CIRRELT

CIRRELT-2026-22

Generalized Network Design with Demand Selection and Incompatible Commodities

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

Novembre 2026

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

Generalized Network Design with Demand Selection and Incompatible Commodities

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

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

Abstract. This study focuses on a profit maximization variant of the generalized network design problem with 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

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 Research 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, 2026

© Mirzaei, Crainic, Rei, Nikbakhsh, Kashan and CIRRELT, 2026

1 Introduction

Network design (ND) problems are fundamental to the management of complex systems in which demand must be satisfied through the organization of an appropriate supply network. In the *Multi-commodity Capacitated 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, particularly for strategic decisions about network configuration and tactical decisions that guide the operations of the system. The objectives of these problems are determined based on requirements, including minimizing the overall cost of network design or maximizing profit, reliability, and service levels (Crainic and Hewitt, 2021; Crainic et al., 2021).

The complexity of the MCND problems is well established (Gendron et al., 1999). Our study focuses on a problem setting characterized by additional sources of combinatorial complexity on both the demand and supply sides, as commonly encountered in various practical applications. On the demand side, one has to decide the commodities to satisfy while accounting for 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, the problem explicitly incorporates the selection of nodes from a predefined set of locations within the network, to determine which ones may serve as transshipment points for commodity flows. These interdependent decisions substantially expand the solution space and increase the overall computational difficulty of the problem.

We refer to this integrated setting as the *Generalized Network Design Problem with Demand Selection and Incompatible Commodities* (*GNDDI*). 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 selected commodities minus the fixed setup costs for selected facilities and arcs, and variable commodity distribution costs. Commodities are to be shipped either directly from origin to destination or via one or more open transshipment centers, incompatible flows being kept separate.

This problem is motivated by multiple real-world applications. In *telecommunication network design*, companies determine the optimal placement of nodes (e.g., 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*, physical 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 the *Dedicated Shipment Agreements*, which require certain commodities to be shipped separately using dedicated 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.

Our contribution is three-fold. First, we introduce a new class of MCND problems that jointly optimizes demand selection, commodity routing, and transshipment facility location under explicit commodity incompatibility constraints. Driven by a profit-maximizing objective, the problem requires selecting a subset of demands to satisfy fully while respecting operational restrictions that prevent incompatible commodities from sharing arc capacities. Intermediate nodes can be established as transshipment facilities, granting access to finite capacities at a fixed activation cost. This setup allows flexible routing, where commodities can move directly, via transshipment points, or through a combination of pathways.

Second, we formulate a mixed-integer programming (MIP) model for the GNDDI problem. This mathematical representation captures trade-offs between the profit of selecting the maximum number of compatible and incompatible demands and the cost of discrete facility deployment and continuous distribution flows, shaped by the routing dynamics of incompatible commodities.

Third, we propose an exact branch-and-Benders-cut (B&BC) algorithm to exploit the structure of the MIP model. A key focus is the systematic investigation of alternative partitioning strategies for the Benders decomposition. Through extensive computational experimentation on different partitioning configurations of the complicating decisions (network design, transshipment location, demand selection, and routing of incompatible commodities), we empirically identify a highly efficient configuration that effectively balances computational burden with search guidance. The decomposition is further enhanced with acceleration techniques and valid inequalities to expedite convergence.

Finally, using standard benchmark instances, this study evaluates the performance of the B&BC algorithm and the efficiency of various decomposition strategies, demonstrating its effectiveness under different problem characteristics. Ultimately, our analysis quantifies the economic and operational benefits of explicitly integrating commodity incompatibilities and transshipment facility selection within a profit-maximizing MCND framework.

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. Finally, conclusions and future research directions are presented in section 7.

2 Related literature

Generalized network design problems involve selecting nodes and arcs and routing commodities through the resulting network. Classic formulations assume all commodities must be transported and aim to minimize total cost. More recent contributions integrate demand selection, revenues, and profits. By contrast, commodity incompatibilities have been studied primarily in operational problems, such as vehicle routing, and have received comparatively little attention in the network design literature. To the best of our knowledge, no existing study addresses the comprehensive problem setting considered in this research. We substantiate this observation by reviewing the literature along two relevant related research streams: profit-oriented network design with facility location, and commodity incompatibility constraints.

2.1 Profit-oriented network design and facility location

Network design problems often involve selecting nodes (facilities) and arcs to construct a network capable of routing multiple commodities. Hub–location (HLP) and hub–network design (HND) models are among the most studied settings. In HLP, a subset of nodes is chosen as hubs to consolidate flows, while HND extends this by selecting specific inter-hub links. Classic HND models assume economies of scale on hub-to-hub links and treat origin/destination nodes as fixed, meaning access-level arcs are not explicitly designed (Contreras, 2021). Some studies relax these assumptions by designing all network layers, including 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 are special cases of MCND when hub locations are fixed. General network design (GND) allows activation of any node, with fixed-charge arcs, requiring joint optimization of facility location, arc activation, routing, and allocation (Contreras and Fernández, 2012). Variants such as transshipment network design (TND) integrate transshipment facility selection with routing (Current, 1988; Current and Pirkul, 1991; Lien et al., 2011), and relay network design (RND) determines relay facilities for vehicle exchanges in long-haul or multimodal systems (Ali et al., 2002; Yıldız et al., 2018). These frameworks highlight that facility location and node selection are strategic decisions interacting strongly with commodity routing under fixed demand.

To align network structures with modern carrier operations, research has shifted toward profit-oriented frameworks where carriers choose which commodities to serve. In service network design (SND), Andersen and Christiansen (2009) studied a railway oper-

ator selecting the most profitable freight contracts under uncertain demand. In MCND, Zetina et al. (2019b) introduced profit-oriented fixed-charge network design with elastic demand, using a gravity model to relate demand to travel time and incorporating service commitments. Huang (2021) and Bilegan et al. (2022) examined revenue management aspects, such as capacity allocation and booking limits, within intermodal freight networks.

Within the HND literature, profit maximizing variants have also attracted substantial attention due to the flexibility they afford carriers to reject low-profit orders. For example, Alibeyg et al. (2014) formulated MIPs for HND with profits, initially restricting the assignment of demand nodes to at most two hubs, a restriction later removed by Taherkhani and Alumur (2019). To capture market dynamics more realistically, Cobeña et al. (2023) modeled demand elasticity in a hub line location problem using a gravity function, evaluating both mixed integer linear and non-linear formulations to highlight the computational challenges inherent to designing hub-line systems under elastic demand.

Despite these advances, simultaneous design of demand and supply decisions has received limited attention; most models optimize network structures and capacities for predetermined demand. In the scheduled service network design (SSND) context, Taherkhani et al. (2022) represents a key exception by coordinating demand and supply sides within a multi-stakeholder freight system. More recent studies integrate pricing and outsourcing. For instance, Fernández et al. (2024) studied a bi-level multi-commodity flow problem in which a primary carrier decides which commodities to transport through a hub network, determines outsourcing fees for non-hub segments, and allocates carriers, while downstream carriers decide whether to accept the assigned shipments to maximize their own profits. Although this framework captures the interactions between multiple decision-makers, it assumes that all commodities can share network resources, leaving commodity-level physical or regulatory incompatibilities unaddressed.

Methodological frameworks for these settings have also evolved to address increasingly complex routing and allocation problems. For instance, Oliveira et al. (2022) developed a Benders decomposition for profit-maximizing multiple-allocation HND, while Oliveira et al. (2023) proposed heuristics for incomplete hub networks. Beyond these, a wide range of exact and heuristic algorithms, 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) have proven effective for various network design and related problems. However, these approaches have typically been applied either to profit-maximizing network design models that assume homogeneous commodities and ignore incompatibilities, or to conflict-constrained flow problems limited to single-commodity flows in fixed networks.

Consequently, there remains no solution approach that simultaneously integrates node selection, demand selection, and incompatibility constraints within a multi-commodity framework. This study aims to bridge these streams by designing both demand and

supply sides of the network while explicitly accounting for unique commodity incompatibilities and maximizing overall profit.

2.2 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. (2024) surveyed multi-commodity vehicle routing problems (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) studied the maximum flow problem with conflicts, in which pairs of conflicting arcs cannot simultaneously carry positive flow. They developed an exact solution method based on Benders decomposition and branch-and-cut. 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 GNDDI 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.

2.3 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. This study fills the gap by integrating all three dimensions.

The GNDDI 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 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, leading to a highly interdependent and combinatorially complex decision problem. The GNDDI 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 GNDDI framework.

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 GNDDI problem consists of selecting a subset of potential arcs $(i, j) \in A$ and a subset of predefined nodes $i \in N$ to activate. 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 indexed 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 $k \in \mathcal{K}$, 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 $r^k d^k$. Also, for each $k \in \mathcal{K}$, there is a unit flow cost $c_{ij}^k \geq 0$ that is associated with each arc $(i, j) \in \mathcal{A}$.

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 imposed 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 GNDDI 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, there needs to be a feasible route that connects the origin and destination of the selected commodities. This feasible route is enabled 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 obtained by selecting the required arcs to be included in the network, and by selecting the proper intermediary nodes to serve as transshipment points.

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 profit gained from satisfying the demands. This 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 the selection of the arcs to be included in the network, and (4) the transportation costs associated with moving the flow of all commodities.

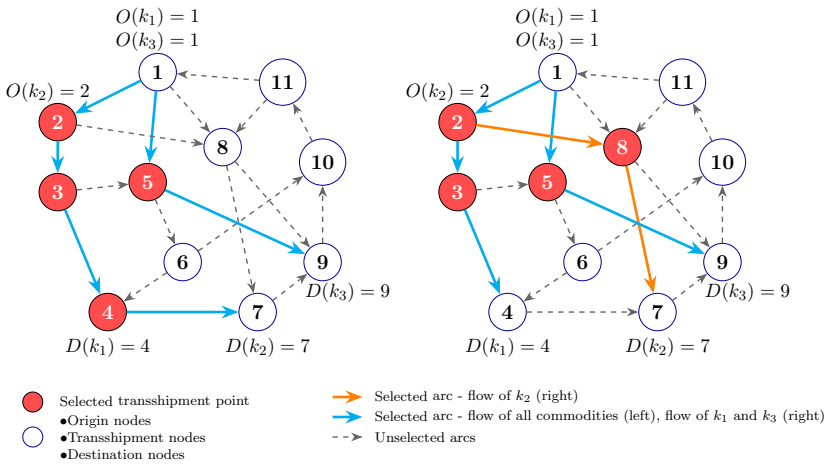


Figure 1: Impact of commodity incompatibility constraints on network flow paths. Prohibiting k_1 and k_2 from sharing arcs alters their routing while no transshipment cost or restriction is considered.

