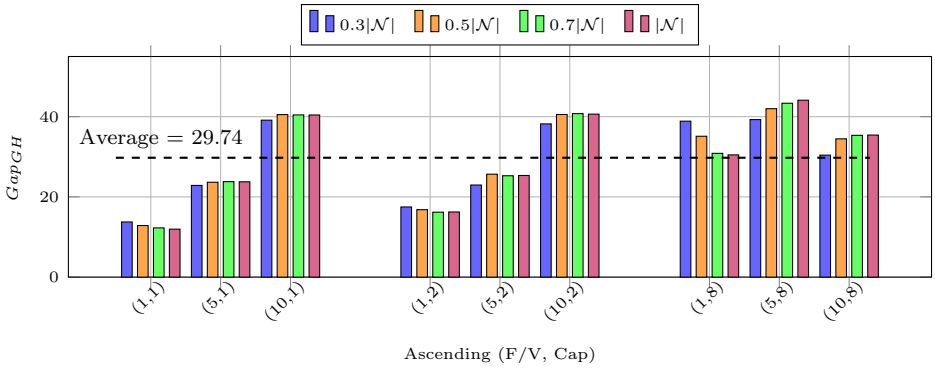


Figure 6: Best integer vs. heuristic solutions for varying capacities and F/V ratios.

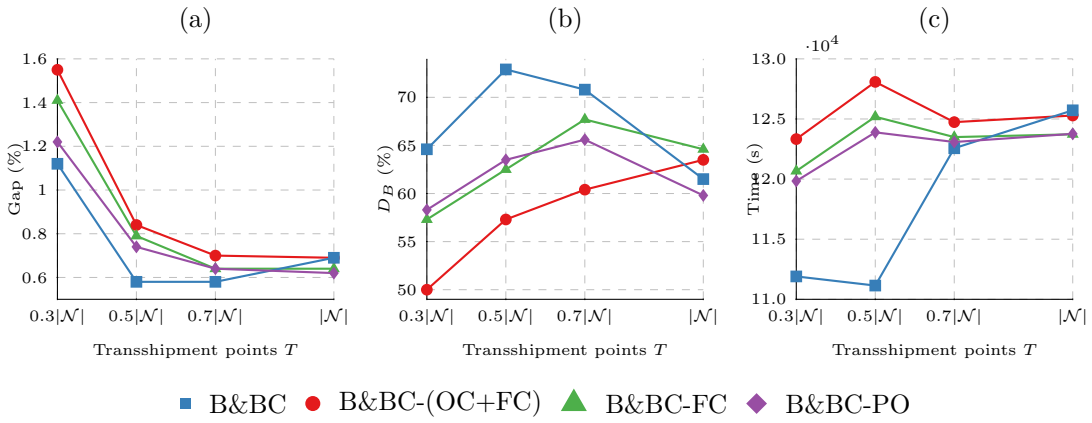


Figure 7: Evaluating the impact of individual components on the performance of the B&BC for different $|T|$ by comparing their (a) Gap, (b) D_B, and (c) Time (s).

Additionally, we evaluate the impact of developed VIs by examining their performance for different values of T. Figure (8) reports the results for each VI in a single block, showing Gap, D_B, and Time. The x-axis denotes the number of transshipment points, while the left y-axis shows Gap% and normalized Time (scaled to Gap for a compact

view). The right y-axis corresponds to D_B. Each point represents the average over 103 large-scale R&C instances. For $T = 0.3|\mathcal{N}|$ and $0.5|\mathcal{N}|$, all proposed VIs improve the algorithm's performance with respect to D_B. When examining all variations of T, we observe notable trends in how different VI influence performance metrics.
For the metric D_B, the MIC and VI_3, demonstrate the greatest impact. On average, across all T, excluding MIC and VI_3 from B&BC reduces D_B by 18.3% and 13.5%, respectively. In contrast, excluding VI_4 and VI_5 increases D_B by 2.3% and 0.8% respectively, however, this comes at the cost of degrade in Time by an average of 343.4 and 166.1 seconds, respectively. Turning to the Gap metric, the most influential VI are VI_3 and MIC. On average, excluding these VIs results in a 31.3%, and 29.7% increase in Gap, respectively. In contrast, excluding ND and VI_5 increase the Gap by 10.3%, and 6.3%, respectively. However this comes at the cost of increase in Time by an average of 88.1 and 166.1 seconds, respectively. A similar pattern emerges when considering the Time metric, where MIC and VI_3 again show the highest contributions. On average, excluding these VI results in increase in Time for 1603.1 and 718.3 seconds respectively. On average excluding none of the VI improves Time metric. Overall, considering all performance metrics and averaging across all variations of T, MIC emerges as the most influential VI, demonstrating the highest overall contribution.

