https://pubsonline.informs.org/journal/trsc

TRANSPORTATION SCIENCE
Vol. 00, No. 0, Xxxxx 0000, pp. 000–000
ISSN 0041-1655, EISSN 1526-5447

Submitted to *Transportation Science*

A Dual-Guided Column-Generation-Based Matheuristic for Integrated Railroad Blocking

Yida Xu[†], Zhengzhong Ricky You[†], Baichuan Mo[*]
Department of Civil Engineering, Tsinghua University, Beijing 100084, China.
[†]These authors contributed equally to this work.
[*]Corresponding author.

Abstract. Integrated railroad blocking is a central planning problem in freight railway operations, which requires coordinated decisions on block design, blocking sequences, and block routing. The interactions among these decisions and the scale of real railway networks make direct mixed-integer optimization computationally challenging. Existing construction and local-search heuristics may overlook promising transportation paths, while expanding the path set increases the size of the resulting optimization problems. To solve this problem, we propose a *dual-guided column-generation-based matheuristic* (DG-CGM) that uses linear programming (LP) dual information to identify promising transportation paths and incorporates them into restricted mixed-integer programs to improve feasible solutions. Starting from an initial plan built by *construction*, DG-CGM alternates *destroy* and *local repair* with *path generation*, which uses dual prices from restricted LPs of a Dantzig–Wolfe reformulation to identify promising transportation paths. When local improvements continue to stagnate, we further apply *joint repair* to combine alternatives across demand groups. To address the coupling between logical and physical path pricing, we introduce a *joint reduced cost* that eliminates path-specific dual variables from pricing, enabling more efficient column identification through exact labeling algorithms. We evaluate the performance of DG-CGM on both generated instances and real-world cases. On 150 generated instances with 50–150 nodes, DG-CGM achieves the optimal cost on 89 of the 91 instances solved to proven optimality by Gurobi, with a geometric mean runtime of only 5.2% of Gurobi's and a mean relative cost reduction of 3.4%. On six full-scale scenarios derived from the 2026 INFORMS Railway Applications Section competition data, each with 47,193 nodes, DG-CGM obtains feasible solutions within two hours that serve 95.9%–99.6% of the requested freight volume, providing practical blocking plans for large-scale railway operations. The code, instances, and computational results are publicly available at https://github.com/Zhengzhong-You/IRBP.

Key words: Integrated railroad blocking; Dantzig–Wolfe decomposition; column generation; matheuristics

1. Introduction

Railroad blocking consolidates compatible railcars into blocks that move together between classification yards. At Union Pacific's Bailey Yard alone, approximately 10,000 railcars are handled daily, with more than 3,000 sorted across 114 classification tracks (Union Pacific 2025, Visit North Platte 2020). Improvements in blocking decisions can yield substantial economic benefits. For example, CSX reported that its Cumberland network optimization generated approximately $15 million in direct savings and reduced railcar handlings by 59,000 (CSX Corporation 2024). At the network level, freight transportation requires three

interdependent decisions on which yard-to-yard blocks to open (*block design*), which sequence of compatible blocks each demand should use (*blocking sequence*), and which physical route each open block should follow (*block routing*). These three decisions define the integrated railroad blocking problem (IRBP) (Ahuja et al. 2007, INFORMS Railway Applications Section 2026). The IRBP seeks to minimize transportation and operating costs while satisfying minimum block loads and shared resource capacities, which couple the decisions across different demands.

Existing studies have developed arc-based mixed-integer programming (MIP) formulations and exact methods for railroad blocking (Newton et al. 1998, Arsalani et al. 2023). However, real-world instances can involve millions of decision variables. For example, the 2026 INFORMS Railway Applications Section (RAS) Problem Solving Competition provides a network with 47,193 physical nodes, 106,570 directed arcs, and more than 2,000 freight demands (INFORMS Railway Applications Section 2026). At this scale, obtaining proven optimal solutions within practical time limits is challenging. Heuristic approaches often restrict candidate blocks or paths to reduce the model size, potentially excluding better solutions (Newton et al. 1998, Fang et al. 2023). Neighborhood search provides greater flexibility by repeatedly optimizing selected subsets of decisions (Ahuja et al. 2007, Yaghini et al. 2021). Nevertheless, its effectiveness depends on the available paths. Even an optimally solved neighborhood may miss improvements if suitable paths have not been generated, while adding too many paths increases the size of the resulting MIPs.

Linear programming (LP) dual prices and reduced costs identify promising decisions from network-wide information (Buson et al. 2014, Hewitt et al. 2013). For the IRBP, such information can be obtained from the LP relaxation of a compact arc-based formulation (ARC). However, this relaxation is weak, which can limit the value of its dual information for guiding integer search. To obtain stronger LP-based guidance, we derive an exact Dantzig–Wolfe (DW) formulation by convexifying the demand and block substructures (Lübbecke and Desrosiers 2005, Lübbecke and Witt 2018). We establish its integer equivalence to ARC and prove that its LP relaxation is at least as strong as that of ARC, with strict improvement possible. As shown in Section 6.2, the mean lower-bound gap decreases from 67.5% for ARC to 3.1% for DW. Although the DW relaxation provides substantially tighter lower bounds, solving it involves challenging pricing subproblems because logical paths and physical block routes are coupled through the DW master constraints. To address this difficulty, we introduce a *joint reduced cost* (JRC) measure that eliminates the dependence on path-specific dual variables in pricing while preserving the exact pricing optimality condition. This allows the corresponding labeling algorithms to identify improving columns more efficiently. In particular, a nonnegative minimum JRC remains a valid stopping criterion for exact pricing, allowing the converged DW relaxation to provide a valid lower bound.

Building on the DW formulation, we develop a *dual-guided column-generation-based matheuristic* (DG-CGM). Its central idea is to construct a feasible blocking plan and progressively improve it by using DW dual information to generate promising transportation paths for restricted MIP optimization. The method

has two main stages. In the first stage, *construction* builds an initial feasible solution by grouping compatible demands around candidate blocks and consolidation hubs. The second stage combines *destroy*, *local repair*, *path generation*, and *joint repair*. Specifically, *destroy* releases related demand assignments, and *local repair* jointly reoptimizes their logical paths, served volumes, and associated physical routes. When this local search stagnates, *path generation* solves a restricted DW LP over the current logical and physical paths and uses its dual prices to identify additional logical paths for repair. If subsequent rounds of local search do not improve the solution, *joint repair* further optimizes selected demand groups jointly over their accumulated paths before the search returns to *destroy* and *local repair*.

Our contributions are threefold.

- We develop the ARC and DW formulations for the IRBP, prove their integer equivalence, and establish the strength of the DW relaxation. We further develop exact JRC-based pricing algorithms that simplify the search for improving columns while preserving pricing optimality. These exact methods also provide benchmarks for evaluating DG-CGM.
- We develop DG-CGM, which constructs an initial feasible blocking plan and iteratively improves it through dual-guided *path generation* and restricted MIP-based neighborhood search. *Path generation* solves a restricted DW LP and returns the logical paths with the lowest JRC as additional candidates for subsequent *local repair* and *joint repair*. This iterative combination of column generation and integer repair allows the search to explore new blocking decisions while keeping the optimization problems manageable.
- We conduct extensive computational experiments to evaluate the effectiveness of DG-CGM on generated instances calibrated to real railway data and full-scale RAS instances. On 150 generated instances, DG-CGM matches 89 of the 91 optima proven by ARC MIP within two hours, while reducing feasible costs by 3.4% on average and up to 40.4%. On larger instances, its mean gap to the DW lower bound after 300 seconds is at most 1.6%, compared with 24.2%–78.8% for ARC MIP after two hours. We further conduct ablation studies to assess the contributions of the individual search components and validate their design choices. On six full-scale RAS configurations, DG-CGM obtains feasible solutions within two hours that serve 95.9%–99.6% of the requested freight volume, demonstrating its ability to construct high-service blocking plans on large railway networks.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 defines the IRBP and presents ARC. Section 4 develops the DW formulation and pricing algorithms. Section 5 presents DG-CGM, followed by computational results in Section 6. Section 7 concludes the paper.

2. Related Work

We review railroad blocking and integrated service network design in Section 2.1, DW decomposition and column generation in Section 2.2, and heuristic and matheuristic methods in Section 2.3.

2.1. Railroad Blocking and Integrated Service Network Design

Early railroad blocking studies focused on coordinating railcar classification and block formation across railway networks. For example, Bodin et al. (1980) modeled network-wide classification strategies as a nonlinear mixed-integer multicommodity flow problem. Extending this perspective to broader railway operations, Crainic et al. (1984) jointly considered freight routing, train services, and yard classification. Subsequently, Keaton (1989) incorporated train frequencies, car routing, and train-size and yard-capacity constraints, using Lagrangian relaxation and heuristics to construct feasible operating plans.

Later studies incorporated more detailed blocking constraints and developed corresponding solution methods. Newton et al. (1998) considered classification limits, shipment priorities, and mileage, handling, and delay costs, developing a column-generation-based branch-and-bound method. Building on this network design perspective, Barnhart et al. (2000) distinguished limits on the number of blocks formed at a yard from limits on the freight volume handled there, using Lagrangian relaxation.

Subsequent research extended railroad blocking to integrated service network design, incorporating additional train operations and service requirements. Zhu et al. (2014) jointly considered train scheduling, blocking, train makeup, and time-dependent shipment routing. More closely related to our setting, Arsalani et al. (2023) integrated train formation and shipment routing under link, classification, and path-length restrictions. More recently, Xin et al. (2025) considered heterogeneous demands under mixed blocking policies, combining time-constrained express shipments with an intree policy for ordinary freight.

These studies address different combinations of blocking, routing, scheduling, and operational constraints. Building on this literature, the IRBP considered here jointly optimizes block design, logical demand paths, and physical block routes in a static setting. It allows continuous partial service while requiring each served demand to follow a single logical path. Minimum and maximum block loads, shared physical link capacities, yard handling limits, and classification track capacities further couple these decisions.

2.2. DW Decomposition and Column Generation

Arc-based formulations represent block or service decisions and shipment movements through variables associated with individual network arcs. Flow conservation, activation constraints, and shared capacities link these decisions. Such formulations provide a direct representation of the network, as illustrated by the integrated models of Yaghini et al. (2021) and Arsalani et al. (2023). Their size grows with the network and the number of commodities, while the quality of the LP bound depends on how the interactions among the discrete decisions are represented.

DW decomposition replaces selected substructures with columns representing complete feasible patterns, coordinated through a master problem (Dantzig and Wolfe 1960, Lübbecke and Desrosiers 2005). When the columns represent integer-feasible subproblem solutions, the reformulation convexifies those substructures and can strengthen the original LP relaxation. The improvement depends on the chosen decomposition, not

simply on replacing arc variables with path variables. Lübbecke and Witt (2018) characterize this dependence for stable set and related problems, showing that different decompositions can yield substantially different relaxation strengths. For railroad blocking, a path representation also allows routing feasibility to be handled within subproblems while the master coordinates shared resources.

However, the resulting master problem may contain too many columns to enumerate explicitly. To address this difficulty, column generation maintains a restricted master problem (RMP) containing only a subset of columns. At each iteration, the RMP is solved to obtain dual prices, which guide the pricing subproblems in identifying improving columns. These columns are then added to the RMP, and the process repeats until pricing confirms that no improving columns remain (Lübbecke and Desrosiers 2005). In railroad blocking, Newton et al. (1998) used shortest-path pricing to generate shipment paths within a branch-and-bound framework, while Voll and Clausen (2013) developed a branch-and-price method for a European variant of the problem.