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

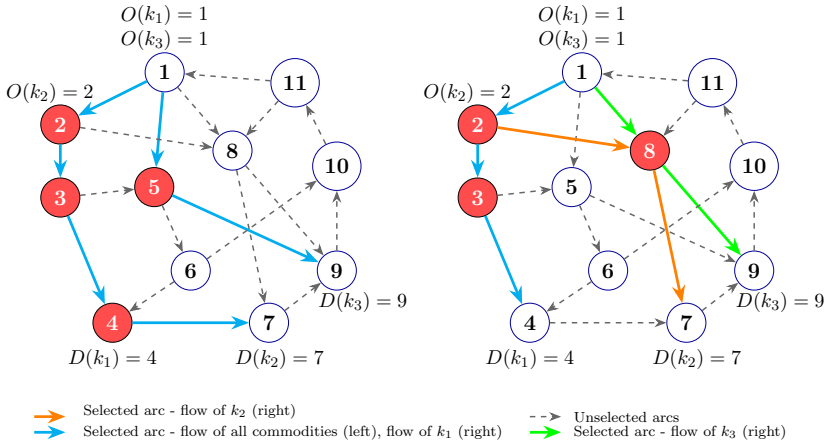


Figure 2: Joint impact of commodity incompatibility constraints and transshipment node selection. Simultaneously enforcing incompatibility restrictions and selecting cost-effective transshipment nodes modifies both network topology and flow patterns.

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 simplified setting in which commodity incompatibilities are considered, but the locations of transshipment facilities are assumed to be fixed (Figure 1, left). In this example, commodities k_1 and k_2 are incompatible, whereas commodity k_3 is unrestricted. As a result, k_1 and k_2 must be assigned different routes, although both are still allowed to pass through the same transshipment node. Their routes therefore become $k_1 : 1 \to 2 \to 3 \to 4$ and $k_2 : 2 \to 8 \to 7$, while k_3 continues to follow the route $1 \to 5 \to 9$. This example illustrates the impact of incompatibility constraints alone. When the fixed costs of opening transshipment facilities are also incorporated into the model, routing decisions are influenced by both incompatibility restrictions and infrastructure investment, leading to the alternative network configuration 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 GNDDI problem. The sets and parameters used in the model are outlined in Table (1).

Table 1: The nomenclature of the GNDDI model

Symbol	Description
Sets	
$\mathcal{N}$	Set of all nodes (candidate transshipment points) $i, j \in \mathcal{N}$.
$\mathcal{A}$	Set of all arcs $(i, j) \in \mathcal{A}$.
$\mathcal{K}$	Set of all commodities $k \in \mathcal{K}$.
$\mathcal{A}_i^+$	Set of outgoing arcs from node $i \in \mathcal{N}$, where $\mathcal{A}_i^+ = \{j \in \mathcal{N} : (i, j) \in \mathcal{A}\}$.
$\mathcal{A}_i^-$	Set of incoming arcs to node $i \in \mathcal{N}$, where $\mathcal{A}_i^- = \{j \in \mathcal{N} : (j, i) \in \mathcal{A}\}$.
λ_k	Set of incompatible commodities with commodity $k \in \mathcal{K}$, where $\lambda_k \in \Lambda$.
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 variable, equals 1 if commodity $k \in \mathcal{K}$ is selected to be served, 0 otherwise.
y_{ij}	A binary variable, equals 1 if arc $(i, j) \in \mathcal{A}$ is selected, 0 otherwise.
z_j	A binary variable, equals 1 if node $j \in \mathcal{N}$ is a transshipment point, 0 otherwise.
x_{ij}^k	Flow quantity of commodity $k \in \mathcal{K}$ on arc $(i, j) \in \mathcal{A}$.
v_{ij}^k	A binary variable, equals 1 if a part of commodity $k \in \mathcal{K}$ is sent via arc $(i, j) \in \mathcal{A}$, 0 otherwise.

The proposed GNDDI model involves five decision variables, designing both the supply and demand side of the network. On the demand side, the commodity 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. The network configuration is represented by the binary variable y_{ij}, which indicates whether arc $(i,j) \in \mathcal{A}$ is included in the network. Similarly, the binary variable z_j specifies whether node $j \in \mathcal{N}$ is selected as a transshipment location. Commodity movements are represented by the continuous variable x_{ij}^k, which sets the flow of commodity $k \in \mathcal{K}$ on arc $(i,j) \in \mathcal{A}$. To enforce the incompatibility requirements specified by Λ, the binary variable v_{ij}^k is introduced to indicate whether any portion of commodity k uses arc (i,j) and zero otherwise, thereby allowing incompatibility restrictions to be imposed directly on arc usage.

GNDDI 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 location and arc 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 assigned to a given arc, any commodity incompatible with it cannot also be assigned to the same arc. 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 GNDDI model. A critical question in the GNDDI 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 GNDDI (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, a central consideration is how the complexity of the GNDDI is distributed between MP and SP. 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, 2021). 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 GNDDI 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 Section S1 of the online supplementary material. 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)-(13), and the SP being a linear program formulated as (14)-(21). We use the generalized Benders decomposition (Geoffrion, 1972), as a result, a generalized optimality cut (11) 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 (12) 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 branch-and-bound (B&B) tree in DS_1 strategy. The MP for DS_1 is formulated as follows:

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

$$s.t.\quad (5), (7), (8)$$

$$Optimality\ Cuts\quad \forall \gamma \in \Gamma_{DS_1} \tag{11}$$

$$Feasibility\ Cuts\quad \forall \omega \in \Omega_{DS_1} \tag{12}$$

$$y_{ij}, w^k, v^k_{ij} \in \{0,1\}\quad \forall (i,j)\in\mathcal{A}, k\in\mathcal{K} \tag{13}$$

The SP for DS_1 is formulated as follows:

$$\max\ Z_{SP_{DS_1}} = -\sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}} c^k_{ij} x^k_{ij} \tag{14}$$

$$s.t.\quad (2), (3), (4), (5), (6)$$

$$y_{ij} = \overline{y}_{ij}\quad \forall (i,j)\in\mathcal{A} \tag{15}$$

$$w_k = \overline{w}_k\quad \forall k\in\mathcal{K} \tag{16}$$

$$z_j = \overline{z}_j\quad \forall j\in\mathcal{N} \tag{17}$$

$$v^k_{ij} = \overline{v}^k_{ij}\quad \forall (i,j)\in\mathcal{A}, k\in\mathcal{K} \tag{18}$$

$$x^k_{ij} \in \mathbb{R}_{\geq 0}\quad \forall (i,j)\in\mathcal{A}, k\in\mathcal{K} \tag{19}$$

$$z_j \in [0,1]\quad \forall j\in\mathcal{N} \tag{20}$$

$$y_{ij}, w^k, v^k_{ij} \in [0,1]\quad \forall (i,j)\in\mathcal{A}, k\in\mathcal{K} \tag{21}$$

Let y_{ij}, w^k, v^k_{ij}, and z_j denote the copy-variables in subproblem SP_{DS_1}, and let $\hat{y}_{ij}$, $\hat{v}^k_{ij}$, $\hat{w}_k$, and $\hat{z}_j$ be their corresponding optimal values obtained by solving SP_{DS_1}. Constraints (15)-(18) are added as a result of GBD, and $\beta^1_{i,j}, \beta^2_k, \beta^3_j, and \beta^4_{ijk}$ represent their dual multipliers respectively. Therefore, the optimality cut for DS_1 is:

$$\theta \leq -\sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}} c^k_{ij}\overline{x}^k_{ij} + \sum_{(i,j)\in\mathcal{A}} \beta^1_{i,j}(y_{i,j}-\hat{y}_{i,j}) + \sum_{k\in\mathcal{K}} \beta^2_k(w_k-\hat{w}_k) + \sum_{j\in\mathcal{N}} \beta^3_j(z_j-\hat{z}_j) + \sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}} \beta^4_{ijk}(v^k_{ij}-\hat{v}^k_{ij}) \tag{22}$$

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^k_{ij}, and z_j denote the copy-variables in subproblem FSP_{DS_1}, and let $\hat{y}_{ij}$, $\hat{v}^k_{ij}$, $\hat{w}_k$, and $\hat{z}_j$ be their corresponding optimal values obtained by solving the following FSP_{DS_1}.

$$\min\ Z_{FSP_{DS_1}} = \sum_{i\in\mathcal{N}}\sum_{\substack{k\in\mathcal{K}\\ i\notin O(k)}}(\epsilon^1_{ik}+\epsilon^2_{ik}) + \sum_{i\in\mathcal{N}}\sum_{\substack{k\in\mathcal{K}\\ i\notin D(k)}}(\epsilon^3_{ik}+\epsilon^4_{ik}) + \sum_{i\in\mathcal{N}}\sum_{\substack{k\in\mathcal{K}\\ i\notin O(k)\\ i\notin D(k)}}(\epsilon^5_{ik}+\epsilon^6_{ik})$$
$$+ \sum_{(i,j)\in\mathcal{A}}\epsilon^7_{ij} + \sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{A}}(\epsilon^8_{ijk}+\epsilon^9_{ijk}) + \sum_{j\in\mathcal{N}}\epsilon^{10}_j \tag{23}$$

$$s.t.\quad \sum_{j\in\mathcal{A}^+_i}x^k_{ij} - \sum_{j\in\mathcal{A}^-_i}x^k_{ji} = \begin{cases} d^kw^k+\epsilon^1_{ik}-\epsilon^2_{ik}, & i=O(k)\\ -d^kw^k+\epsilon^3_{ik}-\epsilon^4_{ik}, & i=D(k)\\ 0+\epsilon^5_{ik}-\epsilon^6_{ik}, & i\neq O(k), i\neq D(k)\end{cases} \quad \forall k\in\mathcal{K}, i\in\mathcal{N} \tag{24}$$

$$\sum_{k\in\mathcal{K}}x^k_{ij} \le u_{ij}y_{ij}+\epsilon^7_{ij} \quad \forall (i,j)\in\mathcal{A} \tag{25}$$

$$x^k_{ij} \le u_{ij}v^k_{ij}+\epsilon^8_{ijk} \quad \forall (i,j)\in\mathcal{A}, k\in\mathcal{K} \tag{26}$$

$$\sum_{k'\in\lambda_k}v^{k'}_{ij} \le |\lambda_k|(1-v^k_{ij})+\epsilon^9_{ijk} \quad \forall (i,j)\in\mathcal{A}, k\in\mathcal{K}, \lambda_k\in\Lambda \tag{27}$$

$$\sum_{\substack{k\in\mathcal{K}:\\ \{O(k)\neq j, D(k)\neq j\}}}\ \sum_{\substack{i\in\mathcal{N}:\\ (i,j)\in\mathcal{A}}}x^k_{ij} \le u_jz_j+\epsilon^{10}_j \quad \forall j\in\mathcal{N} \tag{28}$$

$$y_{ij} = \overline{y}_{ij} \quad \forall (i,j)\in\mathcal{A} \tag{29}$$

$$w_k = \overline{w}_k \quad \forall k\in\mathcal{K} \tag{30}$$

$$z_j = \overline{z}_j \quad \forall j\in\mathcal{N} \tag{31}$$

$$v^k_{ij} = \overline{v}^k_{ij} \quad \forall (i,j)\in\mathcal{A}, k\in\mathcal{K} \tag{32}$$

$$\epsilon^1_{ik},\epsilon^2_{ik},\epsilon^3_{ik},\epsilon^4_{ik},\epsilon^5_{ik},\epsilon^6_{ik},\epsilon^{10}_i \in \mathbb{R}_{\ge 0} \quad \forall k\in\mathcal{K}, i\in\mathcal{N} \tag{33}$$

$$\epsilon^7_{ij},\epsilon^8_{ijk},\epsilon^9_{ijk} \in \mathbb{R}_{\ge 0} \quad \forall (i,j)\in\mathcal{A}, k\in\mathcal{K} \tag{34}$$