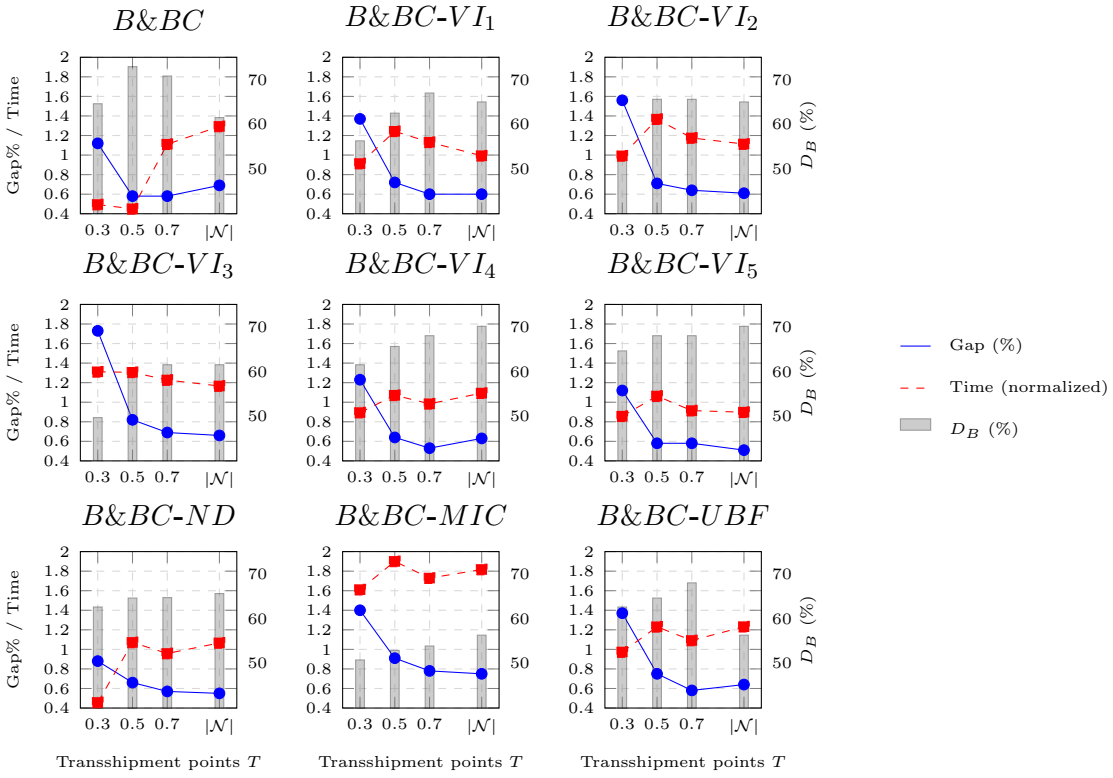
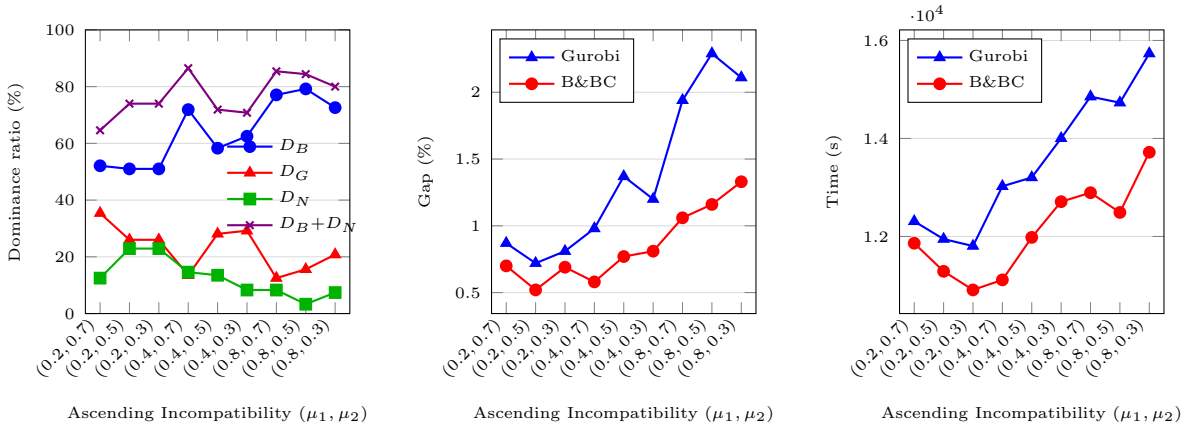


Figure 8: Analyzing the impact of individual VI on large R and C instances using performance metrics: Gap%, Time (normalized), and Dominance (D_B%).

6.4.3 Evaluating the effect of incompatibility

A second factor influencing the computational complexity of the PMICND model is the size of Λ. In this section, we assess the performance of B&BC under varying $|\Lambda|$ and $|\lambda_k|$ using large R and C instances. With $\mu_1 \in \{0.2, 0.4, 0.8\}$ and $\mu_2 \in \{0.3, 0.5, 0.7\}$, nine parameter combinations are examined to capture varying levels of incompatibility. We then analyze the impact of the MIC inequality (69) by varying $|\Lambda|$. Figure 9 compares B&BC and Gurobi across the different (μ_1, μ_2) settings, while Figure 9a presents the dominance ratio as the number of incompatible commodities increases. The results indicate that, as incompatibility grows, D_B increases and D_N decreases. Consequently, the gap between D_B and D_B+D_N becomes smaller, suggesting that B&BC dominates Gurobi more consistently under higher incompatibility. In addition, the B&BC algorithm provides higher-quality solutions and shorter computational times than Gurobi when incompatibility intensifies. Figures 9b and 9c illustrate these trends. Furthermore,



(a) (b) (c)

Figure 9: Impact of varying (μ_1, μ_2) on the performance of the B&BC algorithm on large R and C instances, evaluated using: (a) Dominance ratio, (b) Gap%, and (c) Time (s).

as illustrated in Figure (10), we examine the impact of the MIC inequality using nine distinct combinations of (μ_1, μ_2). The MIC inequality aggregates all incompatibilities of a commodity into a single constraint and, therefore, has greater significance when, for $k \in \mathcal{K}$, $\lambda_k > 2$, i.e., when a commodity has three or more incompatibilities involved. Its influence is thus expected to increase with $\lfloor \mu_1(1-\mu_2)|K| \rfloor$. In our analysis, μ_1 ranges from 20% to 80% and μ_2 varies from 30% to 70%, producing instances where the fraction of commodities with three or four incompatibilities ranges from 6% to 56%. Figure (10) compares the average performance of the *B&BC* algorithm over 103 large instances under different incompatibility settings. Each position on the horizontal axis corresponds to a pair (μ_1, μ_2). The left plot shows results obtained without incorporating MIC (*B&BC-MIC*), whereas the right plot includes them (*B&BC*). When MIC is considered, the Gap remains below about 1.1% across all (μ_1, μ_2). The normalized

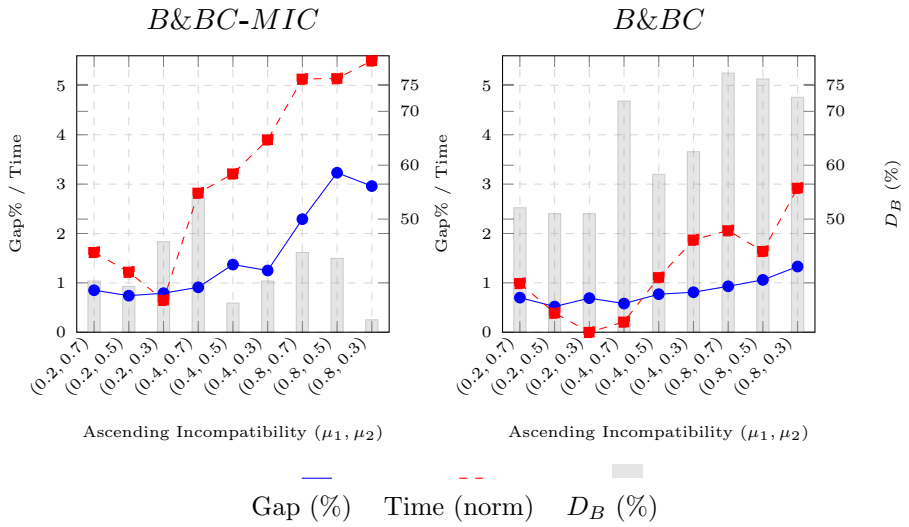


Figure 10: Evaluating the impact of MIC inequality on large R and C instances when $|\lambda_k|$ rises, using Gap%, normalized Time, and D_B.