Computational performance depends heavily on pricing efficiency. In our DW formulation, logical and physical paths are coupled through master constraints involving path selection and volume variables, which complicates the design of conventional pricing algorithms based on reduced costs. In contrast, the proposed JRC measure accounts for this coupling while eliminating the dependence on path-specific dual variables, enabling more efficient pricing.

2.3. Heuristics and Matheuristics

Problem-specific heuristics have been central to solving large railroad blocking and service network design problems. Keaton (1989) used Lagrangian relaxation together with heuristic adjustments to obtain operating plans satisfying train capacities. Barnhart et al. (2000) similarly developed a Lagrangian approach that separates the blocking problem into simpler subproblems and constructs feasible plans. Neighborhood search provides another way to improve a plan without optimizing all decisions simultaneously. Ahuja et al. (2007) developed a very large-scale neighborhood search for railroad blocking and applied it to operating plans for major US railroads.

Metaheuristics broaden the exploration of blocking plans through population search or mechanisms for moving beyond local improvements. Yaghini et al. (2012) developed a genetic algorithm for selecting blocks and assigning commodities, and Yaghini et al. (2013) proposed a tabu search algorithm for the railroad blocking problem. Both studies evaluated solution quality and computation time against a MIP solver. Such methods make large problems accessible to heuristic search, but the moves and candidate decisions available within each neighborhood still determine which coordinated changes can be explored.

Matheuristics use mathematical optimization to guide this search or optimize selected parts of a solution. For scheduled rail service network design, Zhu et al. (2014) combined slope scaling, dynamic block generation, memory-based perturbation, and optimization over large neighborhoods. Particularly relevant to our

setting, Yaghini et al. (2021) developed an LP-based neighborhood search for integrated train blocking and shipment routing. Their method repeatedly solves LP relaxations, modifies objective coefficients through a tabu-inspired penalization scheme, and constructs a restricted MIP from the resulting solutions. They report high-quality solutions within reasonable computation times. These approaches illustrate how network-wide optimization information can support the selection and joint adjustment of blocking decisions.

Candidate generation provides a complementary way to control the size of the optimization problems. Fang et al. (2023) generated promising candidate blocks using shortest-path information and optimized blocking and train design through a two-stage approach. In the related multicommodity fixed-charge network flow problem, Hewitt et al. (2013) used column generation both to construct problem restrictions and to obtain bounds, together with local search around the incumbent. Although their algorithm is exact, its use of generated restrictions to obtain feasible solutions is closely related to the design of matheuristics.

DG-CGM follows this optimization-guided search approach by combining DW-based *path generation* with restricted MIP repair. Generated logical paths expand the choices available to selected demand groups, while repair jointly adjusts their service volumes and associated block paths. The connection between *path generation* and neighborhood optimization is central to the method, since improving the available paths and coordinating their use are complementary parts of the integer search.

3. Problem Definition and Compact Formulation

We first define the IRBP in Section 3.1, describing its physical and logical networks, operating decisions, feasibility constraints, and objective function. Section 3.2 then presents the compact arc-based mixed-integer linear formulation and explains how its constraints represent the problem requirements.

3.1. Problem Definition

In the IRBP, freight demands are assigned to sequences of blocks in what we call the logical network, while each open block follows a route through the physical railway network. The logical network $G^{\mathrm{Y}} = (\mathcal{Y}, \mathcal{B})$ is a directed multigraph, where $\mathcal{Y}$ is the set of classification yards and $\mathcal{B}$ is the set of candidate blocks. Each block $b \in \mathcal{B}$ carries railcars from yard i_b to a distinct yard j_b without intermediate reclassification. Multiple blocks may connect the same pair of yards. Let $\mathcal{K}$ denote the set of freight demands, where each demand $k \in \mathcal{K}$ has origin o_k, destination $d_k \neq o_k$, requested volume $q_k > 0$, and a set $\mathcal{B}_k \subseteq \mathcal{B}$ of compatible blocks. Some demands prohibit intermediate reclassification and can therefore use only direct blocks. A blocking plan selects a set $\mathcal{B}^{\mathrm{O}} \subseteq \mathcal{B}$ of open blocks (*block design*) and assigns each demand a served volume $v_k \in [0, q_k]$ and, if $v_k > 0$, a single logical path p_k from o_k to d_k using blocks in $\mathcal{B}_k \cap \mathcal{B}^{\mathrm{O}}$ (*blocking sequence*). The resulting load of block b is $f_b = \sum_{k \in \mathcal{K}:\, b \in p_k} v_k$, where unserved demands have empty paths. Every open block must satisfy its minimum load requirement $f_b \geq \underline{f}_b > 0$. Each yard $i \in \mathcal{Y}$ has T_i classification track slots and handling capacity H_i. An open block b occupies $\kappa_b \in \{0, 1\}$ track slots at its origin yard, depending on its commodity type, while the total reclassified volume v_i^{H}, comprising freight transferred

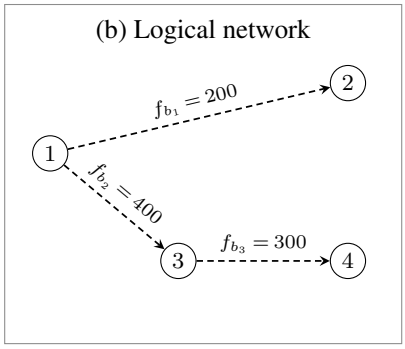
between consecutive blocks at yard i, cannot exceed H_i. The total track slots occupied by open blocks originating at i cannot exceed T_i.

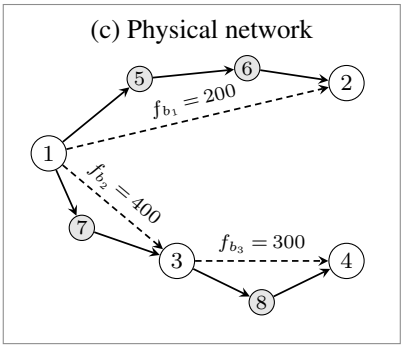
The physical network $G = (\mathcal{N}, \mathcal{E})$ consists of physical nodes $\mathcal{N} \supseteq \mathcal{Y}$ and links $\mathcal{E}$, where the nodes include classification yards and transit points used only for routing. Each link $e \in \mathcal{E}$ permits travel in one or both directions and has capacity C_e, shared across both directions when applicable. Let $\mathcal{A}_e$ denote its permitted directed traversal arcs, and define the directed physical graph $G^{\mathrm{P}} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{A} = \bigcup_{e \in \mathcal{E}} \mathcal{A}_e$. Each arc $a \in \mathcal{A}$ has tail i_a, head j_a, and length $D_a > 0$. For each open block b, *block routing* selects a physical path r_b from i_b to j_b that carries its entire load f_b. The path must satisfy the length limit $\sum_{a \in r_b} D_a \le \overline{D}_b$, and the total loads of blocks traversing each physical link e cannot exceed C_e. Unless stated otherwise, all logical and physical paths are elementary, i.e., each node is visited at most once along a path. These physical capacity constraints couple the routing decisions with the block loads determined in the logical network.

Figure 1 Demand assignment, logical network, and physical network.

(a) Demand assignment

Demand $o_k \to d_k$	(q_k, v_k)	Blocks
$k_1: 1 \to 2$	$(200, 200)$	b_1
$k_2: 2 \to 4$	$(100, 0)$	unserved
$k_3: 1 \to 3$	$(300, 300)$	b_2
$k_4: 3 \to 4$	$(200, 200)$	b_3
$k_5: 1 \to 4$	$(100, 100)$	$b_2 \to b_3$





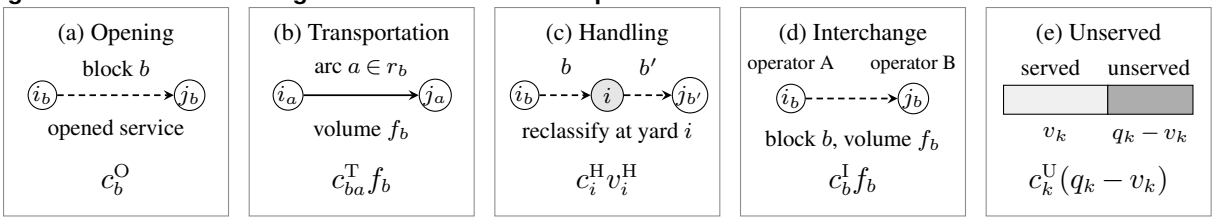
Notes. Dashed arrows represent open blocks, and solid arrows represent physical arcs. Classification track use is limited by the capacity at each yard (assume $T_1 = 3$ and $T_3 = 2$). Yard 3 can reclassify at most 100 units of freight (assume $H_3 = 100$), and each shown arc represents a separate one-way link with capacity 500 (assume $C_e = 500$). Assume unit arc lengths $D_a = 1$, minimum block loads $\underline{f}_b = 100$, and path length budgets $\overline{D}_b = 3$.

Figure 1 illustrates a feasible blocking plan with three open blocks. Demands k_1, k_3, k_4, and k_5 are fully served, while demand k_2 remains unserved. Demand k_5 follows the blocking sequence $b_2 \to b_3$, giving block loads $f_{b_2} = v_{k_3} + v_{k_5} = 400$ and $f_{b_3} = v_{k_4} + v_{k_5} = 300$. All open blocks satisfy the minimum load requirement $f_b \ge \underline{f}_b = 100$ and the physical path length limit $\sum_{a \in r_b} D_a \le \overline{D}_b = 3$. Since blocks b_1 and b_2 originate at yard 1 and block b_3 at yard 3, they occupy two classification track slots at yard 1 and one at yard 3, within the respective capacities $T_1 = 3$ and $T_3 = 2$. Only demand k_5 requires intermediate reclassification, giving $v_3^{\mathrm{H}} = v_{k_5} = 100 \le H_3$, with no reclassification at the other yards. Finally, the load on every physical link is at most 400 units, below its capacity $C_e = 500$.

The objective minimizes the total cost of the blocking plan, comprising the five components illustrated in Figure 2. Opening block b incurs a *fixed opening cost* c_b^{O}, as shown in panel (a). The *transportation cost* $c_{ba}^{\mathrm{T}} \ge 0$ applies per unit of block load carried over arc $a \in r_b$ in panel (b), while the *handling cost* c_i^{H} applies

per unit of freight reclassified at intermediate yard i in panel (c). An *interchange* occurs when freight is transferred between railroad operators, such as from Union Pacific to BNSF Railway, and incurs a per-unit cost c_b^{I} for block b, as shown in panel (d). Finally, the *unserved demand cost* c_k^{U} applies to each of the $q_k - v_k$ units of demand k left unserved, as shown in panel (e).

Figure 2 Decisions that generate the five cost components.



3.2. Arc-Based Formulation

We formulate the IRBP as an arc-based MIP, referred to as ARC, that jointly determines block design, blocking sequences, and block routing. Superscripts D and B distinguish variables associated with demand blocking sequences and physical block routes, respectively. In the logical network, for each demand k and yard $i \in \mathcal{Y}$, let $\mathcal{B}_k^+(i)$ and $\mathcal{B}_k^-(i)$ denote the sets of compatible blocks leaving and entering i. For block design, binary variable y_b indicates whether block b is opened. For each demand k, binary variables x_k^{D} and x_{kb}^{D} indicate whether the demand is assigned a logical path and whether that path uses block $b \in \mathcal{B}_k$, respectively, while continuous variable g_{kb} represents the volume of demand k transported on that block. The served volumes v_k, block loads f_b, and reclassified volumes v_i^{H} defined in Section 3.1 are also continuous variables. The set $\mathcal{K}_b = \{k \in \mathcal{K} : b \in \mathcal{B}_k\}$ contains the demands compatible with block b, and $\overline{f}_b = \sum_{k \in \mathcal{K}_b} q_k$ provides an upper bound on its load. Auxiliary continuous variables ℓ_{ki}^{D} impose an ordering on the yards visited by each demand to exclude cycles in logical paths.