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 (29)-(32) respectively. We can then generate the feasibility cut as follows:

$$0 \ge \sum_{i\in\mathcal{N}}\left(\sum_{\substack{k\in\mathcal{K}\\ i\neq O(k)}}(\hat{\epsilon}^1_{ik}+\hat{\epsilon}^2_{ik}) + \sum_{\substack{k\in\mathcal{K}\\ i\neq D(k)}}(\hat{\epsilon}^3_{ik}+\hat{\epsilon}^4_{ik}) + \sum_{\substack{k\in\mathcal{K}\\ i\neq O(k)\\ i\neq D(k)}}(\hat{\epsilon}^5_{ik}+\hat{\epsilon}^6_{ik})\right) + \sum_{(i,j)\in\mathcal{A}}\hat{\epsilon}^7_{ij} + \sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{A}}(\hat{\epsilon}^8_{ijk}+\hat{\epsilon}^9_{ijk})$$
$$+ \sum_{j\in\mathcal{N}}\hat{\epsilon}^{10}_j + \sum_{(i,j)\in\mathcal{A}}\hat{\beta}^1_{ij}(y_{ij}-\hat{y}_{ij}) + \sum_{k\in\mathcal{K}}\hat{\beta}^2_k(w_k-\hat{w}_k) + \sum_{j\in\mathcal{N}}\hat{\beta}^3_j(z_j-\hat{z}_j) + \sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}}\hat{\beta}^4_{ijk}(v^k_{ij}-\hat{v}^k_{ij}) \tag{35}$$

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 (37) 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 (38) 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:

$$\max\ Z_{MP_{DS_2}} = \sum_{k\in\mathcal{K}} r^k d^k w^k - \sum_{j\in\mathcal{N}} f_j z_j - \sum_{(i,j)\in\mathcal{A}} f_{ij} y_{ij} + \theta \tag{36}$$

$$s.t. \quad (7), (8)$$

$$Optimality\ Cuts \quad \forall \gamma \in \Gamma_{DS_2} \tag{37}$$

$$Feasibility\ Cuts \quad \forall \omega \in \Omega_{DS_2} \tag{38}$$

$$y_{ij}, w^k \in \{0,1\} \quad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K} \tag{39}$$

The SP for DS_2 is formulated as follows:

$$\max\ Z_{SP_{DS_2}} = -\sum_{(i,j)\in\mathcal{A}} \sum_{k\in\mathcal{K}} c_{ij}^k x_{ij}^k \tag{40}$$

$$s.t. \quad \sum_{j\in\mathcal{A}_i^+} x_{ij}^k - \sum_{j\in\mathcal{A}_i^-} x_{ji}^k = \begin{cases} d^k\overline{w}^k, & \forall k \in \mathcal{K}, i \in \mathcal{N}, i = O(k) \\ -d^k\overline{w}^k, & \forall k \in \mathcal{K}, i \in \mathcal{N}, i = D(k) \\ 0, & \forall k \in \mathcal{K}, i \in \mathcal{N}, i \neq O(k), i \neq D(k) \end{cases} \tag{41}$$

$$\sum_{k\in\mathcal{K}} x_{ij}^k \leq u_{ij}\overline{y}_{ij} \quad \forall (i,j) \in \mathcal{A} \tag{42}$$

$$x_{ij}^k \leq u_{ij} v_{ij}^k \quad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K} \tag{43}$$

$$\sum_{k'\in\lambda_k} v_{ij}^{k'} \leq |\lambda_k|(1 - v_{ij}^k) \quad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K}, \lambda_k \in \Lambda \tag{44}$$

$$\sum_{\substack{k\in\mathcal{K}: \\ \{O(k)\neq j, D(k)\neq j\}}} \sum_{\substack{i\in\mathcal{N}: \\ (i,j)\in\mathcal{A}}} x_{ij}^k \leq u_j \overline{z}_j \quad \forall j \in \mathcal{N} \tag{45}$$

$$v_{ij}^k \in \{0,1\},\ x_{ij}^k \in \mathbb{R}_{\geq 0} \quad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K} \tag{46}$$

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

is found.

$$\theta \leq (Z_{\mathrm{SP}_{DS_2}} - UB_{\mathrm{SP}_{DS_2}})\Big(\sum_{(i,j)\in\eta_y} y_{ij} - \sum_{(i,j)\notin\eta_y} y_{ij} + \sum_{k\in\eta_w} w_k - \sum_{k\notin\eta_w} w_k + \sum_{j\in\eta_z} z_j - \sum_{j\notin\eta_z} z_j - (|\eta_y| + |\eta_w| + |\eta_z|)\Big) + Z_{\mathrm{SP}_{DS_2}} \quad (47)$$

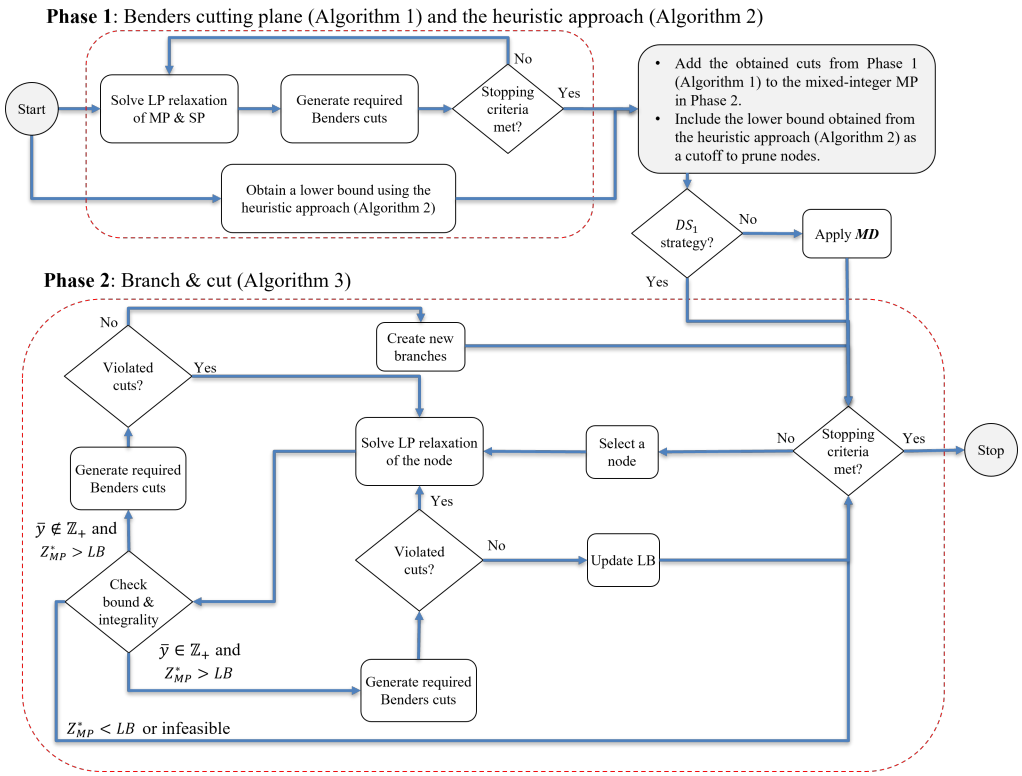
In this context, $UB_{\mathrm{SP}_{DS_2}}$ denotes an upper bound on $Z_{\mathrm{SP}_{DS_2}}$. Furthermore, $\eta_y = \{(i,j) \in \mathcal{A} : y_{ij} = 1\}$ is the set of arcs where the variable y_{ij} takes the value 1 for that integer feasible solution. The sets η_w and η_z are defined similarly. 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.

$$\sum_{(i,j)\in\eta_y} (1 - y_{i,j}) + \sum_{(i,j)\notin\eta_y} y_{i,j} + \sum_{k\in\eta_w} (1 - w_k) + \sum_{k\notin\eta_w} w_k + \sum_{j\in\eta_z} (1 - z_j) + \sum_{j\notin\eta_z} z_j \geq 1 \quad (48)$$

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 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 and its implementation details are provided in Sections S4 and S5 of the online supplementary material. 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 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.

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 linear programming relaxation (LPR), the best lower-bound, and an optimal solution to the problem, respectively. Algorithms 1–3 are provided in Section S4 of the online supplementary material.

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 GNDDI 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 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

The modified decomposition (MD) approach, also referred to as the nonstandard decomposition strategy (Gendron et al., 2016; Rahmaniani et al., 2017), strengthens the MP by explicitly incorporating a copy of the SP's linear relaxation into the MP formulation. We apply this approach to DS_2, DS_3, and DS_4, but not to DS_1. For DS_1, the SP is a linear program, and hence its linear relaxation coincides with the SP itself. Incorporating the complete SP formulation into the MP would therefore transfer the entire SP structure to the MP and effectively eliminate the need for decomposition. In contrast, for DS_2-DS_4, incorporating only the SP's linear relaxation only transfers a partial structural information to the MP. This technique has been shown to accelerate the Benders decomposition algorithm (Gendron et al., 2016), and is therefore adopted to improve the computational efficiency of the proposed 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:

$$\theta \leq Z_{SP} \tag{49}$$

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).