computational time increases gradually from roughly 1 to 2.9 as $|\Lambda|$ increases, indicating stable performance despite increasing incompatibility. Moreover, the dominance measure D_B (gray bars) remains higher than 50%, and rises with the increase of $|\Lambda|$ reflecting that the majority of instances are strictly dominated compared to Gurobi. The consistent rise in the dominance measure D_B, even when 80% of commodities exhibit incompatibilities and more than half have $\lambda_k > 2$, indicates that the solution quality continues to improve despite the increasing computational complexity of the model.
In contrast, omitting the MIC inequality yields comparable performance only when incompatibilities are limited (i.e., less than $(0.2, 0.3)$); however, performance degrades notably as $|\Lambda|$ increases. For example, when $\mu_2 = 0.3$, increasing μ_1 from 0.2 to 0.8 raises the fraction of incompatible commodities from 20% to 80%. In this setting, the proportion of commodities with $\lambda_k \leq 2$ increases from 6% to 24%, while those with $\lambda_k > 2$ increase from 14% to 56%. Under this high-incompatibility scenario, the Gap rises from approximately 0.8% to over 2.9%, while the normalized computational time increases from roughly 0.65 to 5.5, with a considerable reduction in the dominance measure D_B. These results indicate that excluding MIC substantially reduces both solution quality and computational performance when a large share of commodities exhibit multiple incompatibilities, underscoring the importance of integrating MIC into the *B&BC* algorithm.

7 Conclusion

The multi-commodity capacitated network design problem (MCND) is critical for managing diverse flows while balancing cost, reliability, and performance, requiring careful

consideration of complex configuration choices. This study addresses a variant in which both the demand and supply sides are designed simultaneously. On the demand side, we select the most profitable commodities to serve, considering safety and regulatory incompatibility requirements that restrict certain commodities from sharing capacity, thus requiring either dedicated or compatible shipments. On the supply side, the network is designed to maximize overall profit through the selection of arcs and transshipment points, where transshipment nodes are those neither origins nor destinations but necessary routing points. We formally define this integrated optimization problem as the Profit-Maximizing Incompatible Multi-Commodity Network Design Problem with Location Decisions (PMICND) and present a generic mixed-integer programming (MIP) model applicable across telecommunications, transportation, and logistics industries. Our analysis confirms that disregarding transshipment and incompatibility requirements when designing the demand and supply sides leads to substantial financial losses. To navigate the problem's complexity, especially in large-scale settings, we developed a branch & Benders cut (B&BC) algorithm. We tested four different decompositions for the master and subproblems and compared their performance. To boost efficiency, the algorithm incorporates multiple acceleration techniques, such as a modified decomposition scheme, Benders dual and Pareto-optimal cuts, a warm-start procedure, a lower-bounding heuristic, and several classes of valid inequalities derived from the structural characteristics of the PMICND. Extensive computational experiments on benchmark instances show that the proposed algorithm performs reliably across varied network structures and incompatibility settings, often surpassing commercial solvers like Gurobi. In large networks with high levels of incompatibility, the B&BC achieved an average optimality gap of 1.3%, compared with Gurobi's 2.1%, and it outperformed Gurobi in roughly 72.6% of the test instances.

Acknowledgments

While working on the project, first author was a visiting researcher at the Université de Montréal, while the second author was Adjunct Professor, Department of Computer Science and Operations Research, Université de Montréal, and the third held the Canada Research Chair in Stochastic Optimization of Transport and Logistics Systems. We gratefully acknowledge the financial support provided by the Natural Sciences and Engineering Council of Canada (NSERC) through its Discovery Grant program, and the Fonds de recherche du Québec through their CIRRELT infrastructure grants.

References

T. H. Ali, S. Radhakrishnan, S. Pulat, and N. C. Gaddipati. Relay network design in freight transportation systems. *Transportation Research Part E: Logistics and Transportation Review*, 38:405–422, 2002.

A. Alibeyg, I. Contreras, and E. Fernández. A lagrangean relaxation for uncapacitated hub location problems with profits. In *Conference Proceedings of the CLAIO*, 2014.

J. Andersen and M. Christiansen. Designing new european rail freight services. *Journal of the Operational Research Society*, 60:348–360, 2009.

A. Atamtürk, G. L. Nemhauser, and M. W. Savelsbergh. Conflict graphs in solving integer programming problems. *European Journal of Operational Research*, 121(1): 40–55, 2000.

J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252, 1962.

I. C. Bilegan, T. G. Crainic, and Y. Wang. Scheduled service network design with revenue management considerations and an intermodal barge transportation illustration. *European Journal of Operational Research*, 300:164–177, 2022.

M. Bodur and J. R. Luedtke. Mixed-integer rounding enhanced Benders decomposition for multiclass service-system staffing and scheduling with arrival rate uncertainty. *Management Science*, 63(7):2073–2091, 2017.

M. Chouman, T. G. Crainic, and B. Gendron. Commodity representations and cut-set-based inequalities for multicommodity capacitated fixed-charge network design. *Transportation Science*, 51(2):650–667, 2017.

B. Cobeña, I. Contreras, L. I. Martínez-Merino, and A. M. Rodríguez-Chía. The profit-oriented hub line location problem with elastic demand. *Computers & Operations Research*, 159:106335, 2023.

I. Contreras. Hub network design. In *Network design with applications to transportation and logistics*, pages 567–598. Springer, 2021.

I. Contreras and E. Fernández. General network design: A unified view of combined location and network design problems. *European Journal of Operational Research*, 219:680–697, 2012.

A. M. Costa. A survey on Benders decomposition applied to fixed-charge network design problems. *Computers & operations research*, 32(6):1429–1450, 2005.

T. G. Crainic and B. Gendron. Exact methods for fixed-charge network design. In *Network Design with Applications to Transportation and Logistics*, pages 29–89. Springer, 2020.

T. G. Crainic and M. Hewitt. Service network design. In *Network design with applications to transportation and logistics*, pages 347–382. Springer, 2020.

T. G. Crainic, M. Gendreau, and B. Gendron. Fixed-charge network design problems. In *Network Design with Applications to Transportation and Logistics*, pages 15–28. Springer, 2021.

J. Current and H. Pirkul. The hierarchical network design problem with transshipment facilities. *European Journal of Operational Research*, 51(3):338–347, 1991.

J. R. Current. The design of a hierarchical transportation network with transshipment facilities. *Transportation Science*, 22(4):270–277, 1988.

E. Fernandez, I. Ljubic, and N. Zerega. The multi-commodity flow problem with outsourcing decisions. *arXiv preprint arXiv:2406.02439*, 2024.