In the physical network, for each node $n \in \mathcal{N}$, let $\mathcal{A}^+(n)$ and $\mathcal{A}^-(n)$ denote the sets of arcs leaving and entering n. For each block b, binary variable x_{ba}^{B} indicates whether its physical path uses arc $a \in \mathcal{A}$, while continuous variable w_{ba} represents the volume of block b transported on that arc. Auxiliary continuous variables ℓ_{bn}^{B} impose an ordering on the nodes visited by each block to exclude cycles. The indicator $\mathbf{1}\{\cdot\}$ equals one when its condition holds and zero otherwise.

The ARC formulation is given in Eq. (1). Objective (1a) minimizes the total cost of block opening, transportation, handling, interchange, and unserved demand. For blocking sequences, Eqs. (1b)–(1c) enforce flow conservation for the selected logical path and its associated freight volume, while Eqs. (1d)–(1e) prevent branching and eliminate cycles in the selected logical paths. Eqs. (1f) and (1h) connect logical path selections x_{kb}^{D} with freight volumes g_{kb} and block openings y_b, while Eq. (1g) imposes lower and upper bounds on served volumes v_k. For block routing, Eqs. (1i)–(1j) and (1m) ensure that each open block follows exactly one elementary physical path from i_b to j_b, subject to the length budget in Eq. (1k). Eq. (1l) aggregates the freight volumes assigned by compatible demands, while Eqs. (1n)–(1o) ensure that all freight

assigned to each block follows its selected physical path. Eq. (1p) gives the lower and upper bounds on the freight volume of each block, while Eq. (1q) calculates the volume reclassified when freight transfers between blocks at intermediate yards. Eqs. (1r), (1s), and (1t) limit classification track usage, aggregate physical link volumes, and yard handling volumes, respectively. Finally, Eqs. (1u)–(1v) specify the variable domains. We denote the optimal objective values of ARC and its LP relaxation by z^*_{arc} and $z^{\text{lp}}_{\text{arc}}$, respectively.

$$\min \sum_{b\in\mathcal{B}} c^{\text{O}}_b y_b + \sum_{b\in\mathcal{B}}\sum_{a\in\mathcal{A}} c^{\text{T}}_{ba} w_{ba} + \sum_{i\in\mathcal{Y}} c^{\text{H}}_i v^{\text{H}}_i + \sum_{b\in\mathcal{B}} c^{\text{I}}_b f_b + \sum_{k\in\mathcal{K}} c^{\text{U}}_k (q_k - v_k) \tag{1a}$$

s.t.

$$\sum_{b\in\mathcal{B}^+_k(i)} x^{\text{D}}_{kb} - \sum_{b\in\mathcal{B}^-_k(i)} x^{\text{D}}_{kb} = x^{\text{D}}_k\big(\mathbf{1}\{i=o_k\} - \mathbf{1}\{i=d_k\}\big), \quad \forall k\in\mathcal{K},\ i\in\mathcal{Y}, \tag{1b}$$

$$\sum_{b\in\mathcal{B}^+_k(i)} g_{kb} - \sum_{b\in\mathcal{B}^-_k(i)} g_{kb} = v_k\big(\mathbf{1}\{i=o_k\} - \mathbf{1}\{i=d_k\}\big), \quad \forall k\in\mathcal{K},\ i\in\mathcal{Y}, \tag{1c}$$

$$\sum_{b\in\mathcal{B}^+_k(i)} x^{\text{D}}_{kb} \le x^{\text{D}}_k, \qquad \sum_{b\in\mathcal{B}^-_k(i)} x^{\text{D}}_{kb} \le x^{\text{D}}_k, \quad \forall k\in\mathcal{K},\ i\in\mathcal{Y}, \tag{1d}$$

$$\ell^{\text{D}}_{kj_b} \ge \ell^{\text{D}}_{ki_b} + 1 - |\mathcal{Y}|(1 - x^{\text{D}}_{kb}), \quad \forall k\in\mathcal{K},\ b\in\mathcal{B}_k, \tag{1e}$$

$$0 \le g_{kb} \le q_k x^{\text{D}}_{kb}, \quad \forall k\in\mathcal{K},\ b\in\mathcal{B}_k, \tag{1f}$$

$$0 \le v_k \le q_k x^{\text{D}}_k, \quad \forall k\in\mathcal{K}, \tag{1g}$$

$$x^{\text{D}}_{kb} \le y_b, \quad \forall k\in\mathcal{K},\ b\in\mathcal{B}_k. \tag{1h}$$

Assignment

$$\sum_{a\in\mathcal{A}^+(n)} x^{\text{B}}_{ba} - \sum_{a\in\mathcal{A}^-(n)} x^{\text{B}}_{ba} = y_b\big(\mathbf{1}\{n=i_b\} - \mathbf{1}\{n=j_b\}\big), \quad \forall b\in\mathcal{B},\ n\in\mathcal{N}, \tag{1i}$$

$$\sum_{a\in\mathcal{A}^+(n)} x^{\text{B}}_{ba} \le y_b, \qquad \sum_{a\in\mathcal{A}^-(n)} x^{\text{B}}_{ba} \le y_b, \quad \forall b\in\mathcal{B},\ n\in\mathcal{N}, \tag{1j}$$

$$\sum_{a\in\mathcal{A}} D_a x^{\text{B}}_{ba} \le \overline{D}_b y_b, \quad \forall b\in\mathcal{B}, \tag{1k}$$

$$f_b = \sum_{k\in\mathcal{K}_b} g_{kb}, \quad \forall b\in\mathcal{B}, \tag{1l}$$

$$\ell^{\text{B}}_{bj_a} \ge \ell^{\text{B}}_{bi_a} + 1 - |\mathcal{N}|(1 - x^{\text{B}}_{ba}), \quad \forall b\in\mathcal{B},\ a\in\mathcal{A}, \tag{1m}$$

$$0 \le w_{ba} \le \overline{f}_b x^{\text{B}}_{ba}, \qquad w_{ba} \le f_b, \quad \forall b\in\mathcal{B},\ a\in\mathcal{A}, \tag{1n}$$

$$w_{ba} \ge f_b - \overline{f}_b(y_b - x^{\text{B}}_{ba}), \quad \forall b\in\mathcal{B},\ a\in\mathcal{A}, \tag{1o}$$

$$\underline{f}_b y_b \le f_b \le \overline{f}_b y_b, \quad \forall b\in\mathcal{B}. \tag{1p}$$

Paths

$$v^{\text{H}}_i = \sum_{k\in\mathcal{K}} \left(\sum_{b\in\mathcal{B}^+_k(i)} g_{kb} - \mathbf{1}\{i=o_k\} v_k \right), \quad \forall i\in\mathcal{Y}, \tag{1q}$$

$$\sum_{b\in\mathcal{B}:i_b=i} \kappa_b y_b \le T_i, \quad \forall i\in\mathcal{Y}, \tag{1r}$$

$$\sum_{b\in\mathcal{B}}\sum_{a\in\mathcal{A}_e} w_{ba} \le C_e, \quad \forall e\in\mathcal{E}, \tag{1s}$$

$$v^{\text{H}}_i \le H_i, \quad \forall i\in\mathcal{Y}. \tag{1t}$$

Capacity

$$0 \le \ell^{\text{D}}_{ki} \le |\mathcal{Y}| - 1, \qquad 0 \le \ell^{\text{B}}_{bn} \le |\mathcal{N}| - 1, \tag{1u}$$

$$x^{\text{D}}_{kb},\ x^{\text{D}}_k,\ y_b,\ x^{\text{B}}_{ba} \in \{0,1\}. \tag{1v}$$

Domains

4. Dantzig–Wolfe Formulation and Column Generation

The weak LP relaxation of ARC, as reflected in the large lower-bound gaps reported in Section 6.2, limits its usefulness for evaluating heuristic solutions and may provide less informative dual prices for guiding integer search. To address this issue, we further develop a DW formulation in Section 4.1. Its LP relaxation is solved to optimality through column generation using the logical and physical path pricing algorithms developed in Section 4.2.

4.1. Dantzig–Wolfe Formulation

The DW formulation represents demand blocking sequences and block routing decisions using complete logical and physical paths, respectively. For each demand $k \in \mathcal{K}$, let $\mathcal{P}_k$ denote the set of all elementary logical paths from o_k to d_k using compatible blocks in $\mathcal{B}_k$. Similarly, for each block $b \in \mathcal{B}$, let $\mathcal{R}_b$ contain all elementary physical paths from i_b to j_b satisfying the length budget $\overline{D}_b$, with $\mathcal{R}_b = \emptyset$ if $\overline{f}_b < \underline{f}_b$.

To represent demand assignments and block routing using complete paths, we introduce binary variables x^{D}_{kp} and x^{B}_{br} to indicate whether demand k selects logical path p and block b follows physical path r, respectively, while continuous variables v^{D}_{kp} and v^{B}_{br} denote the corresponding freight volumes. Let ι_{bp} equal one if logical path p uses block b, and δ_{ar} equal one if physical path r uses arc a, and zero otherwise. The coefficient $\eta_{ip} = \mathbf{1}\{i \neq o_k\} \sum_{b \in \mathcal{B}^+_k(i)} \iota_{bp}$ indicates whether freight is reclassified at yard i along logical path p. We preserve the cost structure of ARC by expressing its volume-dependent terms at the path level. Specifically, the logical path unit cost $c^{\mathrm{D}}_{kp} = \sum_{i \in \mathcal{Y}} c^{\mathrm{H}}_i \eta_{ip} - c^{\mathrm{U}}_k$ accounts for handling costs at intermediate yards and avoided unserved demand costs, while the physical path unit cost $c^{\mathrm{B}}_{br} = c^{\mathrm{I}}_b + \sum_{a \in \mathcal{A}} c^{\mathrm{T}}_{ba} \delta_{ar}$ includes interchange and transportation costs.

The DW formulation is given in Eq. (2). Objective (2a) minimizes the same total cost of block opening, transportation, handling, interchange, and unserved demand in terms of path variables. Since logical and physical paths are predefined in $\mathcal{P}_k$ and $\mathcal{R}_b$, path connectivity, elementarity, and physical path length limits are incorporated directly into these path sets. Eq. (2b) ensures that the freight volume assigned to each block matches the volume transported along its physical route. For resource capacities, Eqs. (2c)–(2e) limit aggregate physical link volumes, yard handling volumes, and classification track usage, respectively. Eqs. (2f)–(2g) connect path selections with their corresponding freight volumes, imposing demand quantity bounds on logical paths and minimum and maximum load bounds on physical paths. Eqs. (2h)–(2i) restrict each demand to at most one logical path and each block to at most one physical path, while Eq. (2j) ensures that selected logical paths use only open blocks. Finally, Eqs. (2k)–(2l) specify the variable domains. We obtain the LP relaxation by replacing the binary domains with $x^{\mathrm{D}}_{kp}, x^{\mathrm{B}}_{br} \geq 0$. We denote the optimal objective values of DW and its LP relaxation by z^*_{dw} and $z^{\mathrm{lp}}_{\mathrm{dw}}$, respectively.