5.3.2 Benders dual cuts

Benders dual cuts were first introduced by Rahmaniani et al. (2020). This approach proceeds by constructing a new MIP model for the SP, where a local copy of the MP variables is reintroduced into the new MIP sub-problem to project those in the MP with equality constraints. We call this reformulated problem the *sub-problem with projections* (SPP). Next, a Lagrangian relaxation is applied to these equality constraints, using the dual values obtained at the current solution, thereby enabling the exploration of the MP's original decision space. 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 current MP's solution. 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:

$$\text{Max} \quad Z_{SPP_{DS_2}} = -\sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}} c_{ij}^k x_{ij}^k \tag{50}$$

$$s.t.\ (2)-(7),\ (15)-(17),\ (19)-(21)$$

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 (15)-(17) respectively. Therefore, the generalized optimality cut for DS_2 is as follows:

$$\theta \leq -\sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}} c_{ij}^k \hat{x}_{ij}^k + \sum_{(i,j)\in\mathcal{A}} \beta_{i,j}^1 (y_{i,j} - \hat{y}_{i,j}) + \sum_{k\in\mathcal{K}} \beta_k^2 (w_k - \hat{w}_k) + \sum_{j\in\mathcal{N}} \beta_j^3 (z_j - \hat{z}_j) \tag{51}$$

After solving SPP_{DS_2} we can then solve the following SPL_{DS_2} to generate the strength-

ened optimality cut (SOC) as follows (53).

$$\text{Max} \quad Z_{SPL_{DS_2}} = -\sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}} c_{ij}^k x_{ij}^k - \sum_{(i,j)\in\mathcal{A}} \beta_{i,j}^1\big(y_{i,j}-\overline{y}_{i,j}\big) - \sum_{k\in\mathcal{K}} \beta_k^2\big(w_k-\overline{w}_k\big) - \sum_{j\in\mathcal{N}} \beta_j^3\big(z_j-\overline{z}_j\big) \tag{52}$$
$$s.t. (2)-(9)$$

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:

$$\theta \leq -\sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}} c_{ij}^k \tilde{x}_{ij}^k + \sum_{(i,j)\in\mathcal{A}} \beta_{i,j}^1(y_{i,j}-\tilde{y}_{i,j}) + \sum_{k\in\mathcal{K}} \beta_k^2(w_k-\tilde{w}_k) + \sum_{j\in\mathcal{N}} \beta_j^3(z_j-\tilde{z}_j) \tag{53}$$

Similar to the optimality cuts, generalized and strengthened feasibility cuts can also be derived for DS_2, as presented in Section S2 of the online supplementary material. Consequently, each DS requires solving two subproblems to generate strengthened optimality cuts, namely SPP and SPL, and two subproblems to generate strengthened feasibility cuts (SFC), namely 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:

$$\begin{cases} Cy_{ij} = (1-\zeta)Cy_{ij} + \zeta\overline{y}_{ij} \\ Cw_k = (1-\zeta)Cw_k + \zeta\overline{w}_k \\ Cz_j = (1-\zeta)Cz_j + \zeta\overline{z}_j \end{cases} \tag{54}$$

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 GNDDI, 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 (54). Since commodity selections may not always satisfy incompatibility constraints, Cw_k is obtained from SPP to ensure PSP feasibility.

5.3.4 Valid inequalities

We derive a series of valid inequalities to further strengthen the MP and tighten its linear relaxation, thereby improving 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, 2021). 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 (55), directly links the activation of an arc to the flow that can be routed through it.

$$x_{ij}^k \leq b_{ij}^k y_{ij} \qquad \forall (i,j) \in \mathcal{A},\ k \in \mathcal{K} \tag{55}$$

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 $(\ell,\ \overline{\ell}) = \{(i,j) \in \mathcal{A} : i \in \ell,\ j \in \overline{\ell}\}$.

$$\sum_{(i,j)\in(\ell,\overline{\ell})} u_{ij} y_{ij} \geq \sum_{k\in\mathcal{K}(\ell,\overline{\ell})} d^k w^k \qquad \forall\, \ell \subseteq \mathcal{N},\ \overline{\ell} \neq \emptyset \tag{56}$$

where $\mathcal{K}(\ell,\overline{\ell}) = \{k \in \mathcal{K} : O(k) \in \ell, D(k) \in \overline{\ell}\}$. According to (Crainic and Gendron, 2021), 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 GNDDI 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 (57) and (58) represent the linking constraints between defining the demand-side and designing the supply-side. In the GNDDI 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 (57) ensures that once a commodity is selected to be satisfied, at least one outgoing arc from the origin node must be opened. Equation (58) ensures that once a commodity is selected to be satisfied, at least one incoming arc toward the destination node must be opened.

$$w^k \leq \sum_{(i,j)\in\mathcal{A}|i=O(k)} y_{ij} \qquad \forall k \in \mathcal{K} \tag{57}$$

$$w^k \leq \sum_{(i,j)\in\mathcal{A}|j=D(k)} y_{ij} \qquad \forall k \in \mathcal{K} \tag{58}$$

Equations (59) and (60) establish the connection between transshipment-node selection and arc-opening decisions. In the GNDDI model, this dependency was previously implied through the flow variables linking z_j and y_{ij}. Constraint (59) 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 (60) 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.

$$z_j \leq \sum_{i\in\mathcal{A}_j^-} y_{ij} \qquad \forall j \in \mathcal{N} : \exists k \in \mathcal{K} \text{ such that } j \neq D(k) \tag{59}$$

$$z_i \leq \sum_{j\in\mathcal{A}_i^+} y_{ij} \qquad \forall i \in \mathcal{N} : \exists k \in \mathcal{K} \text{ such that } i \neq O(k) \tag{60}$$

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:

$$v_{ij}^k \leq y_{ij} \qquad \forall (i,j) \in \mathcal{A},\ k \in \mathcal{K} \tag{61}$$

$$v_{ij}^k \leq w^k \qquad \forall (i,j) \in \mathcal{A},\ k \in \mathcal{K} \tag{62}$$

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:

$$w^k \leq \sum_{\substack{j\in\mathcal{N}:\\ j\neq O(k),\, j\neq D(k)}} z_j \;+\; y_{(O(k),D(k))} \qquad \forall k \in \mathcal{K} \tag{63}$$

In addition, we observe that considering the transshipment inequality (6) for the outgoing arcs, as illustrated in (64), as well as tightening the right-hand side (RHS) of the equations by replacing u_i with $\rho_i = \min\{u_i, \sum_{j\in\mathcal{A}_i^+} u_{ij}\} \quad \forall i \in \mathcal{N} : (i,j) \in \mathcal{A}$, accelerates the algorithm.

$$\sum_{\substack{k\in\mathcal{K}:\\ O(k)\neq i,\, D(k)\neq i}} \sum_{j\in\mathcal{A}_i^+} x_{ij}^k \leq \rho_i z_i \qquad \forall i \in \mathcal{N} \tag{64}$$

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 $b_{ij}^k = \min\{d^k, u_{ij}\}$.

Maximal incompatible clique cuts. A key challenge of the GNDDI 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

$$v_{ij}^{k'} \leq 1 - v_{ij}^k \qquad \forall (i,j) \in \mathcal{A},\ k \in \mathcal{K},\ \lambda_k \in \Lambda,\ k' \in \lambda_k \tag{65}$$

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 (65) 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 GNDDI model:*

$$\sum_{k\in\mathcal{C}_l} v_{ij}^k \leq 1 \qquad \forall (i,j) \in \mathcal{A},\ \mathcal{C}_l \in \mathcal{C} \tag{66}$$

Proof. See Appendix A. □

This inequality is added to the SP to improve the quality of the duals passed to the MP, and due to the modified decomposition, 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 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 routing cost of the SP is at least the shortest-path cost, hence the SP objective (the negative of cost). 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.

$$\boldsymbol{DS_1}\&\boldsymbol{DS_2}: \quad \theta \leq -\sum_{k\in\mathcal{K}} C^k_{P(O(k),D(k))} d^k w^k \tag{67}$$

Although (49) 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., (47)) at each integer node by deriving the information of UB_{SP} from UBF. The UBF for DS_3 and DS_4 are presented in Section S3 of the online supplementary material.

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 of the online supplementary material.

5.3.6 Lower bounding with a compatible solution

A key challenge in the GNDDI 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 (GH) to tackle the GNDDI. 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 of the online supplementary material.

6 Results and analysis

We report the results of an extensive set of computational experiments, which seek to attain three primary objectives. First, to assess the value of explicitly accounting for commodity incompatibility and transshipment point selection in the joint design of the supply and demand sides of the network. Second, to analyze the computational efficiency of decomposition strategies for the proposed GNDDI formulation and identify the key decisions that effectively guide the search process within the BD framework. Third, to evaluate the efficiency of the proposed B&BC algorithm and examine the performance

of the tailored acceleration strategies under varying supply-side and demand-side characteristics.

All algorithms are implemented in Python 3.12 and solved using the Gurobi Optimizer (version 11). To ensure a fair comparison, Gurobi is restricted to a single computational thread. All experiments are conducted on a Linux server with a 2.67 GHz processor and 32 GB of RAM. All variants of the B&BC algorithm are benchmarked against the GNDDI model solved directly using Gurobi.

We begin by describing the instance sets used to validate the model and solution approach (Section 6.1). We then compare the computational efficiency of the proposed decomposition strategies in Section 6.2 and evaluate the impact of incorporating incompatibility requirements and transshipment point decisions in Section 6.3. Finally, we investigate the effectiveness of the developed acceleration strategies and evaluate the performance of the B&BC algorithm under varying network characteristics in Section 6.4.

6.1 Data and experimental settings

Our computational experiments are conducted on the classic R (Reserve) and C (Canad) instances, which serve as standard benchmarks in the multicommodity network design literature (Larsen et al., 2023; CommaLAB, 2023). The R instances correspond to network configurations with 10 or 20 nodes and are denoted by $rx.y$, where x identifies the underlying network structure in terms of nodes, arcs, and commodities. Instances with the same x share the same network topology. The index y specifies the fixed and variable design costs as well as the arc capacities. In our experiments, we use 135 R instances with $x \in \{04, \ldots, 18\}$ and $y \in \{1, \ldots, 9\}$. The C instances correspond to network configurations with 20 or 30 nodes. As with the R instances, they differ in fixed and variable costs and in capacity settings. However, unlike the R instances, the C instances do not necessarily share the same network topology across different cost and capacity settings. We use all 31 available C instances. 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

Parameter	Value
c_{ij}^k	R& C Instance Data Set
r_k	$U \sim \left(\left\lfloor 10 \times \min_{(i,j)\in\mathcal{A}}\{c_{ij}^k\}\right\rfloor, \left\lfloor 20 \times \max_{(i,j)\in\mathcal{A}}\{c_{ij}^k\}\right\rfloor\right)$
u_{ij}	R& C Instance Data Set
u_j	$U \sim \Big(m(j)\min\{u_{ij} : i \in \mathcal{A}_j^+ \cup \mathcal{A}_j^-\},\, m(j)\max\{u_{ij} : i \in \mathcal{A}_j^+ \cup \mathcal{A}_j^-\}\Big)$ *where* $m(j) = \max\{\lvert\mathcal{A}_j^+\rvert, \lvert\mathcal{A}_j^-\rvert\}$
f_{ij}	R& C Instance Data Set
f_j	$U \sim \left(2 \times \min_{j\in\mathcal{N}\mid(i,j)\in\mathcal{A}}\{f_{ij}\}, 2 \times \max_{j\in\mathcal{N}\mid(i,j)\in\mathcal{A}}\{f_{ij}\}\right)$
d^k	R& C Instance Data Set

6.2 Evaluating decomposition approaches

This section presents a comparative analysis of the decomposition strategies. 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 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 shown in Table 3, DS_1 exhibits the poorest performance, as all discrete variables are in the MP, leading to limited guidance during the search. Although the MP of DS_4 contains fewer discrete variables, these variables provide minimal informative value for effectively guiding the solution process. A comparison between DS_2 and DS_3 reveals that, while DS_3 incorporates decomposition along both supply and demand sides, the explicit selection of demand in the MP of DS_2 plays a critical role in guiding MP, thereby facilitating more effective supply-side decisions. Overall, DS_2 consistently outperforms the other decomposition strategies in terms of solution time, optimality gap, and D_{ratio}. Consequently, we adopt this decomposition strategy for the subsequent analysis of 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	1101.5	0.0%	8/9	6171.9	3.5%	2/9	7200	91.3%	0/9
r12	4288.9	1.5%	7200	100%	0/9	4075.1	1.2%	5/9	6569.1	8.3%	1/9	7200	96.7%	0/9
r13	4428.5	0.3%	7200	100%	0/9	1259.2	0.0%	9/9	3522.5	1.8%	5/9	7200	67.9%	0/9
r14	5404.4	1.5%	7200	100%	0/9	4109.5	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	635.9	0.0%	9/9	2129.9	0.0%	7/9	7200	67.4%	0/9
r17	7200	1.9%	7200	100%	0/9	4632.2	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%	2094.4	0.7%	86.7%	3597.2	2.1%	58.5%	7200	56.3%	0.0%

6.3 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, both solved directly using Gurobi. 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 GNDDI 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, taking the first 50, 100 and 200 commodities. 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.