P. Fontaine, T. G. Crainic, O. Jabali, and W. Rei. Scheduled service network design with resource management for two-tier multimodal city logistics. *European Journal of Operational Research*, 294(2):558–570, 2021.

B. Gendron, T. G. Crainic, and A. Frangioni. Multicommodity capacitated network design. In *Telecommunications network planning*, pages 1–19. Springer, 1999.

B. Gendron, M. G. Scutellà, R. G. Garroppo, G. Nencioni, and L. Tavanti. A branch-and-Benders-cut method for nonlinear power design in green wireless local area networks. *European Journal of Operational Research*, 255(1):151–162, 2016.

A. M. Geoffrion. Generalized Benders decomposition. *Journal of Optimization Theory and Applications*, 10:237–260, 1972.

W. Gu, C. Archetti, D. Cattaruzza, M. Ogier, F. Semet, and M. G. Speranza. Vehicle routing problems with multiple commodities: A survey. *European Journal of Operational Research*, 2023.

M. Hewitt and W. Rei. Perspectives on using benders decomposition to solve two-stage stochastic mixed-integer programs. In *Combinatorial Optimization and Applications: A Tribute to Bernard Gendron*, pages 259–276. Springer, 2024.

Y. Huang. A two-stage stochastic model for intermodal transportation operational planning problem under capacity shortage. In *2021 IEEE International Conference on Recent Advances in Systems Science and Engineering (RASSE)*, pages 1–9. IEEE, 2021.

G. Laporte and F. V. Louveaux. The integer l-shaped method for stochastic integer programs with complete recourse. *Operations Research Letters*, 13(3):133–142, 1993.

R. W. Lien, S. M. Iravani, K. Smilowitz, and M. Tzur. An efficient and robust design for transshipment networks. *Production and Operations Management*, 20(5):699–713, 2011.

W. J. Lipton and J. M. Harvey. Compatibility of fruits and vegetables during transport in mixed loads. *Department of Agriculture, Agricultural Research Service*, 1977.

T. L. Magnanti and R. T. Wong. Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria. *Operations research*, 29(3):464–484, 1981.

D. McDaniel and M. Devine. A modified Benders' partitioning algorithm for mixed integer programming. *Management Science*, 24(3):312–319, 1977.

R. Montemanni and D. H. Smith. On solving the maximum flow problem with conflict constraints. *arXiv preprint arXiv:2503.19685*, 2025.

F. A. Oliveira, E. M. de Sá, and S. R. de Souza. Benders decomposition applied to profit maximizing hub location problem with incomplete hub network. *Computers & Operations Research*, 142:105715, 2022.

F. A. Oliveira, E. M. de Sá, S. R. de Souza, and M. J. F. Souza. Ils-based algorithms for the profit maximizing uncapacitated hub network design problem with multiple allocation. *Computers & Operations Research*, 157:106252, 2023.

N. Papadakos. Practical enhancements to the Magnanti–Wong method. *Operations Research Letters*, 36(4):444–449, 2008.

R. Rahmaniani, T. G. Crainic, M. Gendreau, and W. Rei. The Benders decomposition algorithm: A literature review. *European Journal of Operational Research*, 259(3): 801–817, 2017.

R. Rahmaniani, T. G. Crainic, M. Gendreau, and W. Rei. Accelerating the Benders decomposition method: Application to stochastic network design problems. *SIAM Journal on Optimization*, 28(1):875–903, 2018.

R. Rahmaniani, S. Ahmed, T. G. Crainic, M. Gendreau, and W. Rei. The Benders dual decomposition method. *Operations Research*, 68(3):878–895, 2020.

Z. Şuvak, İ. K. Altınel, and N. Aras. Exact solution algorithms for the maximum flow problem with additional conflict constraints. *European Journal of Operational Research*, 287(2):410–437, 2020.

G. Taherkhani and S. A. Alumur. Profit maximizing hub location problems. *Omega*, 86: 1–15, 2019.

A. M. Voicu, L. Simić, and M. Petrova. Survey of spectrum sharing for inter-technology coexistence. *IEEE Communications Surveys & Tutorials*, 21(2):1112–1144, 2018.

M.-G. Yoon and J. Current. The hub location and network design problem with fixed and variable arc costs: formulation and dual-based solution heuristic. *Journal of the Operational Research Society*, 59:80–89, 2008.

B. Yıldız, O. E. Karaşan, and H. Yaman. Branch-and-price approaches for the network design problem with relays. *Computers & Operations Research*, 92:155–169, 2018.

C. A. Zetina, I. Contreras, and J.-F. Cordeau. Exact algorithms based on Benders decomposition for multicommodity uncapacitated fixed-charge network design. *Computers & Operations Research*, 111:311–324, 2019a.

C. A. Zetina, I. Contreras, and J.-F. Cordeau. Profit-oriented fixed-charge network design with elastic demand. *Transportation Research Part B: Methodological*, 127:1–19, 2019b.

A Decomposition Strategies (DS_3, DS_4)

A.0.1 Strategy DS_3

The MP is formulated as (71)-(76). Similar to DS_2, an integer optimality cut (73) is added to the MP for each $\gamma \in \Gamma_{DS_3}$ where Γ_{DS_3} is the set of all integer feasible solutions for which we have $Z^*_{SP_{DS_3}} < \overline{\theta}_{DS_3}$ in the DS_3 strategy, where $\overline{\theta}_{DS_3}$ is the approximation of the objective function SP_{DS_3}. On the other hand, a combinatorial cut (74) is added to the MP for each $\omega \in \Omega_{DS_3}$ where Ω_{DS_3} is the set of all infeasible solutions found in the B&B tree, in DS_3 strategy. The MP for DS_3 is formulated as follows:

$$\max\ Z_{MP_{DS_3}} = -\sum_{j\in\mathcal{N}} f_j z_j - \sum_{(i,j)\in\mathcal{A}} f_{ij} y_{ij} + \theta \tag{71}$$

$$s.t. \quad (7) \tag{72}$$

$$Optimality\ Cuts \quad \forall \gamma \in \Gamma_{DS_3} \tag{73}$$

$$Feasibility\ Cuts \quad \forall \omega \in \Omega_{DS_3} \tag{74}$$

$$z_j \in \{0,1\} \quad \forall j \in \mathcal{N} \tag{75}$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in \mathcal{A} \tag{76}$$