THEOREM 1. *The* DW *formulation* (2) *is an exact mixed-integer Dantzig–Wolfe reformulation of the* ARC *formulation* (1)*, so* $z^*_{\mathrm{arc}} = z^*_{\mathrm{dw}}$*. Moreover, every feasible solution of the DW LP relaxation can be*

mapped to a feasible solution of the ARC LP relaxation with the same objective value, implying $z^{\mathrm{lp}}_{\mathrm{arc}} \le z^{\mathrm{lp}}_{\mathrm{dw}}$. *There exist instances for which this inequality is strict.*

Proof. See Section EC.1.1 of the electronic companion. □

$$\min \quad \sum_{k\in\mathcal{K}} q_k c^{\mathrm{U}}_k + \sum_{k\in\mathcal{K}}\sum_{p} c^{\mathrm{D}}_{kp} v^{\mathrm{D}}_{kp} + \sum_{b\in\mathcal{B}}\sum_{r} (c^{\mathrm{O}}_b x^{\mathrm{B}}_{br} + c^{\mathrm{B}}_{br} v^{\mathrm{B}}_{br}) \tag{2a}$$

s.t.

$$\sum_{r} v^{\mathrm{B}}_{br} - \sum_{k\in\mathcal{K}_b}\sum_{p} \iota_{bp} v^{\mathrm{D}}_{kp} = 0, \qquad \forall b\in\mathcal{B}, \tag{2b}$$

Balance

$$\sum_{b\in\mathcal{B}}\sum_{r}\left(\sum_{a\in\mathcal{A}_e}\delta_{ar}\right) v^{\mathrm{B}}_{br} \le C_e, \qquad \forall e\in\mathcal{E}, \tag{2c}$$

$$\sum_{k\in\mathcal{K}}\sum_{p} \eta_{ip} v^{\mathrm{D}}_{kp} \le H_i, \qquad \forall i\in\mathcal{Y}, \tag{2d}$$

$$\sum_{b\in\mathcal{B}:i_b=i} \kappa_b \sum_{r} x^{\mathrm{B}}_{br} \le T_i, \qquad \forall i\in\mathcal{Y}, \tag{2e}$$

Capacity

$$0 \le v^{\mathrm{D}}_{kp} \le q_k x^{\mathrm{D}}_{kp}, \qquad \forall k\in\mathcal{K},\ p\in\mathcal{P}_k, \tag{2f}$$

$$\underline{f}_b x^{\mathrm{B}}_{br} \le v^{\mathrm{B}}_{br} \le \overline{f}_b x^{\mathrm{B}}_{br}, \qquad \forall b\in\mathcal{B},\ r\in\mathcal{R}_b, \tag{2g}$$

Volume

$$\sum_{p} x^{\mathrm{D}}_{kp} \le 1, \qquad \forall k\in\mathcal{K}, \tag{2h}$$

$$\sum_{r} x^{\mathrm{B}}_{br} \le 1, \qquad \forall b\in\mathcal{B}, \tag{2i}$$

Convexity

$$\sum_{p} \iota_{bp} x^{\mathrm{D}}_{kp} \le \sum_{r} x^{\mathrm{B}}_{br}, \qquad \forall k\in\mathcal{K},\ b\in\mathcal{B}_k, \tag{2j}$$

Activation

$$x^{\mathrm{D}}_{kp},\ x^{\mathrm{B}}_{br} \in \{0,1\}, \tag{2k}$$

$$v^{\mathrm{D}}_{kp},\ v^{\mathrm{B}}_{br} \ge 0. \tag{2l}$$

Domains

4.2. Column Generation

The DW formulation contains a large number of logical and physical path variables, making direct solution of its LP relaxation impractical. We therefore use column generation, which maintains an RMP and iteratively adds improving columns identified by pricing subproblems. Specifically, Section 4.2.1 develops logical pricing to generate blocking sequences for each demand, while Section 4.2.2 presents physical pricing to identify feasible routes for each block. Both procedures use the dual prices of the RMP to identify improving columns, which are then added for reoptimization.

When both pricing procedures are solved exactly and no improving columns remain, the resulting DW LP optimum provides a valid lower bound for the IRBP. When heuristic pricing is used, the generated columns can still provide promising paths for improving integer solutions, although the resulting LP solution does not certify a valid lower bound. This distinction allows column generation to support both exact bound computation and the heuristic search developed in Section 5.

Let σ_b, μ_e, μ_i^{H}, μ_i^{T}, π_k, π_b, and γ_{kb} denote the dual variables associated with Eqs. (2b), (2c), (2d), (2e), (2h), (2i), and (2j), respectively. The block-balance duals σ_b are unrestricted in sign, while $\mu_e, \mu_i^{\mathrm{H}}, \mu_i^{\mathrm{T}}, \pi_k, \pi_b, \gamma_{kb} \leq 0$.

4.2.1. Logical Path Generation. For each demand $k \in \mathcal{K}$ and logical path $p \in \mathcal{P}_k$, define

$$\bar{c}_{kp}^{\mathrm{D},x} = -\pi_k - \sum_{b \in \mathcal{B}_k} \gamma_{kb} \iota_{bp}, \qquad \bar{c}_{kp}^{\mathrm{D},v} = c_{kp}^{\mathrm{D}} + \sum_{b \in \mathcal{B}_k} \sigma_b \iota_{bp} - \sum_{i \in \mathcal{Y}} \mu_i^{\mathrm{H}} \eta_{ip}. \tag{3}$$

For a path included in the RMP, let $\lambda_{kp} \leq 0$ denote the dual variable associated with $v_{kp}^{\mathrm{D}} - q_k x_{kp}^{\mathrm{D}} \leq 0$ in Eq. (2f). Using the coefficients in Eq. (3), the reduced costs of the corresponding selection and volume variables are

$$\begin{aligned} \mathrm{rc}(x_{kp}^{\mathrm{D}}) &= \bar{c}_{kp}^{\mathrm{D},x} + q_k \lambda_{kp}, \\ \mathrm{rc}(v_{kp}^{\mathrm{D}}) &= \bar{c}_{kp}^{\mathrm{D},v} - \lambda_{kp}. \end{aligned} \tag{4}$$

Conventional pricing uses the reduced costs in Eq. (4) to identify improving logical paths. Labeling algorithms typically extend partial paths while accumulating reduced costs, allowing unpromising labels to be discarded before reaching the destination. However, applying this approach directly to our formulation is challenging because each logical path introduces both a selection variable and a volume variable, linked by $v_{kp}^{\mathrm{D}} \leq q_k x_{kp}^{\mathrm{D}}$, whose reduced costs depend on the path-specific dual variable λ_{kp}. Since this constraint has a zero right-hand side and involves no existing RMP variables, $\lambda_{kp} \leq 0$ can be chosen without affecting the current RMP dual objective or existing reduced costs. Thus, conventional pricing would require finding a suitable λ_{kp} for each candidate path such that the reduced costs of both variables are nonnegative whenever possible. This path-dependent choice complicates the incremental evaluation of reduced costs and the design of dominance rules in labeling algorithms.

To address this difficulty, we define the JRC to combine the reduced costs of the selection and volume variables without depending on λ_{kp}. Specifically, the JRC of logical path p is defined as

$$\tilde{c}_{kp}^{\mathrm{D}} = \bar{c}_{kp}^{\mathrm{D},x} + q_k \bar{c}_{kp}^{\mathrm{D},v}. \tag{5}$$

By Proposition 4, a nonnegative minimum JRC in Eq. (5) over all feasible logical paths for every demand certifies that no improving logical columns remain. Moreover, the JRC can be decomposed into an initial cost $\tilde{c}_k^{\mathrm{D,init}} = -q_k c_k^{\mathrm{U}} - \pi_k$ and a block extension cost

$$\tilde{c}_{kb}^{\mathrm{D}} = q_k \sigma_b - \gamma_{kb} + q_k \mathbf{1}\{i_b \neq o_k\}(c_{i_b}^{\mathrm{H}} - \mu_{i_b}^{\mathrm{H}}). \tag{6}$$

Using Eq. (6), the JRC of logical path p is $\tilde{c}_{kp}^{\mathrm{D}} = \tilde{c}_k^{\mathrm{D,init}} + \sum_{b \in \mathcal{B}_k} \iota_{bp} \tilde{c}_{kb}^{\mathrm{D}}$.

We use a labeling algorithm to find minimum-JRC elementary logical paths. For each demand k, a label $L = (i, S, c)$ records the current yard i, visited yards S, and accumulated JRC c. We initialize $L_0 = (o_k, \{o_k\}, \tilde{c}_k^{\mathrm{D,init}})$. For each compatible block $b \in \mathcal{B}_k^+(i)$ with $j_b \notin S$, the label is extended to $L' = (j_b, S \cup \{j_b\}, c + \tilde{c}_{kb}^{\mathrm{D}})$. A label reaching d_k represents a complete logical path with JRC c. The dominance rule used to reduce the labeling search space is given in Proposition 1.

PROPOSITION 1. *Fix demand k. For two feasible labels $L_1 = (i, S_1, c_1)$ and $L_2 = (i, S_2, c_2)$ at the same yard, write $L_1 \preceq L_2$ if $S_1 \subseteq S_2$ and $c_1 \le c_2$. Then L_1 dominates L_2, and discarding L_2 while retaining L_1 preserves the minimum JRC over all feasible logical paths for demand k.*

Proof. See Section EC.1.2 of the electronic companion. □

4.2.2. Physical Path Generation.

For each block $b \in \mathcal{B}$ and physical path $r \in \mathcal{R}_b$, define

$$\begin{aligned} \bar{c}^{\mathrm{B},x}_{br} &= c^{\mathrm{O}}_b - \pi_b - \kappa_b \mu^{\mathrm{T}}_{i_b} + \sum_{k \in \mathcal{K}_b} \gamma_{kb}, \\ \bar{c}^{\mathrm{B},v}_{br} &= c^{\mathrm{B}}_{br} - \sigma_b - \sum_{a \in \mathcal{A}} \mu_{e_a} \delta_{ar}, \end{aligned} \tag{7}$$

where e_a denotes the unique physical link containing arc a. For each block b and physical path $r \in \mathcal{R}_b$, we define its JRC at load t using the coefficients in Eq. (7) as

$$\tilde{c}^{\mathrm{B}}_{br}(t) = \bar{c}^{\mathrm{B},x}_{br} + t \bar{c}^{\mathrm{B},v}_{br}. \tag{8}$$

A selected physical path carries a load $t \in [\underline{f}_b, \overline{f}_b]$. Since the JRC in Eq. (8) is affine in t, Proposition 4 establishes that it suffices to consider the two endpoints $t \in \{\underline{f}_b, \overline{f}_b\}$. If both endpoint JRCs are nonnegative, path r cannot yield an improving column and need not be added to the RMP. Thus, physical pricing is complete when this condition holds for every physical path.

Moreover, the JRC can be decomposed into an initial cost $\tilde{c}^{\mathrm{B,init}}_{bt} = c^{\mathrm{O}}_b + t(c^{\mathrm{I}}_b - \sigma_b) - \pi_b - \kappa_b \mu^{\mathrm{T}}_{i_b} + \sum_{k \in \mathcal{K}_b} \gamma_{kb}$ and an arc extension cost

$$\tilde{c}^{\mathrm{B}}_{bta} = t \left(c^{\mathrm{T}}_{ba} - \mu_{e_a} \right), \tag{9}$$

Using Eq. (9), the JRC of physical path r is $\tilde{c}^{\mathrm{B}}_{br}(t) = \tilde{c}^{\mathrm{B,init}}_{bt} + \sum_{a \in \mathcal{A}} \delta_{ar} \tilde{c}^{\mathrm{B}}_{bta}$. Since the initial cost is independent of r and the arc extension costs are proportional to t, both endpoint JRCs rank physical paths identically. Thus, one search at $\underline{f}_b$ identifies a minimum-JRC path for both loads, and its JRC at $\overline{f}_b$ can be evaluated directly.