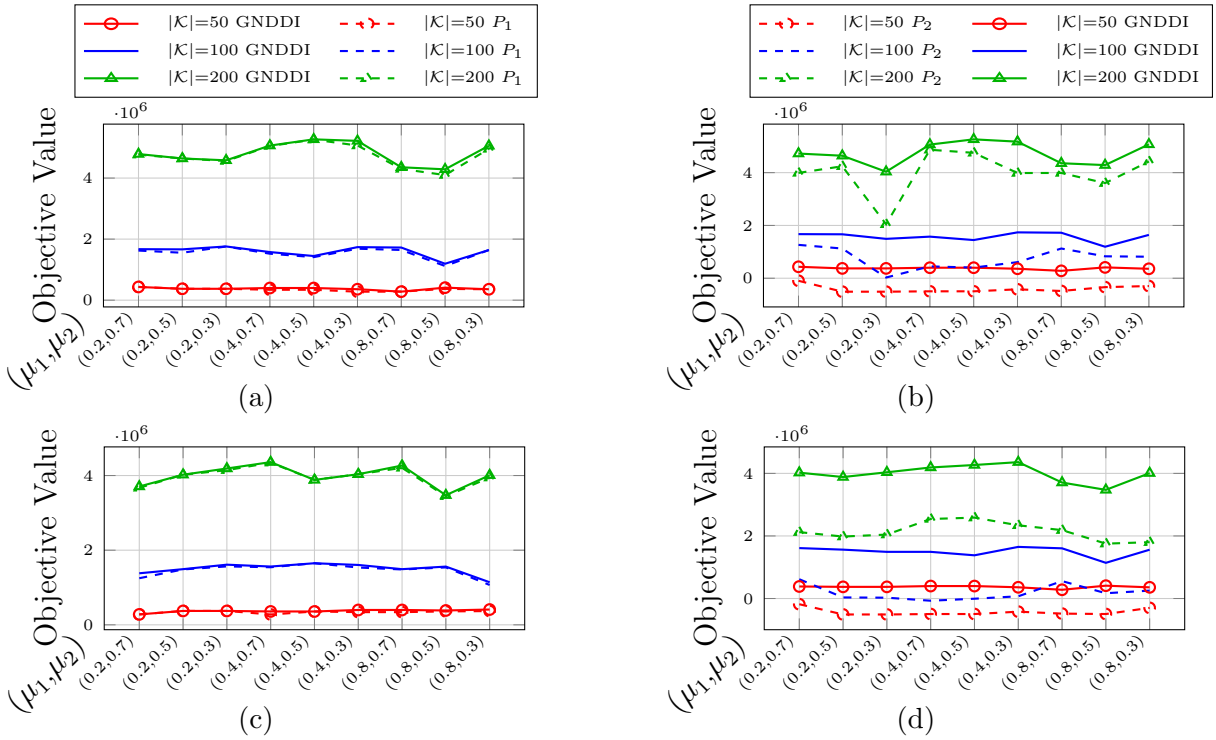


Figure 4: Comparing total profit in P_1 ((4a),(4c)) and P_2 ((4b),(4d)) vs $GNDDI$ for different $|\mathcal{K}|$ and (μ_1, μ_2).

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 excluding (49) 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 (49) significantly reduces the algorithm's ability to find feasible solutions.

Table 4: Evaluating the effect of MD

R	*B&BC*			*B&BC*-(49)			*B&BC-MD*		
	Time(s)	Gap	*#Sol*	Time(s)	Gap	*#Sol*	Time(s)	Gap	*#Sol*
r04	0.8	0.0%	9/9	2817	14.2%	8/9	3600	100%	0/9
r05	17.9	0.0%	9/9	3600	16.8%	8/9	3600	100%	0/9
r06	35.5	0.1%	9/9	3600	7.4%	9/9	3600	100%	0/9
r07	401.4	0.0%	9/9	3600	57.9%	4/9	3600	100%	0/9
r08	15.9	0.0%	9/9	3600	25.6%	7/9	3600	100%	0/9
r09	40.0	0.0%	9/9	3600	18.5%	8/9	3600	100%	0/9
r10	1239.8	0.2%	9/9	3600	45.0%	6/9	3600	100%	0/9
Average	250.2	0.0%	100%	3488.2	26.5%	79.4%	3600	100%	0.0%

Unlike MD, and (49), which show a noticeable impact even on small instances, the value of the remaining acceleration strategies becomes 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) $B\&BC\text{-}GH$, where the greedy heuristic is removed, preventing the use of its initial lower bound in the second phase, and (2) $B\&BC\text{-}WS$, where the first phase is omitted. As illustrated, for $T = 0.5|\mathcal{N}|$, dropping either of these two components from the B&BC worsens the average performance in terms of Time, Gap, and D_B. The effect is not uniform across instances (e.g., on C53-C56 both variants attain a higher D_B than $B\&BC$), and its dependence on T is examined in Figure 5.

Since there are different classes of valid inequalities (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 (57), (58), b) VI_2, which consists of (59), (60), c) VI_3, which consists of (61), (62), d) VI_4, represents (63), e) VI_5, represents (64), f) network design based VIs (ND), g) maximal incompatible clique cuts (MIC), and h) upper bounding function (UBF). The results of this analysis are presented in Table S2 of the online supplementary material. As illustrated, for $T = 0.5|\mathcal{N}|$, removing any single VI increases the average computation time and lowers D_B compared with the full B&BC; the Gap also increases, except for VI_5, whose removal leaves it unchanged. Among these VIs, those whose removal from the algorithm results in a higher Gap and a lower D_B are considered the most impactful. 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: $MIC > VI_3 > UBF > VI_1 > VI_2 > ND > VI_4 > VI_5$, where the differences among the lower-ranked VIs are small. The MIC directly exploits the interdiction relationships, which themselves strongly shape the commodity flows. Including these inequalities therefore provides substantial information about the combinatorial restrictions governing the use of the network. By narrowing the

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	30	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%

space of viable flows in advance, MIC implicitly signals which designs are worth building. Similarly, VI_3 links the flow variables to both the commodity-selection and the arc-design variables, signaling which commodities to select and which designs are worth building. This dual influence on both flow and design decisions explains why removing VI_3 or MIC causes the largest deterioration in Gap and D_B among all tested VIs. The UBF bounds the routing cost of the SP through the shortest-path cost of each commodity, expressed in terms of the commodity-selection variables, and therefore provides early information on which commodities are worth serving while also strengthening the integer optimality cuts. Since it is built on shortest paths, it carries no information on capacities or incompatibilities and thus acts less on flow feasibility than MIC and VI_3. Although VI_1 attains a slightly lower D_B than UBF (62.5% vs. 63.5%), UBF yields the higher Gap (0.75% vs. 0.72%) and is therefore ranked higher according to our tie-breaking rule. By contrast, VI_2 and ND strengthen only the design-side variables, and VI_1 links commodity selection to the design only at the origin and destination nodes, leaving flow feasibility to be resolved afterward. Finally, VI_4 and VI_5, which only tighten the use of transshipment points, have the smallest effect.

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 four cases: (1) excluding only feasibility cuts, which is the current approach ($B\&BC$), (2) excluding feasibility and optimality cuts ($B\&BC$-($OC\&FC$)), (3) maintaining both feasibility and optimality cuts ($B\&BC$+FC), and the (4) effect of strengthening the SOC cuts by deriving the duals from PSP instead of SPP ($B\&BC$-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,

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	16593.6	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%	2/4	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%

for $T = 0.5|\mathcal{N}|$, keeping the optimality cuts and omitting the feasibility cuts, as in the B&BC, yields the best overall performance.

The performance of the algorithm for small-, medium-, and large-scale instances is summarized in Table S1 of the online supplementary material. 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 GNDDI 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, two components of the B&BC algorithm. Figure 5 shows that their value depends strongly on T. Each point on the plot represents the average over the 96 large-scale R & C instances. For $T = 0.5|\mathcal{N}|$, excluding WS or GH increases the average time by about 1350 and 1280 seconds, respectively. Additionally, excluding WS reduces D_B from 72.9% to 64.6% and increases the Gap from 0.58% to 0.74%, while excluding GH reduces D_B to 67.7% and increases the Gap to 0.62%. For $T = 0.3|\mathcal{N}|$, both components again reduce the time by about 880 and 660 seconds respectively. In addition, excluding WS lowers D_B from 64.6% to 58.3% at an unchanged Gap, whereas excluding GH leaves D_B

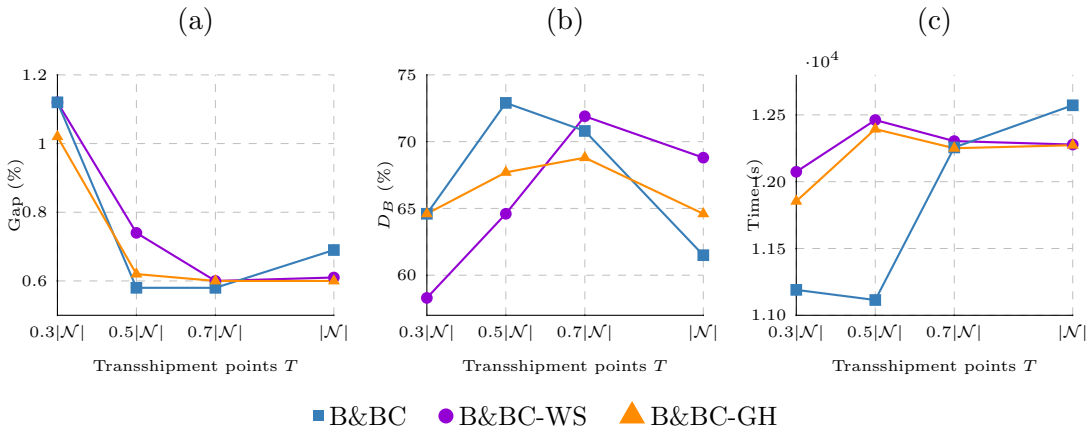


Figure 5: Analyzing the effect of WS and GH on the overall performance of the B&BC: (a) Gap, (b) D_B, and (c) Time (s) across different transshipment point configurations

unchanged and slightly lowers the Gap (1.02% vs. 1.12%). For $T = 0.7|\mathcal{N}|$, the three variants are practically indistinguishable. For $T = |\mathcal{N}|$ the effect reverses. *B&BC-WS* and *B&BC-GH* are slightly faster (about 300 seconds), have a lower Gap (0.61% and 0.60% vs. 0.69%), and a higher D_B (68.8% and 64.6% vs. 61.5%).