The SP for DS_3 is formulated as follows:

$$\max\ Z_{SP_{DS_3}} = \sum_{k\in\mathcal{K}} r^k d^k w^k - \sum_{(i,j)\in\mathcal{A}} \sum_{k\in\mathcal{K}} c^k_{ij} x^k_{ij} \tag{77}$$

$$s.t. \quad (2), (4), (5) \tag{78}$$

$$\sum_{k\in\mathcal{K}} x^k_{ij} \le u_{ij} \overline{y}_{ij} \quad \forall (i,j) \in \mathcal{A} \tag{79}$$

$$\sum_{\substack{k\in\mathcal{K}:\\ \{O(k)\neq j, D(k)\neq j\}}} \sum_{\substack{i\in\mathcal{N}:\\ (i,j)\in\mathcal{A}}} x^k_{ij} \le u_j \overline{z}_j \quad \forall j \in \mathcal{N} \tag{80}$$

$$w^k, v^k_{ij} \in \{0,1\},\ x^k_{ij} \in \mathbb{R}_{\ge 0} \quad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K} \tag{81}$$

The following optimality cut is added to MP_{DS_3} once an integer feasible solution is found.

$$\theta \le \left(Z_{\mathrm{SP}_{DS_3}} - UB_{\mathrm{SP}_{DS_3}}\right) \times \left(\sum_{(i,j)\in\eta_y} y_{ij} - \sum_{(i,j)\notin\eta_y} y_{ij} + \sum_{j\in\eta_z} z_j - \sum_{j\notin\eta_z} z_j - (|\eta_y| + |\eta_z|) \right) + Z_{\mathrm{SP}_{DS_3}} \tag{82}$$

$UB_{\mathrm{SP}_{DS_3}}$ denotes an upper bound on $Z_{\mathrm{SP}_{DS_3}}$. Furthermore, $\eta_y = \{(i,j) \in \mathcal{A} : y_{ij} = 1\}$ is the set of arcs where the variable y_{ij} takes the value 1 for that integer feasible solution. η_z is defined similarly. In cases where the SP_{DS_3} results in an infeasible solution, we add

the following combinatorial cut to exclude the current infeasible solution from further being investigated.

$$\sum_{(i,j)\in\eta_y}(1-y_{i,j})+\sum_{(i,j)\notin\eta_y}y_{i,j}+\sum_{j\in\eta_z}(1-z_j)+\sum_{j\notin\eta_z}z_j\geq 1 \tag{83}$$

A.0.2 Strategy DS_4

The MP is formulated as (84)-(87). Likewise DS_2 and DS_3 an integer optimality cut (85) is added to the MP for each $\gamma\in\Gamma_{DS_4}$ where Γ_{DS_4} is the set of all integer feasible solutions for which we have $Z^*_{SP_{DS_4}}<\overline{\theta}_{DS_4}$ in the DS_4 strategy, where $\overline{\theta}_{DS_4}$ is the approximation of the SP_{DS_4} objective function. On the other hand, a combinatorial cut (86) is added to the MP for each $\omega\in\Omega_{DS_4}$ where Ω_{DS_4} is the set of all infeasible solutions found in the B&B tree, in DS_4 strategy. The MP for DS_4 is as follows:

$$\max\ Z_{MP_{DS_4}}=\sum_{k\in\mathcal{K}}r^kd^kw^k-\sum_{(i,j)\in\mathcal{A}}f_{ij}y_{ij}+\theta \tag{84}$$

$$s.t.\quad Optimality\ Cuts\quad \forall\gamma\in\Gamma_{DS_4} \tag{85}$$

$$Feasibility\ Cuts\quad \forall\omega\in\Omega_{DS_4} \tag{86}$$

$$y_{ij},w^k\in\{0,1\}\quad\forall(i,j)\in\mathcal{A},k\in\mathcal{K} \tag{87}$$

The SP for DS_4 is formulated as follows:

$$\max\ Z_{SP_{DS_4}}=-\sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}}c^k_{ij}x^k_{ij}-\sum_{j\in\mathcal{N}}f_jz_j \tag{88}$$

$$s.t.\quad (4),(5),(6),(7),(8) \tag{89}$$

$$\sum_{j\in\mathcal{A}^+_i}x^k_{ij}-\sum_{j\in\mathcal{A}^-_i}x^k_{ji}=\begin{cases}d^k\overline{w}^k, & i=O(k)\\ -d^k\overline{w}^k, & i=D(k)\\ 0, & i\neq O(k), i\neq D(k)\end{cases}\qquad\forall k\in\mathcal{K},i\in\mathcal{N} \tag{90}$$

$$\sum_{k\in\mathcal{K}}x^k_{ij}\leq u_{ij}\overline{y}_{ij}\quad\forall(i,j)\in\mathcal{A} \tag{91}$$

$$v^k_{ij}\in\{0,1\},\ x^k_{ij}\in\mathbb{R}_{\geq 0}\quad\forall(i,j)\in\mathcal{A},k\in\mathcal{K} \tag{92}$$

The following optimality cut is added to the MP_{DS_4} once an integer feasible solution is found.

$$\theta\leq\left(Z_{\mathrm{SP}_{DS_4}}-UB_{\mathrm{SP}_{DS_4}}\right)\times\left(\sum_{(i,j)\in\eta_y}y_{ij}-\sum_{(i,j)\notin\eta_y}y_{ij}+\sum_{k\in\eta_w}w_k-\sum_{k\notin\eta_w}w_k-(|\eta_y|+|\eta_w|)\right)+Z_{\mathrm{SP}_{DS_4}} \tag{93}$$

In this context, $UB_{\mathrm{SP}_{DS_4}}$ denotes an upper bound on $Z_{\mathrm{SP}_{DS_4}}$. Furthermore, $\eta_y = \{(i,j) \in \mathcal{A} : y_{ij} = 1\}$ is the set of arcs where the variable y_{ij} takes the value 1 for that integer feasible solution. The sets η_w and η_z are defined similarly. In cases where the SP_{DS_4} results in an infeasible solution, we add the following combinatorial cut to exclude the current infeasible solution from further being investigated.

$$\sum_{(i,j)\in\eta_y} (1 - y_{i,j}) + \sum_{(i,j)\notin\eta_y} y_{i,j} + \sum_{k\in\eta_w} (1 - w_k) + \sum_{k\notin\eta_w} w_k \geq 1 \tag{94}$$

A.1 Feasibility Problems

The SPF_{DS_2} is presented as follows:

$$\begin{aligned} \text{Min} \quad Z_{SPF_{DS_2}} = & \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq O(k)}} \epsilon^1_{ik} + \epsilon^2_{ik} + \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq D(k)}} \epsilon^3_{ik} + \epsilon^4_{ik} + \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq O(k) \\ i\neq D(k)}} \epsilon^5_{ik} + \epsilon^6_{ik} \\ & + \sum_{(i,j)\in\mathcal{A}} \epsilon^7_{ij} + \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}} \epsilon^8_{ijk} + \epsilon^9_{ijk} + \sum_{j\in\mathcal{N}} \epsilon^{10}_j \\ & s.t. \ (21)-(23), (26)-(33), (35)-(36) \end{aligned} \tag{95}$$