We use a labeling algorithm to find minimum-JRC elementary physical paths. For each block b with $\overline{f}_b \ge \underline{f}_b$, since $\mathcal{R}_b = \emptyset$ otherwise, a label $L = (n, S, d, c)$ records the current node n, visited nodes S, accumulated physical length d, and JRC c. We initialize $L_0 = (i_b, \{i_b\}, 0, \tilde{c}^{\mathrm{B,init}}_{bt})$. For each arc $a \in \mathcal{A}^+(n)$ with $j_a \notin S$, the label extends to $L' = (j_a, S \cup \{j_a\}, d + D_a, c + \tilde{c}^{\mathrm{B}}_{bta})$, subject to the length budget $\overline{D}_b$. A label reaching j_b represents a complete physical path with JRC c. The dominance rule used to reduce the labeling search space is given in Proposition 2.

PROPOSITION 2. *Fix block b, load t, and the current dual prices. For feasible labels $L_1 = (n, S_1, d_1, c_1)$ and $L_2 = (n, S_2, d_2, c_2)$ at the same node, write $L_1 \preceq L_2$ if $S_1 \subseteq S_2$, $d_1 \le d_2$, and $c_1 \le c_2$. Then L_1 dominates L_2, and discarding L_2 while retaining L_1 preserves the minimum JRC over all feasible physical paths for block b at load t.*

Proof. See Section EC.1.2 of the electronic companion. □

To further reduce the search space, we use shortest-path distances to identify labels that cannot satisfy the physical path length budget. Let $\underline{D}_b(n)$ denote the shortest directed length from node n to j_b, ignoring visited nodes, with value $+\infty$ if unreachable. These distances are computed using Dijkstra's algorithm on the reversed physical graph. We discard a label when $d + \underline{D}_b(n) > \overline{D}_b$ and reject an extension along arc a when $d + D_a + \underline{D}_b(j_a) > \overline{D}_b$. Proposition 3 establishes the validity of these tests. Since the distances depend only on the physical graph and destination, they can be reused across pricing calls.

PROPOSITION 3. *The distances $\underline{D}_b(n)$ are lower bounds on every feasible suffix length. The resulting prefix and extension tests discard only prefixes without a feasible completion and preserve the minimum physical path JRC.*

Proof. See Section EC.1.3 of the electronic companion. □

4.2.3. Overall Procedure. Algorithm 1 solves the RMP to optimality and uses its dual prices for exact logical and physical pricing. The returned sets $\mathcal{P}^+$ and $\mathcal{R}^+$ contain logical pairs (k, p) and physical pairs (b, r) with negative JRC. Each path is added once with its selection and volume variables, local bounds, and coupling coefficients. We reoptimize and repeat until both sets are empty.

Algorithm 1 Exact Column Generation

```
Initialize a feasible RMP.
repeat
    Solve the RMP and obtain dual prices.
    P+ ← ExactLogicalPricing.
    R+ ← ExactPhysicalPricing.
    Add the generated paths and associated constraints to the RMP.
until P+ = ∅ and R+ = ∅
return optimal DW LP solution and dual prices.
```

PROPOSITION 4. *Suppose the RMP is solved to optimality. If $\widetilde{c}^{\mathrm{D}}_{kp} \geq 0$ for every demand k and logical path $p \in \mathcal{P}_k$, and $\widetilde{c}^{\mathrm{B}}_{br}(t) \geq 0$ for every block b, physical path $r \in \mathcal{R}_b$, and $t \in \{\underline{f}_b, \overline{f}_b\}$, then the RMP solution is optimal for the DW LP relaxation, and its objective value provides a valid lower bound on z^*_{dw}.*

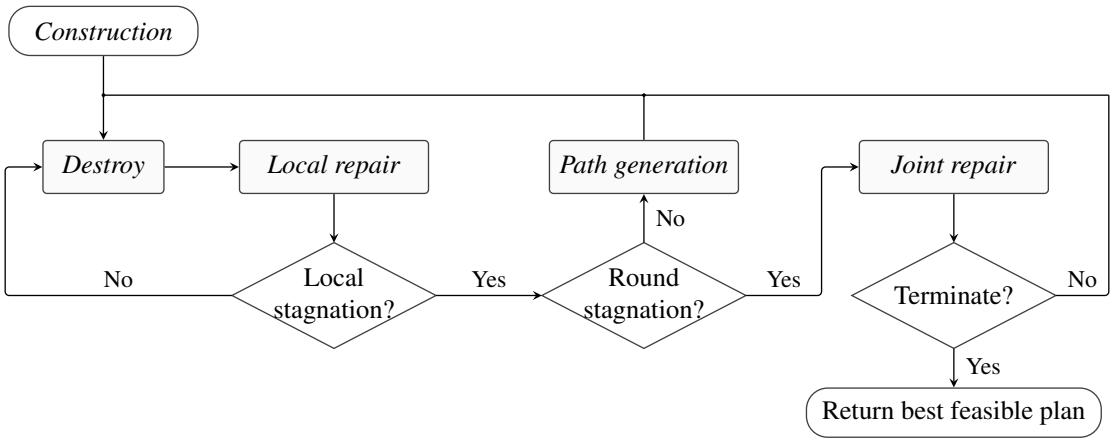
Proof. See Section EC.1.4 of the electronic companion. □

5. Dual-Guided Column-Generation-Based Matheuristic

DG-CGM improves feasible blocking plans by combining restricted MIP-based neighborhood search with dual-guided *path generation*. The framework consists of five components, as illustrated in Figure 3. First, *construction* builds an initial feasible plan and candidate physical paths (Section 5.1). The algorithm then repeatedly applies *destroy* and *local repair* to reconsider demand assignments and associated physical paths (Sections 5.2 and 5.3). When local improvement stagnates, *path generation* uses restricted DW LPs and

logical pricing to introduce additional logical paths for repair (Section 5.4). If several rounds of *path generation* and local search yield no improvement, *joint repair* combines the accumulated paths across demand groups and reallocates shared resources (Section 5.5). Both procedures return the search to *destroy* and *local repair*. The process terminates when the time limit is reached or a complete cycle yields no new LP or MIP solve and no new candidate path, returning the best feasible solution found.

Figure 3 The framework of DG-CGM.



5.1. *Construction*

Let ξ denote a feasible blocking plan and $z(\xi)$ its total cost. Algorithm 2 summarizes the *construction* procedure, starting from the all-unserved feasible plan. Each iteration processes an ordered sequence $\mathcal{G}$ of demand sets, beginning with individual demands and then considering groups of demands for freight consolidation (Section 5.1.1). For each demand set $\mathcal{K}_{\text{cand}} \subseteq \mathcal{K}$, CANDIDATES generates candidate assignments specifying the demands to adjust, their logical paths, and target served volumes (Section 5.1.2). SERVE then evaluates these candidates by determining feasible physical paths and actual served volumes, accepting the lowest-cost feasible plan if it improves the incumbent (Section 5.1.3). The procedure first operates over a restricted block domain and then expands the domain while retaining the constructed plan. Within each domain, iterations continue until the relative cost improvement $(z(\bar{\xi}) - z(\xi))/\max\{1, |z(\bar{\xi})|\} \le \varepsilon_{\text{C}}$, where $\bar{\xi}$ is the incumbent at the start of the iteration and $\varepsilon_{\text{C}} \in [0, 1)$ is the stopping tolerance.

5.1.1. *Demand Sets.* Demand grouping allows multiple demands to be considered jointly, creating opportunities for freight consolidation and satisfying minimum block loads. Accordingly, the sequence $\mathcal{G}$ comprises four types of demand sets, ordered by increasing complexity of their logical paths. *Singleton sets* $\{k\}$ are processed first in input order. *Direct groups* collect, for each candidate block b, demands satisfying $o_k = i_b$, $d_k = j_b$, and $b \in \mathcal{B}_k$. *One-hub groups* collect demands that admit compatible logical paths $o_k \to h \to d_k$ through a common intermediate yard h, while *two-hub groups* consider a common ordered pair of distinct intermediate yards (h_1, h_2), following $o_k \to h_1 \to h_2 \to d_k$.

To limit the number of two-hub groups, we rank yards by $s_h = \sum_{k:\, h \in \{o_k, d_k\}} q_k c_k^{\text{U}}$, which measures the total potential unserved demand cost associated with yard h. We form all ordered pairs of distinct yards

Algorithm 2 *Construction*

$\xi \leftarrow$ all-unserved feasible plan.
for all block domains, from restricted to expanded **do**
  repeat
    $\bar{\xi} \leftarrow \xi$.
    Form the ordered demand sets $\mathcal{G}$. ▷ Section 5.1.1
    for all $\mathcal{K}_{\text{cand}}$ in $\mathcal{G}$, in order **do**
      CANDIDATES: generate candidate assignments for $\mathcal{K}_{\text{cand}}$. ▷ Section 5.1.2
      SERVE: evaluate candidate assignments and update ξ. ▷ Section 5.1.3
    end for
  until $(z(\bar{\xi}) - z(\xi))/\max\{1, |z(\bar{\xi})|\} \le \varepsilon_{\text{C}}$
end for
return ξ.

among the N_{hub} highest-ranked yards, then add pairs (i_b, j_b) from available blocks whose endpoints match the origin and destination of at least one demand, excluding pairs already considered.

5.1.2. *Candidate Assignments.* During *construction*, we use the *estimated unit cost* (EUC) to rank candidate logical paths. We first define the block-level EUC c^{C}_{kb}, which estimates the cost per unit of demand k transported on block b. Let $\bar{y}_b$ denote the value of y_b in the current solution, and let r^{ref}_b be the current physical path of an open block or a retained physical path with minimum unit cost for an unopened block. The block-level EUC is

$$c^{\text{C}}_{kb} = c^{\text{B}}_{b,r^{\text{ref}}_b} + (1 - \bar{y}_b)\frac{c^{\text{O}}_b}{q_k} + \mathbf{1}\{i_b \neq o_k\} c^{\text{H}}_{i_b}. \tag{10}$$

The three terms account for transportation and interchange costs, the opening cost of an unopened block allocated over q_k, and intermediate-yard handling costs, respectively. Since a logical path consists of a sequence of blocks, we define its EUC as $\widehat{c}_k(p) = \sum_{b \in p} c^{\text{C}}_{kb}$, which estimates the unit cost of serving demand k along path p.

For each demand set $\mathcal{K}_{\text{cand}}$, CANDIDATES generates assignments of the form $\{(k, p_k, q_k) : k \in U\}$, where $\emptyset \neq U \subseteq \mathcal{K}_{\text{cand}}$ contains the demands to be reassigned, $p_k \in \mathcal{P}_k$ is the proposed logical path, and q_k is the target served volume. For a singleton set $\mathcal{K}_{\text{cand}} = \{k\}$, we enumerate compatible direct and one-hub paths, followed by minimum-EUC path searches over all available compatible blocks and, separately, over currently open blocks. For direct groups, each demand uses the designated block, while one-hub and two-hub groups follow their prescribed yard sequences using compatible blocks. Physical path candidates are generated as needed through shortest-path searches (see Section EC.2.1).