Additionally, we evaluate the quality of the 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 29.76%. Considering that the GH approach (Algorithm 2 of the online supplementary material) solves large-sized instances (R & C) in less than one second (approximately 0.2 sec on average), compared to an average of approximately 6597.9 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 2 shows a 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 2 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 $\frac{f_{ij}}{c^k_{ij}}$ ratio being lower than in r12.9.

Figure 7 shows the effect of the cuts derived from the LPR of the SP for different values of T. Adding the feasibility cuts (*B&BC+FC*) worsens the performance of the algorithm when $T \leq 0.7|\mathcal{N}|$ and yields only a marginal improvement for D_B and Time at $T = |\mathcal{N}|$. For $T = 0.5|\mathcal{N}|$, for instance, the Gap increases from 0.58% to 0.79%,

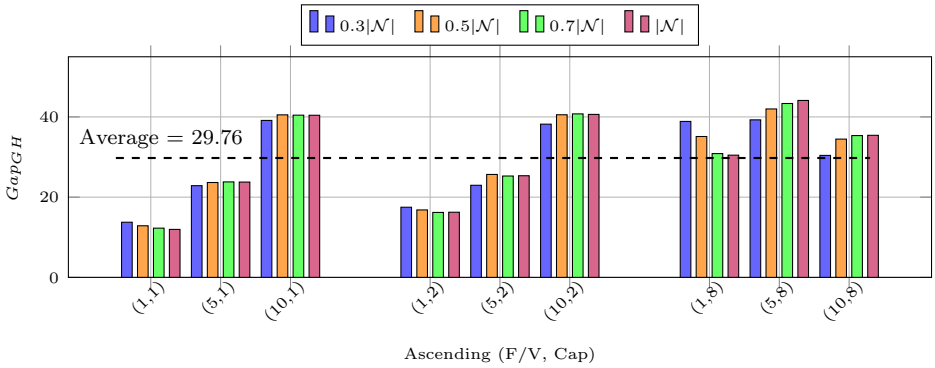


Figure 6: Best integer vs. heuristic solutions for varying capacities and F/V ratios.

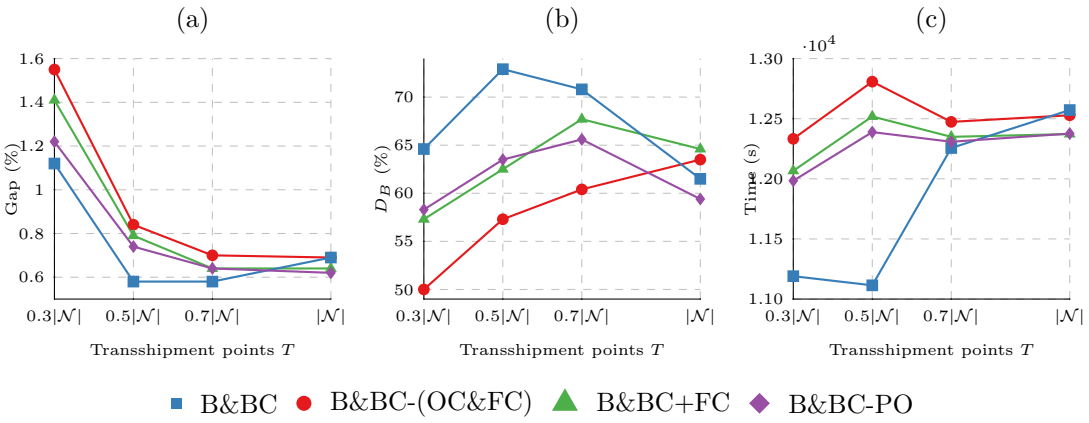


Figure 7: Evaluating the impact of individual components on the performance of the B&BC for different $|T|$ by comparing their (a) Gap, (b) D_B, and (c) Time (s).

D_B falls from 72.9% to 62.5%, and the average time increases by about 1400 seconds. The difference between $B\&BC$ and $B\&BC$-$(OC\&FC)$ measures the contribution of the optimality cuts. This contribution is largest for small T. At $T = 0.3|\mathcal{N}|$ and $0.5|\mathcal{N}|$, removing the optimality cuts raises the Gap by 0.43 and 0.26 percentage points and lowers D_B by 14.6 and 15.6 points, respectively. It decreases as T grows (at $T = 0.7|\mathcal{N}|$ the increase in Gap is 0.12 and the decrease in D_B is 10.4) and disappears at $T = |\mathcal{N}|$, where both $B\&BC$-$(OC\&FC)$ and $B\&BC$+FC perform as well as or slightly better than $B\&BC$. The Pareto-optimal cuts follow the same pattern, with the largest effect at $T = 0.5|\mathcal{N}|$.

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 96 large-scale R&C instances. For $T = 0.3|\mathcal{N}|$ and $0.5|\mathcal{N}|$, all proposed VIs improve the

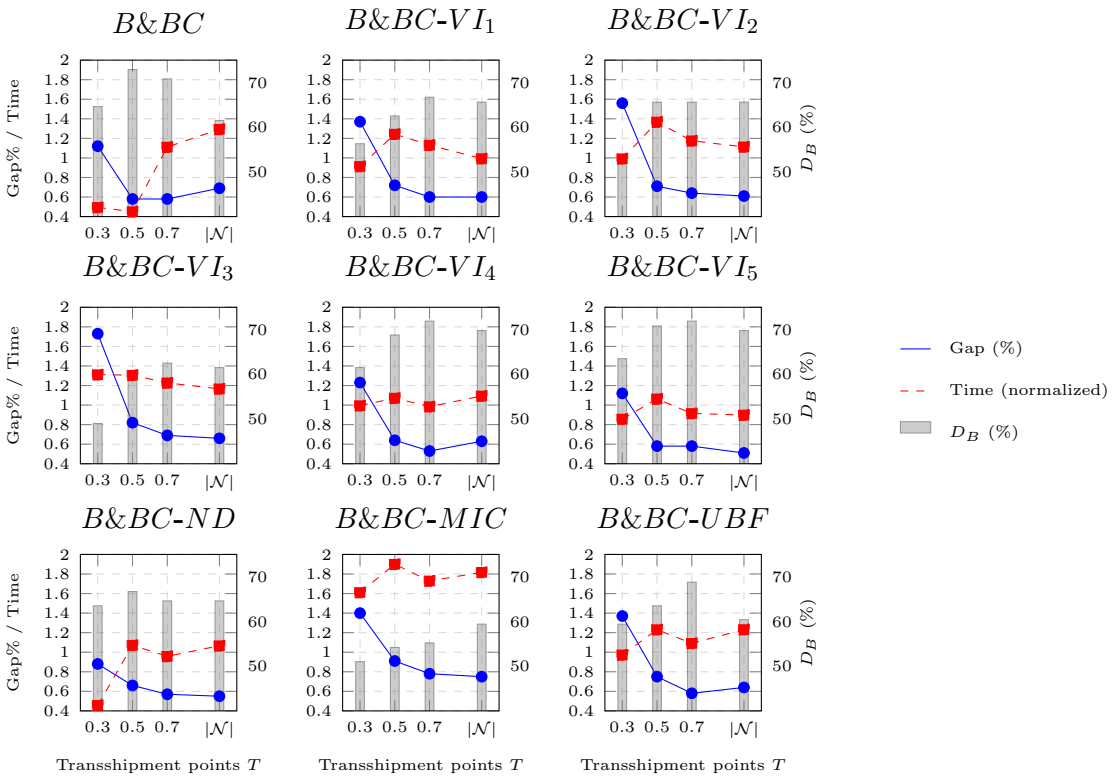
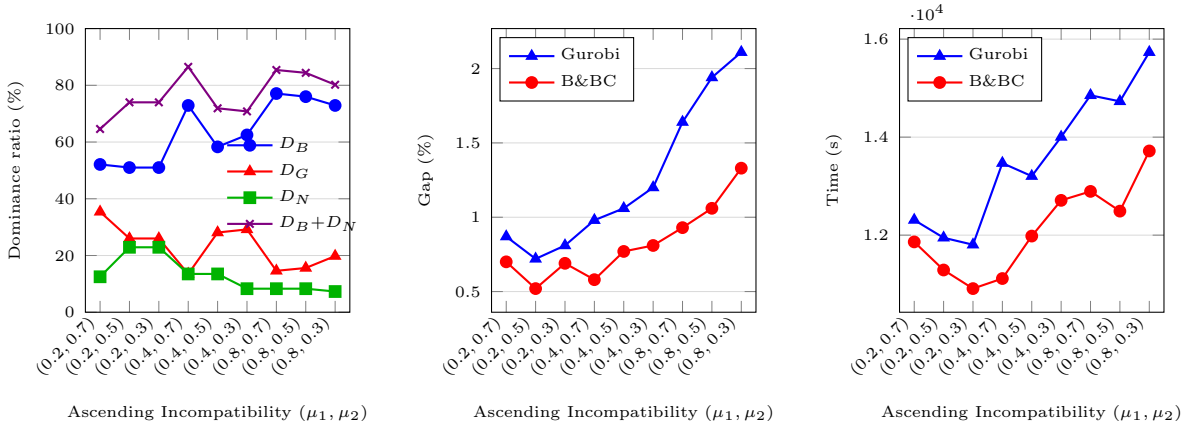


Figure 8: Analyzing the impact of individual VI on large R and C instances using performance metrics: Gap%, Time (normalized), and Dominance (D_B%).

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 18.5% and 13.5%, respectively. In contrast, excluding VI_4 and VI_5 increases D_B by 0.8% and 2.3% 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, across all T, excluding these VIs results in a 31.3%, and 29.7% increase in Gap, respectively. In contrast, excluding ND and VI_5 decreases 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, across all T, 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.



(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).