Consider $\hat{\beta}^1_{i,j}$, $\hat{\beta}^2_k$, and $\hat{\beta}^3_j$ to be the dual multipliers of constraints (31)-(33) respectively. We can then generate the feasibility cut as follows:

$$\begin{aligned} 0 \geq & \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq O(k)}} (\hat{\epsilon}^1_{ik} + \hat{\epsilon}^2_{ik}) + \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq D(k)}} (\hat{\epsilon}^3_{ik} + \hat{\epsilon}^4_{ik}) + \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq O(k) \\ i\neq D(k)}} (\hat{\epsilon}^5_{ik} + \hat{\epsilon}^6_{ik}) + \sum_{(i,j)\in\mathcal{A}} \hat{\epsilon}^7_{ij} + \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}} (\hat{\epsilon}^8_{ijk} + \hat{\epsilon}^9_{ijk}) \\ & + \sum_{j\in\mathcal{N}} \hat{\epsilon}^{10}_j + \sum_{(i,j)\in\mathcal{A}} \hat{\beta}^1_{i,j}(y_{i,j} - \hat{y}_{i,j}) + \sum_{k\in\mathcal{K}} \hat{\beta}^2_k(w_k - \hat{w}_k) + \sum_{j\in\mathcal{N}} \hat{\beta}^3_j(z_j - \hat{z}_j) \end{aligned} \tag{96}$$

After generating feasibility cuts (96), the following feasibility problem ($SPFL_{DS_2}$) can be solved to generate strengthened feasibility cuts (98). Similarly, ϵ denotes a decision variable that captures the amount of deviation from feasibility to be minimized, and $\tilde{\epsilon}$ represents its value.

$$\begin{aligned} \text{Min} \quad Z_{SPFL} = & \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq O(k)}} \epsilon^1_{ik} + \epsilon^2_{ik} + \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq D(k)}} \epsilon^3_{ik} + \epsilon^4_{ik} + \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq O(k) \\ i\neq D(k)}} \epsilon^5_{ik} + \epsilon^6_{ik} \\ & + \sum_{(i,j)\in\mathcal{A}} \epsilon^7_{ij} + \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}} \epsilon^8_{ijk} + \epsilon^9_{ijk} + \sum_{j\in\mathcal{N}} \epsilon^{10}_j - \sum_{(i,j)\in\mathcal{A}} \hat{\beta}^1_{i,j}(y_{i,j} - \overline{y}_{i,j}) \\ & - \sum_{k\in\mathcal{K}} \hat{\beta}^2_k(w_k - \overline{w}_k) - \sum_{j\in\mathcal{N}} \hat{\beta}^3_j(z_j - \overline{z}_j) \end{aligned} \tag{97}$$

$$s.t.(8),(9),(26)-(33),(35)-(36)$$

Assume that, $\tilde{y}_{i,j}$, $\tilde{w}_k$ and $\tilde{z}_j$ are the values obtained from solving $SPFL_{DS_2}$. We can then generate the strengthened feasibility cut as follows:

$$\begin{aligned}0 \geq & \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K}\\ i\neq O(k)}} \tilde{\epsilon}^1_{ik} + \tilde{\epsilon}^2_{ik} + \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K}\\ i\neq D(k)}} \tilde{\epsilon}^3_{ik} + \tilde{\epsilon}^4_{ik} + \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K}\\ i\neq O(k)\\ i\neq D(k)}} \tilde{\epsilon}^5_{ik} + \tilde{\epsilon}^6_{ik} + \sum_{(i,j)\in\mathcal{A}} \tilde{\epsilon}^7_{ij} + \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}} \tilde{\epsilon}^8_{ijk} + \tilde{\epsilon}^9_{ijk} \\ & + \sum_{j\in\mathcal{N}} \tilde{\epsilon}^{10}_j + \sum_{(i,j)\in\mathcal{A}} \hat{\beta}^1_{i,j}(y_{i,j} - \tilde{y}_{i,j}) + \sum_{k\in\mathcal{K}} \hat{\beta}^2_k(w_k - \tilde{w}_k) + \sum_{j\in\mathcal{N}} \hat{\beta}^3_j(z_j - \tilde{z}_j)\end{aligned} \tag{98}$$

A.2 Proof of Proposition 1.

Proof. Proof. A clique in the conflict graph $\mathcal{C}$ is a complete subgraph where every pair of vertices (commodities) is connected by an edge, meaning all commodities in the clique are mutually incompatible. A maximal clique is one not properly contained in any larger clique. Due to mutual incompatibility, at most one commodity from any clique can be routed on a given arc (i, j), as selecting two or more would violate the incompatibility constraints enforced in the model. Thus, the sum of the binary (or relaxed) variables v^k_{ij} over commodities $k \in \mathcal{C}_l$ cannot exceed 1. It should be noted that the following always holds: $|\mathcal{C}_l| \geq 2$. However, in the case of $|\mathcal{C}_l| = 2$, cliques are already imposed via (68), therefore cliques found for $|\mathcal{C}_l| \geq 3$ aggregates multiple pairwise inequalities from (68) into a single, stronger cut, reducing redundancy while preserving validity. □

A.3 UBF for DS_3 and DS_4

Considering the objective function of the SP for each of DS_3 and DS_4, their corresponding upper-bounding function will be as follows:

$$\boldsymbol{DS_3}: \quad \theta \leq \sum_{k\in\mathcal{K}} w^k r^k d^k - \sum_{k\in\mathcal{K}} C^k_{P(O(k),D(k))} d^k w^k \tag{99}$$

$$\boldsymbol{DS_4}: \quad \theta \leq -\sum_{k\in\mathcal{K}} C^k_{P(O(k),D(k))} d^k w^k - \sum_{j\in\mathcal{N}} f_j z_j \tag{100}$$

A.4 Pseudo codes

Algorithm 1 Benders Decomposition Cutting Plane (Phase I)

Relax the integrality of MP variables (based on DS)
Init $cp_{iter} = 0$, $stable_{iter} = 0$, $prev_{obj} = \infty$, max_{iter}, α_0, α_1
while $cp_{iter} < max_{iter}$ **do**
  $cp_{iter} \leftarrow cp_{iter} + 1$
  Solve MP, get $curr_{obj}$
  if $|curr_{obj} - prev_{obj}| < 10^{-6}$ **then**
    $stable_{iter} \leftarrow stable_{iter} + 1$
    if $stable_{iter} \geq \alpha_0$ **then**
      break
    end
  else
    $stable_{iter} \leftarrow 0$
  end
  $prev_{obj} \leftarrow curr_{obj}$
  Solve SP with MP solution
  if *SP infeasible* **then**
    Add generalized feasibility cut
  else if $SP.objVal < \theta$ **then**
    Add generalized optimality cut
end
Remove top α_1% slack cuts and restore integrality of MP variables