When $\mathcal{K}_{\text{cand}}$ is a multi-demand group, we construct candidate assignments by ranking demands according to the estimated unit savings of their proposed logical paths. Let $\bar{v}_k$ denote the incumbent served volume and $\bar{c}_k$ the unit variable cost of its incumbent logical and physical paths, excluding block opening costs, with $\bar{c}_k = 0$ for unserved demands. The estimated unit saving is

$$\Delta_k = \frac{\bar{v}_k \bar{c}_k + (q_k - \bar{v}_k) c^{\text{U}}_k}{q_k} - \widehat{c}_k(p_k). \tag{11}$$

We order the demands $k_1, \ldots, k_{|\mathcal{K}_{\text{cand}}|}$ by decreasing Δ_k and consider progressively larger subsets $\{k_1, k_2\}, \ldots, \{k_1, \ldots, k_{|\mathcal{K}_{\text{cand}}|}\}$, each with its proposed logical paths and target volumes. For each subset U, replacing the incumbent assignments with the proposed changes gives the block loads $f_b^{\text{tar}}(U) = f_b(\xi) - \sum_{k \in U} \iota_{b\bar{p}_k} \bar{v}_k + \sum_{k \in U} \iota_{bp_k} q_k$, where $\bar{p}_k$ is the incumbent logical path of demand k. To satisfy minimum block loads, we iteratively update

$$U \leftarrow \{k \in U : f_b^{\text{tar}}(U) = 0 \text{ or } f_b^{\text{tar}}(U) \geq \underline{f}_b,\ \forall b \in \bar{p}_k \cup p_k\}. \tag{12}$$

We repeat Eq. (12) until U remains unchanged. Each nonempty resulting subset forms a candidate assignment for SERVE, which evaluates its feasibility and total cost.

5.1.3. *Assignment Evaluation.* For each candidate assignment $\{(k, p_k, q_k) : k \in U\}$, SERVE fixes the proposed logical paths of demands in U and retains the incumbent assignments of all other demands. It also retains the incumbent physical paths of open blocks, selects feasible physical paths for newly opened blocks, and scales the target served volumes q_k by a common factor $\theta \in [0, 1]$. Given these path choices, the block load bounds and resource capacities determine a feasible interval $[\theta_-, \theta_+]$ for θ. Since the total cost is affine in θ, taking the form $z_0 + \theta z_1$, the minimum-cost served volumes are

$$v_k = \theta^\star q_k \quad (k \in U), \qquad \theta^\star = \begin{cases} \theta_+, & z_1 < 0, \\ \theta_-, & z_1 \geq 0. \end{cases} \tag{13}$$

Using Eq. (13), SERVE evaluates the feasible plans obtained from all candidate assignments and accepts the lowest-cost plan only if it improves the incumbent. The resulting blocking plan provides the starting point for *destroy* and *local repair*, described next.

5.2. *Destroy*

Destroy selects a group of demands $\mathcal{K}_{\text{free}} \subseteq \mathcal{K}$ for *local repair*, which jointly reoptimizes their logical paths and served volumes while keeping all other demand assignments fixed. The selection proceeds in two steps. First, we choose a reference demand k_0. During the first outer cycle, successive calls select k_0 in decreasing order of its current unserved demand cost $(q_k - \bar{v}_k)c_k^{\text{U}}$, while subsequent cycles select k_0 uniformly at random. Second, we rank demands according to their relationships with k_0 and their unserved demand costs using the priority score

$$\psi_k = \omega_0 \mathbf{1}\{k = k_0\} + \omega_1 \mathbf{1}\{o_k = o_{k_0} \text{ or } d_k = d_{k_0}\} + \omega_2 m_k + \frac{(q_k - \bar{v}_k)c_k^{\text{U}}}{\max\{1, \max_{j \in \mathcal{K}}(q_j - \bar{v}_j)c_j^{\text{U}}\}}, \tag{14}$$

where $m_k = \sum_{b \in \mathcal{B}_k} \bar{x}_{kb}^{\text{D}} \mathbf{1}\{i_b = o_{k_0} \text{ or } j_b = d_{k_0}\}$ counts the blocks on the incumbent logical path of demand k that depart from the reference origin or arrive at the reference destination. The weights $\omega_0, \omega_1, \omega_2$ set the selection priorities.

Given the current group size limit $N_{\text{D}} \leq |\mathcal{K}|$, we select the N_{D} demands with the highest scores in Eq. (14) to form $\mathcal{K}_{\text{free}}$. Ties are broken by candidate order, which is randomized in subsequent outer cycles. The group size schedule and priority weights are specified in Section EC.2.2 and Table EC.1.

5.3. *Local Repair*

Local repair constructs a restricted MIP of the DW formulation (2) to jointly reoptimize the logical paths and served volumes of demands in $\mathcal{K}_{\text{free}}$, together with the physical paths of affected blocks. Assignments of other demands remain fixed.

For each demand $k \in \mathcal{K}_{\text{free}}$, let $\widetilde{\mathcal{P}}_k \subseteq \mathcal{P}_k$ denote its candidate logical path set, including the incumbent path, if any, paths retained from *path generation* and *joint repair*, and low-EUC paths generated using the procedures in Section 5.1.2. After consecutive calls without improvement, we increase the group size to allow more joint reassignments. We also consider bridge paths connecting chains of open blocks through additional blocks, allowing released demands to reuse existing services. The corresponding settings are given in Section EC.2.2.

The blocks whose physical paths may change are given by $\mathcal{B}_{\text{free}} = \bigcup_{k \in \mathcal{K}_{\text{free}}} \bigcup_{p \in \widetilde{\mathcal{P}}_k} \{b : \iota_{bp} = 1\}$. Let $\mathcal{K}_{\text{fixed}} = \mathcal{K} \setminus \mathcal{K}_{\text{free}}$ and $\mathcal{B}_{\text{fixed}} = \mathcal{B} \setminus \mathcal{B}_{\text{free}}$ denote the remaining demands and blocks. For each block b, let $\widetilde{\mathcal{R}}_b \subseteq \mathcal{R}_b$ denote its retained physical path set. The path sets of fixed demands and blocks contain only their incumbent paths, when present. We substitute the fixed decisions into the DW block balance and resource capacity constraints to obtain

$$
\begin{aligned}
\sum_{r \in \widetilde{\mathcal{R}}_b} v^{\mathrm{B}}_{br} - \sum_{\substack{k \in \mathcal{K}_{\text{free}} \\ p \in \widetilde{\mathcal{P}}_k}} \iota_{bp} v^{\mathrm{D}}_{kp} &= \sum_{\substack{k \in \mathcal{K}_{\text{fixed}} \\ p \in \widetilde{\mathcal{P}}_k}} \iota_{bp} \bar{v}^{\mathrm{D}}_{kp}, && b \in \mathcal{B}_{\text{free}}, \\
\sum_{\substack{b \in \mathcal{B}_{\text{free}} \\ r \in \widetilde{\mathcal{R}}_b}} \Big(\sum_{a \in \mathcal{A}_e} \delta_{ar} \Big) v^{\mathrm{B}}_{br} &\le C_e - \sum_{\substack{b \in \mathcal{B}_{\text{fixed}} \\ r \in \widetilde{\mathcal{R}}_b}} \Big(\sum_{a \in \mathcal{A}_e} \delta_{ar} \Big) \bar{v}^{\mathrm{B}}_{br}, && e \in \mathcal{E}, \\
\sum_{\substack{k \in \mathcal{K}_{\text{free}} \\ p \in \widetilde{\mathcal{P}}_k}} \eta_{ip} v^{\mathrm{D}}_{kp} &\le H_i - \sum_{\substack{k \in \mathcal{K}_{\text{fixed}} \\ p \in \widetilde{\mathcal{P}}_k}} \eta_{ip} \bar{v}^{\mathrm{D}}_{kp}, && i \in \mathcal{Y}, \\
\sum_{\substack{b \in \mathcal{B}_{\text{free}} \\ i_b = i}} \kappa_b \sum_{r \in \widetilde{\mathcal{R}}_b} x^{\mathrm{B}}_{br} &\le T_i - \sum_{\substack{b \in \mathcal{B}_{\text{fixed}} \\ i_b = i}} \kappa_b \sum_{r \in \widetilde{\mathcal{R}}_b} \bar{x}^{\mathrm{B}}_{br}, && i \in \mathcal{Y}.
\end{aligned}
\tag{15}
$$

In Eq. (15), the physical path variables of each free block account for its full load, including contributions from fixed demands. All other applicable DW constraints, including service activation, remain enforced after substitution.

5.4. *Path Generation*

Path generation identifies additional logical paths for *local repair* and *joint repair* using dual information from restricted DW LPs. It follows the column generation framework in Section 4.2, with heuristic logical pricing and a restricted search graph. All demands remain free in the RMP, while the retained physical path sets remain unchanged.

5.4.1. *Heuristic Logical Path Pricing. Path generation* focuses on logical paths because additional physical paths offer limited benefit in our instances. Transportation costs are proportional to physical distance, making shortest physical paths preferable, while physical link capacities rarely become binding.

Nevertheless, some physical paths may be useful when capacity constraints become binding. In this case, we generate them during *joint repair*, restricting the search to open blocks to limit computational cost.

We initialize the RMP with at most N_{LP} logical paths per demand, retaining the incumbent path, if any, and selecting the remaining paths in increasing EUC order, as defined in Section 5.1.2. All applicable coupling constraints, including service activation, are enforced. To improve pricing efficiency, we modify the labeling algorithm in Section 4.2.1 in two ways. First, for labels $L_1 = (i, S_1, c_1)$ and $L_2 = (i, S_2, c_2)$ at the same yard, we use the heuristic dominance rule

$$L_1 \preceq_{\mathrm{H}} L_2 \quad \Longleftrightarrow \quad c_1 \le c_2. \tag{16}$$

Unlike Proposition 1, Eq. (16) omits the visited-set inclusion condition $S_1 \subseteq S_2$. We still maintain visited-yard sets to ensure elementary paths, but the heuristic dominance rule no longer guarantees finding a minimum-JRC path.

Second, we restrict the logical pricing graph by selecting successor yards based on the retained physical paths. For demand k, define

$$d_k(i,j) = \min_{\substack{b \in \mathcal{B}_k^+(i):\, j_b = j \\ r \in \widetilde{\mathcal{R}}_b}} \sum_{a \in r} D_a, \tag{17}$$

with $d_k(i,j) = +\infty$ if no eligible block and physical path exist. At each yard i, we select up to N_{nbr} successor yards with the smallest distances in Eq. (17), retaining all compatible blocks with physical path candidates leading to these yards. Compatible direct blocks and blocks on the incumbent logical path are also retained.

After each optimal RMP solve, heuristic pricing uses the current dual prices to identify logical paths with negative JRC, as defined in Eq. (5). We add up to N_{batch} new paths across all demands, together with their selection and volume variables and the corresponding service-activation constraints, before reoptimizing the RMP. The process terminates when no new paths are found, LP improvement stagnates, or the computational limits are reached. Because pricing is heuristic, termination does not certify optimality of the complete DW relaxation.

5.4.2. *Logical Path Selection.* After the final RMP solve, we consider the newly generated logical paths represented in the RMP that were absent from the shared path pool before the search. For each demand, we retain up to N_{add} paths with the lowest JRCs based on the final dual prices.