6.4.3 Evaluating the effect of incompatibility

A second factor influencing the computational complexity of the GNDDI 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 (66) 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, 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 96 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 incorporat-

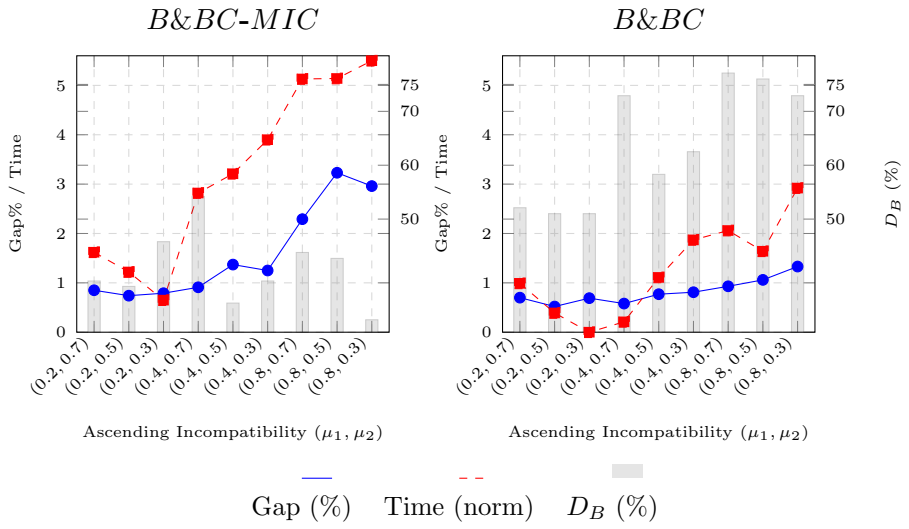


Figure 10: Evaluating the impact of MIC inequality on large R and C instances when $|\lambda_k|$ rises, using Gap%, normalized Time, and D_B.

ing MIC ($B\&BC\text{-}MIC$), whereas the right plot includes them ($B\&BC$). When MIC is considered, the Gap stays between 0.52% and 1.33% across all (μ_1, μ_2) and exceeds 1.1% only at $(0.8, 0.3)$. The average computational time grows roughly from 10,900 s at $(0.2, 0.3)$ to 13,700 s at $(0.8, 0.3)$, although not monotonically, which indicates stable performance despite increasing incompatibility. Moreover, the dominance measure D_B (gray bars) remains above 50% in all nine settings and is higher for larger μ_1, reflecting that the majority of instances are strictly dominated compared to Gurobi. This overall rise in 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 average computational time increases from roughly 11,500 s to 16,200 s, with a considerable reduction in the dominance measure D_B (from about 46% to 31%).

These results are consistent with the intended role of the MIC inequality, which specifically exploits commodity incompatibilities; therefore, its contribution becomes increasingly relevant as the number of incompatibilities grows. Accordingly, excluding MIC from the $B\&BC$ algorithm leads to even larger optimality gaps and longer computational times under high-incompatibility settings.

7 Conclusions

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 Generalized Network Design with Demand selection and Incompatible commodities (GNDDI) 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 further 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 GNDDI. Extensive computational experiments on benchmark instances show that the proposed algorithm performs reliably across varied network structures and incompatibility settings, surpassing commercial solvers like Gurobi on large size instances. 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 72.9% of the test instances.

The proposed framework lays the groundwork for several future research directions. In particular, explicitly accounting for the uncertainty in demand-side requirements represents a relevant avenue for future research. Another direction is based on a key structural assumption in the GNDDI, that incompatible commodities cannot share the same arc capacity. However, in transportation settings, regulatory frameworks might permit the co-loading of certain incompatible goods if they are physically separated by compartments, bulkheads, or minimum distances. Future research could extend this study by introducing capacity compartmentalization. This idea could also be very well extended to asset-management based network design frameworks. Finally, while the current study assumes a fixed revenue parameter r^k for each commodity and enforces full-demand fulfillment (binary accept/reject decisions). Future research could integrate dynamic pricing models or elastic demand functions to coordinate pricing strategies with capacity allocation and incompatibility restrictions.

Data availability

Most of the data used are from standard benchmarks in the literature and rest are explained in the text. The benchmark instances and computational data associated with this research will be made available upon reasonable request.

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.

CommaLAB. The Mulgen generator for Canad and Reserve instances. `https://commalab.di.unipi.it/datasets/mmcf`, 2023. Accessed: 2026-06-20.

I. Contreras. Hub Network Design. In T.G. Crainic, M. Gendreau, and B. Gendron, editors, *Network Design with Applications to Transportation and Logistics*, chapter 18, pages 567–598. Springer, Boston, 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 T.G. Crainic, M. Gendreau, and B. Gendron, editors, *Network Design with Applications to Transportation and Logistics*, chapter 3, pages 29–89. Springer, Boston, 2021.

T.G. Crainic and M. Hewitt. Service Network Design. In T.G. Crainic, M. Gendreau, and B. Gendron, editors, *Network Design with Applications to Transportation and Logistics*, chapter 12, pages 347–382. Springer, Boston, 2021.

T.G. Crainic, M. Gendreau, and B. Gendron. Fixed-Charge Network Design Problems. In T.G. Crainic, M. Gendreau, and B. Gendron, editors, *Network Design with Applications to Transportation and Logistics*, chapter 2, pages 15–28. Springer, Boston, 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. Fernández, I. Ljubić, 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 B. Sansò and P. Soriano, editors, *Telecommunications Network Planning*, pages 1–19. Kluwer Academic Publishers, Boston, 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*, 317(1):1–15, 2024.

M. Hewitt and W. Rei. Perspectives on Using Benders Decomposition to Solve Two-Stage Stochastic Mixed-Integer Programs. In T.G. Crainic, M. Gendreau, and A. Frangioni, editors, *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.

E. Larsen, S. Bisaillon, J.-F. Cordeau, and E. Frejinger. Pseudo-random instance generators in C++ for deterministic and stochastic multi-commodity network design problems. *arXiv preprint arXiv:2312.10717*, 2023.

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.

G. Taherkhani, I. C. Bilegan, T. G. Crainic, M. Gendreau, and W. Rei. Tactical capacity planning in an integrated multi-stakeholder freight transportation system. *Omega*, 110: 102628, 2022.

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.

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.

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.

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 Proof of Proposition 1.

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_{ij}^k 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 (65), therefore cliques found for $|\mathcal{C}_l| \geq 3$ aggregates multiple pairwise inequalities from (65) into a single, stronger cut, reducing redundancy while preserving validity. □

B Decomposition Strategies (DS3, DS4)

B.1 Strategy DS3

The MP is formulated as (68)-(72). Similar to DS_2, an integer optimality cut (69) 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 (70) 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:

$$\max\ Z_{MP_{DS_3}} = -\sum_{j \in \mathcal{N}} f_j z_j - \sum_{(i,j) \in \mathcal{A}} f_{ij} y_{ij} + \theta \tag{68}$$

$$s.t. \quad (7)$$

$$Optimality\ Cuts \quad \forall \gamma \in \Gamma_{DS_3} \tag{69}$$

$$Feasibility\ Cuts \quad \forall \omega \in \Omega_{DS_3} \tag{70}$$

$$z_j \in \{0, 1\} \quad \forall j \in \mathcal{N} \tag{71}$$

$$y_{ij} \in \{0, 1\} \quad \forall (i, j) \in \mathcal{A} \tag{72}$$

The SP for DS_3 is formulated as follows:

$$\max\ Z_{SP_{DS_3}} = \sum_{k\in\mathcal{K}} r^k d^k w^k - \sum_{(i,j)\in\mathcal{A}} \sum_{k\in\mathcal{K}} c_{ij}^k x_{ij}^k \tag{73}$$

$$s.t. \quad (2), (4), (5)$$

$$\sum_{k\in\mathcal{K}} x_{ij}^k \leq u_{ij}\overline{y}_{ij} \quad \forall (i,j)\in\mathcal{A} \tag{74}$$

$$\sum_{\substack{k\in\mathcal{K}:\\ \{O(k)\neq j, D(k)\neq j\}}} \sum_{\substack{i\in\mathcal{N}:\\ (i,j)\in\mathcal{A}}} x_{ij}^k \leq u_j\overline{z}_j \quad \forall j\in\mathcal{N} \tag{75}$$

$$w^k, v_{ij}^k \in \{0,1\},\ x_{ij}^k \in \mathbb{R}_{\geq 0} \quad \forall (i,j)\in\mathcal{A}, k\in\mathcal{K} \tag{76}$$

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

$$\theta \leq \left(Z_{\mathrm{SP}_{DS_3}} - UB_{\mathrm{SP}_{DS_3}}\right) \times \left(\sum_{(i,j)\in\eta_y} y_{ij} - \sum_{(i,j)\notin\eta_y} y_{ij} + \sum_{j\in\eta_z} z_j - \sum_{j\notin\eta_z} z_j - (|\eta_y| + |\eta_z|)\right) + Z_{\mathrm{SP}_{DS_3}} \tag{77}$$

$UB_{\mathrm{SP}_{DS_3}}$ 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.

$$\sum_{(i,j)\in\eta_y} (1 - y_{i,j}) + \sum_{(i,j)\notin\eta_y} y_{i,j} + \sum_{j\in\eta_z} (1 - z_j) + \sum_{j\notin\eta_z} z_j \geq 1 \tag{78}$$

B.2 Strategy DS4

The MP is formulated as (79)-(82). Likewise DS_2 and DS_3 an integer optimality cut (80) 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 (81) 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:

$$\max\ Z_{MP_{DS_4}} = \sum_{k\in\mathcal{K}} r^k d^k w^k - \sum_{(i,j)\in\mathcal{A}} f_{ij} y_{ij} + \theta \tag{79}$$

$$s.t. \quad Optimality\ Cuts \quad \forall\gamma\in\Gamma_{DS_4} \tag{80}$$

$$Feasibility\ Cuts \quad \forall\omega\in\Omega_{DS_4} \tag{81}$$

$$y_{ij}, w^k \in \{0,1\} \quad \forall (i,j)\in\mathcal{A}, k\in\mathcal{K} \tag{82}$$

The SP for DS_4 is formulated as follows:

$$\max\ Z_{SP_{DS_4}} = -\sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}} c_{ij}^k x_{ij}^k - \sum_{j\in\mathcal{N}} f_j z_j \tag{83}$$

$$s.t.\quad (4),(5),(6),(7),(8)$$

$$\sum_{j\in\mathcal{A}_i^+} x_{ij}^k - \sum_{j\in\mathcal{A}_i^-} x_{ji}^k = \begin{cases} d^k\overline{w}^k, & i = O(k) \\ -d^k\overline{w}^k, & i = D(k) \\ 0, & i \neq O(k), i \neq D(k) \end{cases} \qquad \forall k\in\mathcal{K}, i\in\mathcal{N} \tag{84}$$

$$\sum_{k\in\mathcal{K}} x_{ij}^k \leq u_{ij}\overline{y}_{ij} \quad \forall (i,j)\in\mathcal{A} \tag{85}$$

$$v_{ij}^k \in \{0,1\},\ x_{ij}^k \in \mathbb{R}_{\geq 0} \quad \forall (i,j)\in\mathcal{A}, k\in\mathcal{K} \tag{86}$$

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

$$\theta \leq \left(Z_{\mathrm{SP}_{DS_4}} - UB_{\mathrm{SP}_{DS_4}}\right) \times \left(\sum_{(i,j)\in\eta_y} y_{ij} - \sum_{(i,j)\notin\eta_y} y_{ij} + \sum_{k\in\eta_w} w_k - \sum_{k\notin\eta_w} w_k - (|\eta_y| + |\eta_w|)\right) + Z_{\mathrm{SP}_{DS_4}} \tag{87}$$