Algorithm 2 Greedy Heuristic for Incompatible Commodities

Procedure GREEDY HEURISTIC*(Network data)*
  Initialize data structures for network state
    `// Phase 1: Process compatible commodities`
  for *each commodity k without incompatibilities* **do**
    path $\leftarrow$ FINDFEASIBLEPATH(k)
    if *path exists and satisfies capacity* **then**
      if *profit of k on path* > 0 **then**
        Select k and update network
      end
    end
  end
    `// Phase 2: Process incompatible commodities`
  Create priority queue Q prioritized by high profit & low degree of incompatibility
  for *each commodity k with incompatibilities* **do**
    path $\leftarrow$ FINDFEASIBLEPATH(k)
    if *path exists and profit* > 0 **then**
      Add k to Q
    end
  end
  while $Q \neq \emptyset$ **do**
    Extract most profitable k from Q
    if *k is compatible and fits capacity* **then**
      Select k and update network
    end
  end
  Output Heuristic OFV

Algorithm 3 Accelerated branch & Benders cut algorithm (Phase II)

Initialization:

1. Apply MD on MP and initialize tree $\mathcal{T}$ with MP.
2. Initialize $\mathcal{T}$ with cuts from Phase I.
3. Consider the Heuristic OFV as a Cutoff to prune nodes.

while $\mathcal{T}$ *not empty* **do**
  select a node from $\mathcal{T}$ and solve:
  if *in root node* **then**
    if DS_1 **then**
      for *each fractional solution* **do**
        if *Corresponding SP is feasible* **then**
          Add POC & SOC
        else
          Add SFC
        end
      end
    end
    for *each integer solution* **do**
      if *Corresponding SP is feasible* **then**
        DS_1 : Add GOC, POC & UBF / DS_2,DS_3,DS_4 : also add SOC and IOC
      else
        DS_1 : Add GFC / feasibility cuts discarded in DS_2,DS_3,DS_4
      end
    end
  else
    if DS_1 **then**
      for *once every* α_2 *nodes in fractional solutions* **do**
        if *SP is feasible* **then**
          Add POC & SOC
        else
          Add SFC
        end
      end
    end
    for *each integer solution* **do**
      if *Corresponding SP is feasible* **then**
        DS_1 : Add GOC, POC & UBF / DS_2,DS_3,DS_4 : also add SOC and IOC
      else
        DS_1 : Add GFC / feasibility cuts discarded in DS_2,DS_3,DS_4
      end
    end
  end
end

A.5 Implementation details of the B&BC

As discussed in Section 5.2, we implement the Benders decomposition algorithm in two phases. In the first phase, we solve the root node by relaxing the MP and SP (in

DS_2, DS_3, and DS_4) and apply a classical cutting-plane procedure. We also incorporate the VIs into the MP (for DS_2: equations (59), (60)–(63), and (66)). Since both MP and SP integrality constraints are relaxed in this stage, the resulting SP often becomes highly infeasible, leading to only marginal improvements in the objective value. Based on preliminary experiments, we set $cp_{\text{max-iter}} = 20$, $\alpha_0 = 3$, and $\alpha_1 = 10\%$. In the second phase, we restore the integrality conditions and apply MD (excluding DS_1) and solve the problem using a B&C. In this phase we start exploring the search tree and for every solution found in the tree we add an optimality or a feasibility cut. To efficiently explore the tree, we use a greedy approach to find a feasible solution to act as a lower bound for the MP. The VIs (for DS_2: equations (58), (67), and (69)) are incorporated in this phase as a result of MD. VIs (64), (65), and (70) are added on the fly. As illustrated in Algorithm 3, fractional solutions are only explored for DS_1 in the root node and once every $\alpha_2 = 100$ nodes in the tree to strengthen the LPR. In the case of DS_2,DS_3 and DS_4, these fractional solutions do not provide good information for the optimality or feasibility cuts, therefore, they are only explored at the root node to strengthen the LPR by adding VIs ((64), (65)). Additionally, for every integer solution in the root node (for DS_2,DS_3 and DS_4), the Benders dual and Pareto optimal cuts are added. To enhance the efficiency of the B&BC, in the second phase, we omit the feasibility cuts derived from the LPR of SP (for DS_2, DS_3, and DS_4). These cuts are derived from the extreme rays of the LPR of SP, whereas in these decomposition strategies, the focus is on the infeasibility of SP with integrality constraints. As a result, feasibility cuts fail to convey valuable information and have a limited impact on improving the objective function value. However, this is not the case for DS_1, where such cuts remain informative and beneficial.

A.6 Complementary results

Table 7: B&BC vs Gurobi comparison for $T = 0.5|\mathcal{N}|$ candidate transshipment points.

Category	Instance	$\|\mathcal{N}\|$	$\|\mathcal{A}\|$	$\|\mathcal{K}\|$	Gurobi		*B&BC*		Dominance Ratio		
					Time(s)	Gap	Time(s)	Gap	D_G	D_B	D_N
Small	r04	10	60	10	0.3	0.0%	1.0	0.0%	-	-	9
	r05	10	60	25	2.7	0.0%	17.5	0.0%	1	-	8
	r06	10	60	50	40.2	0.0%	35.2	0.0%	3	-	6
	C33–36	20	230	40	6.5	0.0%	18.1	0.0%	-	-	4
	C41–44	20	300	40	25.3	0.0%	29.7	0.0%	-	-	4
	Avg. Small				15.0	0.0%	20.3	0.0%	20.0%	0.0%	80.0%
Medium	r07	10	82	10	1.0	0.0%	801.7	0.0%	1	-	8
	r08	10	83	25	5.1	0.0%	12.1	0.0%	-	-	9
	r09	10	83	50	38.5	0.0%	40.0	0.0%	-	-	9
	r10	20	120	40	519.5	0.0%	1117.3	0.0%	3	-	3
	Avg. Medium				141.0	0.0%	492.8	0.0%	11.1%	0.0%	88.9%
Large	r11	20	120	100	4399.5	0.2%	1102.7	0.0%	1	1	7
	r12	20	120	200	6420.5	0.7%	4931.7	0.4%	4	3	2
	r13	20	220	40	5204.9	0.0%	1273.2	0.0%	4	-	5
	r14	20	220	100	10194.3	0.8%	6518.7	0.4%	-	8	1
	r15	20	220	200	15434.2	2.2%	14065.2	0.6%	3	6	-
	C37–40	20	230	200	18000	1.6%	18000	1.0%	-	3	1
	C45–48	20	300	200	18000	0.9%	18000	0.8%	-	3	1
	r16	20	314	40	5689.9	0.1%	653.0	0.0%	-	7	2
	r17	20	318	100	17035.8	1.4%	7745.5	0.2%	1	2	6
	r18	20	315	200	18000	3.5%	16514.1	2.8%	2	7	-
	C49–52	30	520	100	18000	0.4%	15794.3	0.3%	-	1	3
	C53–56	30	520	400	18000	0.9%	18000	1.1%	-	2	2
	C57–60	30	700	100	15986.4	0.1%	15022.5	0.0%	1	-	3
	C61–64	30	700	400	18000	1.0%	18000	0.9%	1	3	-
	Avg. Large				13469.6	1.0%	11114.4	0.5%	13.5%	72.9%	13.6%