If computational limits terminate the search before convergence, we select paths using three criteria: JRC, resource diversity, and EUC. For each demand k, we first retain up to N_{JRC} paths with the lowest JRCs. Let $\mathcal{C}_k$ denote the newly generated paths in the final RMP and $\mathcal{C}_k^{\mathrm{sel}} \subseteq \mathcal{C}_k$ the paths selected so far. For each path p, let $\mathcal{U}_{\mathrm{res}}(p)$ contain its blocks, departure yards, and physical links along their reference physical paths. We measure resource diversity using the Jaccard distance (Jaccard 1912) and select each additional path by

$$\begin{aligned} d_{\mathrm{res}}(p,p') &= 1 - \frac{|\mathcal{U}_{\mathrm{res}}(p) \cap \mathcal{U}_{\mathrm{res}}(p')|}{|\mathcal{U}_{\mathrm{res}}(p) \cup \mathcal{U}_{\mathrm{res}}(p')|}, \\ p^* &\in \arg \max_{p \in \mathcal{C}_k \setminus \mathcal{C}_k^{\mathrm{sel}}} \min_{p' \in \mathcal{C}_k^{\mathrm{sel}}} d_{\mathrm{res}}(p,p'), \end{aligned} \tag{18}$$

with lower JRC breaking ties. Using Eq. (18), we retain up to N_{div} such paths, followed by up to N_{EUC} additional paths in increasing EUC order, where $N_{\mathrm{JRC}} + N_{\mathrm{div}} + N_{\mathrm{EUC}} = N_{\mathrm{add}}$. The EUC search considers compatible logical paths supported by retained physical paths, including those absent from the RMP. All selected paths are added to the shared pool without removing existing paths, allowing subsequent *local repair* and *joint repair* calls to use them.

5.5. *Joint Repair*

Joint repair combines logical and physical paths accumulated across different demand groups by jointly reoptimizing a larger set of demands. Before each call, we expand the retained path sets using the procedures of *local repair*, including bridge paths, and generate additional physical paths for open blocks. We also consider *partner chains*, which allow demands to jointly support the minimum load of an unopened block (Section EC.2.3). To control the MIP size, at most N_{J} demands are released. When $|\mathcal{K}| \le N_{\mathrm{J}}$, we simply set $\mathcal{K}_{\mathrm{free}} = \mathcal{K}$.

For larger instances, we select demands based on their unserved demand costs, potential EUC savings, and contributions to current block opening costs. Let $\bar{p}_k$ denote the incumbent logical path of demand k, with $\bar{p}_k = \emptyset$ if unserved. For each demand with at least one retained logical path, we define

$$\rho_k = (q_k - \bar{v}_k)c_k^{\mathrm{U}} + \bar{v}_k \max\left\{0, \widehat{c}_k(\bar{p}_k) - \min_{p \in \widetilde{\mathcal{P}}_k} \widehat{c}_k(p)\right\} + \sum_{b \in \bar{p}_k} \frac{c_b^{\mathrm{O}} \bar{v}_k}{f_b(\xi)}, \tag{19}$$

where the last two terms are zero for unserved demands.

We construct five types of candidate groups. The first three follow the grouping principles of *destroy* and *construction*, comprising current users of a common open block, demands whose minimum-EUC paths share a block with sufficient combined volume to meet its minimum load, and singleton demands with retained paths. The remaining two target shared resource constraints. One groups demands whose incumbent blocks use a common saturated physical link, while the other pairs a demand whose direct block is prevented by insufficient classification tracks with users of an open block whose closure could release the required tracks. For the latter, we consider the highest-scoring eligible demand at each yard and the blocks whose closure would provide sufficient track capacity.

To avoid leaving insufficient loads on blocks used by fixed demands, we complete each candidate group using a minimum-load closure. For a demand set U, define $f_b^-(U) = f_b(\xi) - \sum_{k \in U \cap \mathcal{K}_b} g_{kb}(\xi)$. Starting from $U^{(0)} = \mathcal{K}_{\mathrm{cand}}$, we iteratively update

$$U^{(j+1)} = U^{(j)} \cup \bigcup_{\substack{b \in \mathcal{B}: \\ 0 < f_b^-(U^{(j)}) < \underline{f}_b}} \{k \in \mathcal{K}_b : g_{kb}(\xi) > 0\} \tag{20}$$

We repeat Eq. (20) until the set remains unchanged. Using the demand scores in Eq. (19), we rank the resulting groups by decreasing average score $\rho(\mathcal{K}_{\mathrm{cand}}) = |\mathcal{K}_{\mathrm{cand}}|^{-1} \sum_{k \in \mathcal{K}_{\mathrm{cand}}} \rho_k$, breaking ties by total score and demand index order. Starting with $\mathcal{K}_{\mathrm{free}} = \emptyset$, we process groups in this order and add the closure of $\mathcal{K}_{\mathrm{free}} \cup \mathcal{K}_{\mathrm{cand}}$ whenever its size does not exceed N_{J}; otherwise, the group is skipped.

Given $\mathcal{K}_{\text{free}}$, the joint MIP uses all retained logical paths of the released demands and physical paths of their associated blocks. Assignments outside the selected group remain fixed, and the model construction and solution procedures follow *local repair* in Section 5.3. When the model size permits, we additionally allow at most one released demand to select a logical path outside the retained pool through the compact constraints in Eqs. (1b)–(1h), using compatible blocks with retained physical paths. The free block set is expanded accordingly, and any newly selected path is retained if it improves the solution. Section EC.2.4 describes this extension. After the joint MIP solve, the algorithm returns to *destroy* and *local repair*.

6. Computational Experiments

Section 6.1 describes the common experimental setup. On the small generated instances (Section 6.2), we compare solution quality and computational effort against exact benchmarks, evaluate the search components, and assess the robustness of DG-CGM. Section 6.3 uses larger generated instances to study how performance varies with the number of demands and network size. On the large RAS instances (Section 6.4), we evaluate the search components at full scale and examine how demand volume, demand patterns, and resource limits affect service, consolidation, and resource use.

6.1. Experimental Setup

6.1.1. Implementation and Computing Environment. All methods are implemented in C++17, compiled with Microsoft Visual C++ 14.44 (Visual Studio 2022), and run on a Windows workstation equipped with AMD EPYC 9654 processors at 2.4 GHz and 1.5 TiB of RAM. Gurobi 12.0.3 (Gurobi Optimization, LLC 2026) solves the LPs and MIPs through the RouteOpt interface (You and Yang 2026), and the local and joint MIPs of DG-CGM use `IntegralityFocus=1`. The ARC LP, ARC MIP, and DW LP benchmarks use eight solver threads per run, with eight pricing workers for the DW LP. Each benchmark run has a two-hour time limit. Unless stated otherwise, DG-CGM uses the default controls in Table EC.1 and, on generated instances, one solver thread, random seed 1, and a 60-second time limit, including *construction*. The six RAS scenarios use eight solver threads and a two-hour time limit, including *construction*, and each scenario is run with random seeds 1, 2, and 3. All runs share the 384 logical processors of the workstation, and a new run starts only when at most 185 are busy. The ARC LP and DW LP times in Table 2 are measured separately, with at most four concurrent runs. All runtimes are reported as wall-clock times. The code, the generated instances, the configuration of every experiment, and the results of all runs are publicly available at `https://github.com/Zhengzhong-You/IRBP`.

ARC LP relaxes the binary variables of the ARC formulation, while ARC MIP uses the default Gurobi cut settings. We solve the LP relaxation of the DW formulation (2) using logical and physical pricing, as described in Section 4.2. Service-activation constraints (2j) are separated as needed, with zero dual variables assigned to constraints not yet added. All three exact benchmarks use the original instance data and do not receive solutions from DG-CGM.

6.1.2. Instance Sets. For the operational study, we use data from the 2026 RAS Problem Solving Competition (INFORMS Railway Applications Section 2026). The L1, L2, and L3 data sets share the same physical network with 47,193 nodes, 106,570 directed arcs, and identical resource capacities. L1 and L2 contain the same 2,044 demand records up to identifiers and ordering, so we select L1 and L3. L3 contains 2,427 demands, 18.7% more than L1, and 0.3% more total requested volume, so L1 and L3 have nearly the same volume but different demand patterns. For each data set, we apply demand multipliers $\alpha \in \{0.5, 1, 2\}$ to every q_k and the corresponding block volume upper bounds derived from these quantities. Minimum block loads, resource capacities, and cost coefficients remain unchanged. Table 1 summarizes the instances.

The generated instances reproduce the regional connectivity, degree distribution, travel distances, yard operating characteristics, freight composition, and demand volume distributions of the RAS data. Section EC.3 of the electronic companion describes the generation procedure. We generate 50 instances with 12 demands at each network size $|\mathcal{N}| \in \{50, 100, 150\}$, denoted as N50, N100, and N150. To examine larger instances, we generate four additional sets following the same procedure and retain for each set the first ten valid instances.

Table 1 Instance sets and their roles in the numerical study.

Source	Set	#Inst.	$\lvert\mathcal{N}\rvert$	$\lvert\mathcal{Y}\rvert$	$\lvert\mathcal{A}\rvert$	$\lvert\mathcal{K}\rvert$	Role
Generated	N50	50	50	10.4	153.1	12	Mechanisms, solution quality, and computational effort
	N100	50	100	18.3	318.4	12	
	N150	50	150	27.9	487.3	12	
	N150K24	10	150	27.4	485.0	24	Scalability
	N150K48	10	150	27.3	487.0	48	
	N300K48	10	300	55.0	985.0	48	
	N600K48	10	600	108.9	1,997.0	48	
RAS	L1×0.5	1	47,193	1,477	106,570	2,044	Low demand
	L1×1	1	47,193	1,477	106,570	2,044	Base demand
	L1×2	1	47,193	1,477	106,570	2,044	High demand
	L3×0.5	1	47,193	1,477	106,570	2,427	Low demand
	L3×1	1	47,193	1,477	106,570	2,427	Base demand
	L3×2	1	47,193	1,477	106,570	2,427	High demand

Note. For generated instances, $|\mathcal{Y}|$ and $|\mathcal{A}|$ are arithmetic means over the instances of each set.

6.1.3. Performance Measures. For each of the 150 small generated instances, we compare the cost z of a feasible plan with a common reference cost z^{ref}, equal to the optimal cost when ARC MIP proves optimality and otherwise to the lowest feasible cost obtained by any ARC MIP or DG-CGM run in Section 6.2. The relative difference $(z - z^{\mathrm{ref}})/z^{\mathrm{ref}}$ is an optimality gap when the reference is proven optimal and otherwise measures the deviation from the reference. For a valid lower bound $\underline{z}$, the bound gap is $(z - \underline{z})/z$. We denote the cost immediately after *construction*, before any *local repair*, by z^0, and the final cost by z^{H}. The improvement over *construction* is $(z^0 - z^{\mathrm{H}})/z^0$. The volume-weighted service

ratio is $\sum_k v_k / \sum_k q_k$. Across instances, we report arithmetic means for costs, gaps, improvements, and service ratios, and geometric means for positive runtimes unless stated otherwise. For paired comparisons on the same instances, we report Win/Tie/Loss counts and, where indicated, the p-value from a two-sided Wilcoxon signed-rank test (Wilcoxon 1945) applied to paired cost differences. The test uses the signs and ranks of these differences to assess whether their distribution is symmetric about zero. Smaller p-values indicate stronger statistical evidence of a cost difference.

6.2. Small Generated Instances

6.2.1. Bound Strength and DG-CGM Performance. Table 2 compares the lower bounds and computation times of the ARC and DW formulations. Across all 150 instances, the mean lower-bound gaps relative to z^{ref} are 67.5% for ARC and 3.1% for DW, while the corresponding geometric mean computation times are 19.3 and 1.3 seconds. The substantially tighter DW bounds and shorter computation times support our use of the DW formulation to identify promising logical paths in Section 5.

Table 2 LP bounds and computation times for the ARC and DW formulations.