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 set η_w is 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.

$$\sum_{(i,j)\in\eta_y} (1 - y_{i,j}) + \sum_{(i,j)\notin\eta_y} y_{i,j} + \sum_{k\in\eta_w} (1 - w_k) + \sum_{k\notin\eta_w} w_k \geq 1 \tag{88}$$

C Feasibility Problem

The SPF_{DS_2} is presented as follows:

$$\begin{aligned} \text{Min}\quad Z_{SPF_{DS_2}} = &\sum_{i\in\mathcal{N}}\sum_{\substack{k\in\mathcal{K}\\ i\neq O(k)}} \epsilon_{ik}^1 + \epsilon_{ik}^2 + \sum_{i\in\mathcal{N}}\sum_{\substack{k\in\mathcal{K}\\ i\neq D(k)}} \epsilon_{ik}^3 + \epsilon_{ik}^4 + \sum_{i\in\mathcal{N}}\sum_{\substack{k\in\mathcal{K}\\ i\neq O(k)\\ i\neq D(k)}} \epsilon_{ik}^5 + \epsilon_{ik}^6 \\ &+ \sum_{(i,j)\in\mathcal{A}} \epsilon_{ij}^7 + \sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{A}} \epsilon_{ijk}^8 + \epsilon_{ijk}^9 + \sum_{j\in\mathcal{N}} \epsilon_j^{10} \end{aligned} \tag{89}$$

$$s.t.\ (19)-(21),(24)-(31),(33)-(34)$$

Consider $\hat{\beta}^1_{i,j}$, $\hat{\beta}^2_k$, and $\hat{\beta}^3_j$ to be the dual multipliers of constraints (29)-(31) respectively. We can then generate the feasibility cut as follows:

$$\begin{aligned} 0 \geq & \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq O(k)}} (\hat{\epsilon}^1_{ik} + \hat{\epsilon}^2_{ik}) + \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq D(k)}} (\hat{\epsilon}^3_{ik} + \hat{\epsilon}^4_{ik}) + \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq O(k) \\ i\neq D(k)}} (\hat{\epsilon}^5_{ik} + \hat{\epsilon}^6_{ik}) \\ & + \sum_{(i,j)\in\mathcal{A}} \hat{\epsilon}^7_{ij} + \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}} (\hat{\epsilon}^8_{ijk} + \hat{\epsilon}^9_{ijk}) + \sum_{j\in\mathcal{N}} \hat{\epsilon}^{10}_j \\ & + \sum_{(i,j)\in\mathcal{A}} \hat{\beta}^1_{i,j}(y_{i,j} - \hat{y}_{i,j}) + \sum_{k\in\mathcal{K}} \hat{\beta}^2_k(w_k - \hat{w}_k) + \sum_{j\in\mathcal{N}} \hat{\beta}^3_j(z_j - \hat{z}_j) \end{aligned} \tag{90}$$

After generating feasibility cuts (90), the following feasibility problem ($SPFL_{DS_2}$) can be solved to generate strengthened feasibility cuts (92). Similarly, ϵ denotes a decision variable that captures the amount of deviation from feasibility to be minimized, and $\tilde{\epsilon}$ represents its value.

$$\begin{aligned} \text{Min} \quad Z_{SPFL} = & \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq O(k)}} \epsilon^1_{ik} + \epsilon^2_{ik} + \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq D(k)}} \epsilon^3_{ik} + \epsilon^4_{ik} + \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq O(k) \\ i\neq D(k)}} \epsilon^5_{ik} + \epsilon^6_{ik} \\ & + \sum_{(i,j)\in\mathcal{A}} \epsilon^7_{ij} + \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}} \epsilon^8_{ijk} + \epsilon^9_{ijk} + \sum_{j\in\mathcal{N}} \epsilon^{10}_j \\ & - \sum_{(i,j)\in\mathcal{A}} \hat{\beta}^1_{i,j}(y_{i,j} - \overline{y}_{i,j}) - \sum_{k\in\mathcal{K}} \hat{\beta}^2_k(w_k - \overline{w}_k) - \sum_{j\in\mathcal{N}} \hat{\beta}^3_j(z_j - \overline{z}_j) \end{aligned} \tag{91}$$

$$s.t. (8), (9), (24) - (28), (33) - (34)$$

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:

$$\begin{aligned} 0 \geq & \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq O(k)}} \tilde{\epsilon}^1_{ik} + \tilde{\epsilon}^2_{ik} + \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq D(k)}} \tilde{\epsilon}^3_{ik} + \tilde{\epsilon}^4_{ik} + \sum_{i\in\mathcal{N}} \sum_{\substack{k\in\mathcal{K} \\ i\neq O(k) \\ i\neq D(k)}} \tilde{\epsilon}^5_{ik} + \tilde{\epsilon}^6_{ik} \\ & + \sum_{(i,j)\in\mathcal{A}} \tilde{\epsilon}^7_{ij} + \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}} \tilde{\epsilon}^8_{ijk} + \tilde{\epsilon}^9_{ijk} + \sum_{j\in\mathcal{N}} \tilde{\epsilon}^{10}_j \\ & + \sum_{(i,j)\in\mathcal{A}} \hat{\beta}^1_{i,j}(y_{i,j} - \tilde{y}_{i,j}) + \sum_{k\in\mathcal{K}} \hat{\beta}^2_k(w_k - \tilde{w}_k) + \sum_{j\in\mathcal{N}} \hat{\beta}^3_j(z_j - \tilde{z}_j) \end{aligned} \tag{92}$$

D UBF for DS3 and DS4

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

$$\boldsymbol{DS_3}: \quad \theta \leq \sum_{k \in \mathcal{K}} w^k r^k d^k - \sum_{k \in \mathcal{K}} C^k_{P(O(k),D(k))} d^k w^k \tag{93}$$

$$\boldsymbol{DS_4}: \quad \theta \leq -\sum_{k \in \mathcal{K}} C^k_{P(O(k),D(k))} d^k w^k - \sum_{j \in \mathcal{N}} f_j z_j \tag{94}$$

E Pseudo codes

Algorithm 1 presents the warm start cutting plane Benders decomposition algorithm. Algorithm 2 presents the two-phase greedy heuristic approach. Finally Algorithm 3 describes the branch and Benders cut algorithm. The pseudo codes for Algorithms 1, 2 and 3:

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 ← 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 ← FINDFEASIBLEPATH(k)
        if path exists and profit > 0 then
            Add k to Q
        end
    end
    while Q ≠ ∅ 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
```

F 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 (56), (57)–(60), and (63)). 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 (55), (64), and (66)) are incorporated in this phase as a result of MD. VIs (61), (62), and (67) are added on the fly. As illustrated in Algorithm 3, fractional solutions are only explored for DS_1 in the

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

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 ((61), (62)). 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.

G Complementary results

The results for small, medium and large size instances to evaluate the B&BC compared to Gurobi for $T = 0.5|\mathcal{N}|$ candidate transshipment points are presented in Table 7. The results for comparing the effect of each class of VI for $T = 0.5|\mathcal{N}|$ are presented in Table 8.

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%	-	-	9
	r06	10	60	50	40.2	0.0%	35.2	0.0%	-	-	9
	C33–36	20	230	40	6.5	0.0%	18.1	0.0%	-	-	3
	C41–44	20	300	40	25.3	0.0%	29.7	0.0%	-	-	4
	Avg. Small				15.0	0.0%	20.3	0.0%	0.0%	0.0%	100.0%
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	3
	Avg. Medium				141.0	0.0%	492.8	0.0%	11.1%	8.3%	80.6%
Large	r11	20	120	100	4399.5	0.2%	1102.7	0.0%	1	4	4
	r12	20	120	200	6420.5	0.7%	4931.1	0.0%	4	3	2
	r13	20	220	40	5528.9	0.0%	1272.7	0.0%	-	7	2
	r14	20	220	100	10014.3	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%	-	4	-
	C45–48	20	300	200	18000	0.9%	18000	0.8%	-	3	1
	r16	20	314	40	5689.9	0.1%	633.0	0.0%	-	7	2
	r17	20	318	100	17035.8	1.4%	7745.5	0.2%	-	9	-
	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%	-	4	-
	C53–56	30	520	400	18000	0.9%	18000	1.1%	2	1	1
	C57–60	30	700	100	15989.6	0.1%	15022.5	0.1%	-	4	-
	C61–64	30	700	400	18000	1.0%	18000	0.9%	1	3	-
	Avg. Large				13469.6	1.0%	11114.4	0.58%	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	10030.4	0.3%	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	0.3%	4/9
r12	6420.5	0.7%	4931.1	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%	4/9	5331.7	0.0%	5/9	4999.4	0.0%	5/9	5178.8	0.0%	5/9	4310.9	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	0.5%	5/9	7220.9	0.6%	5/9	7224.0	0.4%	6/9	7127.3	0.5%	5/9	6243.7	0.3%	6/9	6616.9	0.5%	6/9
r15	15434.2	2.2%	14065.2	0.6%	6/9	18000.1	2.2%	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	18003.5	1.6%	18000.1	1.0%	4/4	18000.2	1.2%	4/4	18000.1	0.7%	4/4	18000.2	1.1%	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	0.9%	2/4	18000.2	0.6%	3/4	18000.2	0.7%	4/4	18000.1	0.8%	4/4	18000.2	0.8%	3/4	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%	7/9	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%	7745.5	0.2%	9/9	11762.5	0.6%	6/9	7759.0	0.5%	8/9	9341.1	0.5%	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	8139.9	0.2%	9/9
r18	18004.1	3.5%	16514.1	2.8%	7/9	17724.1	3.7%	4/9	17189.7	2.7%	8/9	16610.6	3.5%	5/9	16848.5	2.9%	7/9	16884.6	3.2%	6/9	16954.1	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	15989.6	0.1%	15022.5	0.1%	4/4	15992.6	0.1%	1/4	14142.0	0.0%	4/4	15330.0	0.1%	4/4	15436.2	0.1%	4/4	15223.5	0.1%	4/4	15302.8	0.1%	4/4	14374.3	0.1%	3/4	14688.2	0.0%	4/4
C61-64	18025.7	1.0%	18000.7	0.9%	3/4	18000.7	1.0%	2/4	18000.8	0.9%	3/4	18001.1	0.9%	3/4	18000.7	0.9%	2/4	18000.7	0.9%	2/4	18000.8	1.3%	1/4	18000.7	0.9%	3/4	18000.7	0.9%	3/4
Average	13469.6	1.0%	11114.4	0.58%	72.9%	13620.9	0.91%	54.2%	12188.3	0.66%	66.7%	12464.3	0.75%	63.5%	12485.3	0.72%	62.5%	12699.6	0.71%	65.6%	12593.4	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 $T = 0.5|\mathcal{N}|$