Instance	Gurobi		*B&BC*			*B&BC-MIC*			*B&BC-ND*			*B&BC-UBF*			*B&BC-VI*$_1$			*B&BC-VI*$_2$			*B&BC-VI*$_3$			*B&BC-VI*$_4$			*B&BC-VI*$_5$		
	Time	Gap	Time	Gap	D_B	Time	Gap	D_B	Time	Gap	D_B	Time	Gap	D_B	Time	Gap	D_B	Time	Gap	D_B	Time	Gap	D_B	Time	Gap	D_B	Time	Gap	D_B
r11	4399.5	0.2%	1102.7	0.0%	4/9	10303.0	0.4%	4/9	6207.8	0.3%	4/9	6765.0	0.2%	4/9	6358.8	0.2%	4/9	7404.6	0.3%	4/9	6140.5	0.2%	4/9	6483.2	0.2%	4/9	6420.3	0.1%	4/9
r12	6420.5	0.7%	4921.1	0.0%	3/9	10295.6	0.6%	3/9	7836.3	0.4%	3/9	8598.7	0.4%	3/9	8801.9	0.5%	3/9	9301.2	0.1%	3/9	8701.7	0.6%	2/9	7735.2	0.1%	3/9	8215.5	0.1%	3/9
r13	5528.9	0.0%	1272.7	0.0%	7/9	7350.0	0.3%	4/9	5184.4	0.0%	6/9	5331.7	0.0%	5/9	4999.4	0.0%	5/9	5178.8	0.0%	5/9	5310.9	0.0%	5/9	4775.7	0.0%	5/9	4720.1	0.0%	5/9
r14	10014.3	0.8%	6518.7	0.4%	8/9	8937.5	0.5%	5/9	6829.0	0.5%	6/9	6969.0	0.4%	5/9	7220.9	0.6%	5/9	7224.0	0.4%	6/9	7127.3	0.3%	5/9	6243.7	0.3%	6/9	6610.9	0.5%	6/9
r15	15434.2	2.2%	14065.2	0.6%	6/9	18000.1	1.2%	6/9	14556.7	1.3%	6/9	15089.4	1.4%	6/9	15213.6	1.2%	5/9	15391.3	1.4%	6/9	15478.4	1.6%	6/9	15412.5	1.1%	6/9	14560.9	0.8%	6/9
C37-40	18003.3	1.6%	18000.1	1.0%	4/4	18000.2	1.2%	4/4	18000.1	1.7%	4/4	18000.2	0.9%	4/4	18000.1	1.0%	4/4	18000.2	1.0%	4/4	18000.1	1.1%	4/4	18000.2	0.9%	4/4	17940.6	0.9%	4/4
C45-48	18004.1	0.9%	18000.1	0.8%	3/4	18000.2	0.6%	3/4	18000.1	0.8%	3/4	18000.2	0.7%	4/4	18000.2	0.8%	3/4	18000.2	0.9%	2/4	18000.2	0.9%	2/4	18000.2	0.8%	4/4	18000.1	0.7%	4/4
r16	5689.9	0.1%	633.0	0.0%	7/9	2745.0	0.0%	6/9	2609.1	0.0%	6/9	2602.0	0.0%	6/9	2482.1	0.0%	6/9	2502.6	0.0%	6/9	2507.6	0.0%	6/9	2469.0	0.0%	6/9	2624.5	0.0%	6/9
r17	17035.8	1.4%	1744.5	0.2%	9/9	11762.5	0.9%	6/9	7759.0	0.5%	8/9	9341.1	0.3%	7/9	9151.6	0.4%	7/9	9500.2	0.3%	8/9	9782.2	0.3%	9/9	8569.5	0.3%	9/9	8133.9	0.2%	9/9
r18	18004.1	3.5%	16514.1	2.8%	7/9	17724.1	3.7%	4/9	17159.7	2.7%	8/9	16610.6	3.5%	5/9	16845.2	2.9%	7/9	16684.6	3.2%	6/9	16934.1	3.2%	6/9	16656.8	2.9%	8/9	16645.0	2.7%	7/9
C49-52	18004.2	0.4%	15794.3	0.3%	4/4	15853.9	0.4%	3/4	16320.6	0.2%	4/4	15861.2	0.2%	4/4	16279.9	0.3%	3/4	17182.2	0.3%	4/4	18000.2	0.4%	3/4	15919.7	0.3%	4/4	15853.6	0.2%	4/4
C53-56	18020.5	0.9%	18000.6	1.1%	1/4	18000.7	1.1%	1/4	18000.7	1.0%	1/4	18000.6	1.0%	1/4	18000.7	1.0%	1/4	18000.6	1.1%	2/4	18000.6	1.2%	1/4	18000.6	1.0%	1/4	18000.7	0.7%	3/4
C57-60	15988.6	0.1%	15022.5	0.1%	4/4	15992.6	0.1%	1/4	14412.0	0.0%	4/4	15330.0	0.1%	4/4	14536.2	0.1%	2/4	15223.5	0.1%	2/4	15302.8	0.1%	1/4	14374.3	0.1%	3/4	14688.2	0.0%	3/4
C61-64	18002.7	1.0%	18000.7	0.9%	3/4	18000.7	1.0%	2/4	18000.8	0.9%	3/4	18000.1	0.9%	3/4	18000.7	0.9%	2/4	18000.7	0.9%	2/4	19000.8	1.3%	1/4	18000.7	0.9%	3/4	18000.7	0.7%	3/4
Average	13469.6	1.0%	11114.4	0.58%	72.2%	13620.0	0.91%	54.2%	12128.3	0.66%	66.7%	12464.3	0.75%	63.5%	12485.3	0.72%	62.5%	12699.6	0.71%	65.6%	12523.4	0.82%	60.4%	12188.7	0.64%	68.8%	12173.3	0.58%	70.8%

Table 8: Evaluating the effect of different classes of VI for $T = 0.5|\mathcal{N}|$