		ARC LP			DW LP			ARC MIP	
Set	$z^{\mathrm{ref}}/10^6$	$\underline{z}_{\mathrm{arc}}/10^6$	Gap/%	Time/s	$\underline{z}_{\mathrm{dw}}/10^6$	Gap/%	Time/s	Time/s	#Opt.
N50	17.2	4.9	69.8	2.1	16.8	2.6	0.1	52.1	50
N100	17.5	6.3	64.6	23.9	16.9	3.4	1.5	2,790.1	32
N150	17.3	5.5	68.0	146.7	16.7	3.3	10.0	6,365.9	9
All	–	–	67.5	19.3	–	3.1	1.3	974.2	91

Table 3 Feasible solution quality and runtime of ARC MIP and DG-CGM.

Set	Method	$z/10^6$	Max. gap/%	Mean gap/%	#Opt.	Mean improvement/%	Win/Tie/Loss	Time/s
N50	ARC MIP	17.2	0.0	0.0	50	–	–	52.1
	DG-CGM	17.2	2.9	0.1	48	−0.1	0/48/2	36.7
N100	ARC MIP	17.5	1.8	0.1	32	–	–	2,790.1
	DG-CGM	17.5	1.5	< 0.1	32	0.0	4/44/2	59.5
N150	ARC MIP	19.6	67.9	13.5	9	–	–	6,365.9
	DG-CGM	17.3	5.8	0.3	9	10.2	39/11/0	60.1
All	ARC MIP	–	67.9	4.5	91	–	–	974.2
	DG-CGM	–	5.8	0.1	89	3.4	43/103/4	50.8

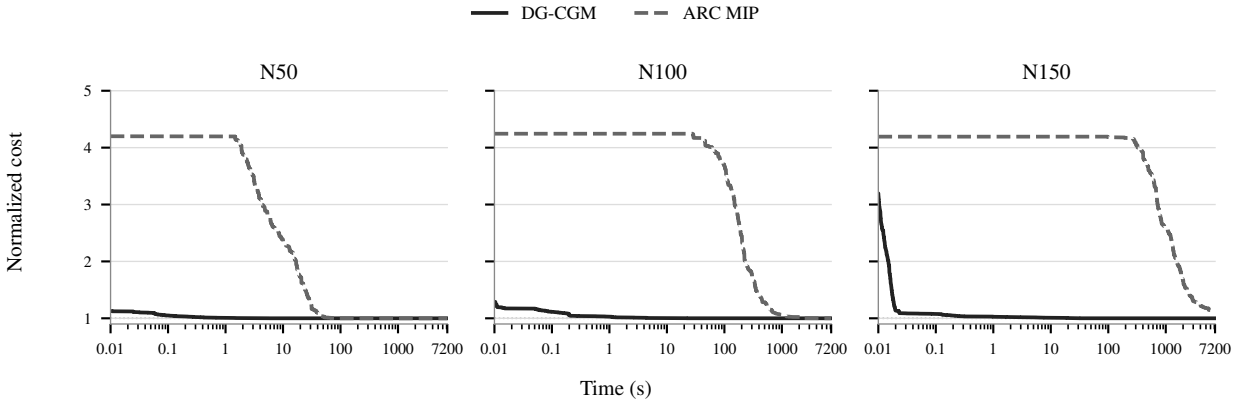
Note. One N100 instance is excluded from the proven-optimality count because a lower-cost feasible solution was found than Gurobi's reported optimum.

Table 3 compares the best feasible solutions obtained by ARC MIP and DG-CGM, with gaps relative to z^{ref}. Among the 91 instances with proven optima, DG-CGM achieves the optimal cost on 89. Across all 150 instances, DG-CGM lowers the ARC MIP cost by 3.4% on average while using 5.2% of its geometric mean runtime. DG-CGM obtains lower costs on 43 instances, by up to 40.4%, and higher costs on four, by at most 2.9%. These four instances belong to N50 and N100, where exact optimization is most effective; on N50, ARC MIP proves optimality for every instance with a geometric mean runtime of 52.1 seconds.

The two optima missed by DG-CGM are both on N50. It reaches one with a 300-second time limit but remains 0.7% above the other across all runs and time limits, suggesting that this optimal plan uses a path outside the candidate pools of DG-CGM. The advantage of DG-CGM grows with instance size. On the largest set, N150, DG-CGM lowers the ARC MIP cost by 10.2% on average while using 0.9% of the ARC MIP runtime, with 39 wins, 11 ties, and no losses.

Figure 4 shows the solution trajectories of DG-CGM and ARC MIP for the three instance sets. Most of the cost reduction by DG-CGM occurs within the first 0.1 seconds, with further improvements as the search continues. After one second, the mean excess over the final DG-CGM cost is 0.7%, 3.0%, and 2.7% for N50, N100, and N150, respectively. On N150, this one-second excess is smaller than the 13.1% excess of ARC MIP at the two-hour time limit. Thus, DG-CGM obtains solutions close to its final cost within one second.

Figure 4 Mean incumbent cost relative to the final DG-CGM cost over time.



6.2.2. Contributions of the Search Components. We compare the complete DG-CGM with four variants, *Construction only*, *Without path generation*, *Without local repair*, and *Without joint repair*. The *Construction only* variant terminates after *construction*. Each of the other three variants skips the corresponding component whenever it would otherwise be invoked. Table 4 reports the same measures as Table 3, along with Win/Tie/Loss counts comparing DG-CGM with each variant.

Table 4 Component comparisons on the 150 small generated instances with a 60-second time limit.

Variant	$z/10^6$	Max. gap/%	Mean gap/%	#Opt.	Time/s	DG-CGM Win/Tie/Loss
Construction only	19.1	174.0	12.6	12	< 0.1	135/15/0
Without path generation	17.4	5.8	0.2	85	45.6	10/135/5
Without local repair	17.3	5.8	0.1	89	46.6	6/139/5
Without joint repair	17.4	9.0	0.4	71	11.7	42/107/1
DG-CGM	17.3	5.8	0.1	89	50.8	–

Joint repair has the largest effect. With 12 demands, each *joint repair* call releases all demands and solves the all-demand restricted MIP over the accumulated paths (Section 5.5). Compared with the variant without *joint repair*, DG-CGM obtains lower costs on 42 instances and higher costs on one, while the

variant matches 71 proven optima compared with 89 for DG-CGM (Wilcoxon $p < 0.001$). The effects of *path generation* and *local repair* depend on the time limit. With a 60-second time limit, the variant without *path generation* still matches 85 proven optima, with DG-CGM obtaining lower costs on ten instances and higher costs on five ($p = 0.45$); without *local repair*, the costs coincide on 139 instances ($p = 0.58$).

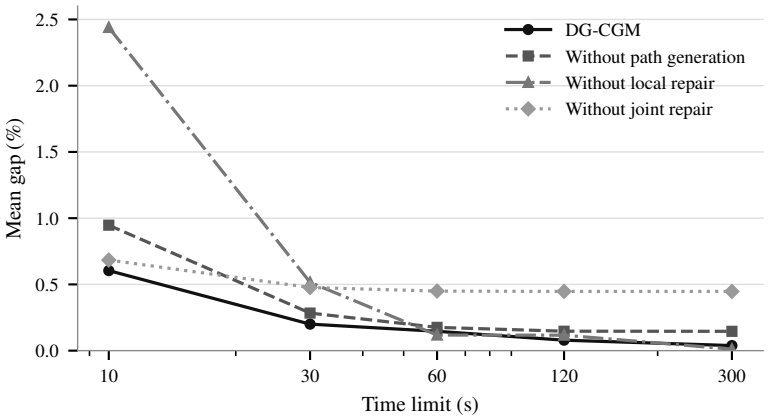
We then repeat the comparison with time limits of 10, 30, 120, and 300 seconds, keeping all other settings fixed. Figure 5 reports the mean gap. Removing *path generation* increases the mean gap at every time limit, from 0.60% to 0.95% at 10 seconds and from 0.04% to 0.15% at 300 seconds, with little further improvement after 120 seconds. Without *joint repair*, the search terminates after two consecutive pricing rounds without improvement; with a 300-second time limit, 136 of the 150 runs terminate early. Consequently, the mean gap of this variant remains between 0.45% and 0.48% from 30 seconds onward, whereas DG-CGM continues to improve, reaching 0.04% at 300 seconds. *Local repair* drives early improvement. With a 10-second time limit, removing it increases the mean gap from 0.60% to 2.44%. From 60 seconds onward, this variant performs about as well as DG-CGM; with 300 seconds, it obtains lower costs on eight instances and higher costs on one, with a mean gap of 0.01% instead of 0.04% (Wilcoxon $p = 0.01$). Skipping *local repair* leaves more time for *path generation* and *joint repair*, which this variant invokes on average 54.6 and 27.2 times per run with 300 seconds, compared with 42.6 and 20.9 times for DG-CGM. Because each *joint repair* call already releases all 12 demands, the neighborhoods of *local repair*, with eight or twelve released demands, largely overlap with those of *joint repair* when the time limit allows both to be applied. *Local repair* is therefore most useful under short time limits.

Path generation is not the only source of new paths in DG-CGM. *Joint repair* also adds candidate paths before and during each call (Section 5.5), which can substitute for paths supplied by *path generation*. To isolate the contribution of *path generation*, we restrict *joint repair* to incumbent paths and paths already in the shared pool, excluding its candidate supplements and arc options while keeping all other settings unchanged. Under this restriction, *path generation* becomes decisive. Removing it increases cost by 2.5% on average and by up to 53.7%, with DG-CGM obtaining lower costs on 56 instances and higher costs on none (Wilcoxon $p < 0.001$); the restricted variant matches 71 proven optima with *path generation* and 52 without. Without *path generation*, the restricted search terminates after 0.4 seconds (geometric mean) when local search stagnates, whereas with *path generation* it recovers 97% of the improvement achieved by the complete DG-CGM over *construction*.

6.2.3. Value of Dual Guidance. We compare three *path generation* procedures on all 150 instances. Each starts from the same plan immediately after *construction*, before *local repair*, with identical retained logical and physical paths. *DG-CGM path generation* follows the procedure in Section 5.4.

The two shortest-path procedures independently search the full compatible logical graph supported by the retained physical paths. *Transportation cost* assigns block b the weight $c^{\mathrm{B}}_{b,r^{\mathrm{ref}}_b} - c^{\mathrm{I}}_b$, equal to the sum

Figure 5 Mean gap of DG-CGM and its variants under different time limits.



of transportation costs along its reference physical path. *EUC* uses the block weights in Eq. (10), which also account for interchange cost, the allocated opening cost of an unopened block, and handling cost at intermediate yards. Both procedures enumerate elementary logical paths in increasing order of total weight. All three procedures use the same reference physical paths and fixed physical path pool, retain the initial logical paths, and may add at most twelve new paths per demand.

Each procedure is allotted five seconds for generation, followed by five seconds for integer repair, including model construction, optimization, and validation. Each repair releases all twelve demands and disables the arc options of *joint repair*. Generation time includes RMP preparation, restricted DW LP solves, pricing, and path selection where applicable. A diagnostic LP over each final pool, with a separate ten-second time limit, measures the LP gain; every diagnostic LP is solved to optimality.

Table 5 compares the resulting feasible costs and paired integer and LP gains relative to EUC on the same instance. Each paired gain is the cost reduction divided by the corresponding EUC cost, while Win/Tie/Loss counts compare each procedure with EUC. Reported time, an arithmetic mean, includes generation and integer repair but excludes diagnostic LP time.

Dual-guided paths yield the lowest repaired costs. DG-CGM *path generation* reduces the cost of the *construction* plan by 4.9% on average, compared with 3.1% for EUC and 1.9% for transportation cost. Relative to EUC paths on the same instance, dual-guided paths reduce the repaired cost by 1.9% on average, with 51 wins, 84 ties, and 15 losses, whereas paths ranked by transportation cost, which ignores opening, handling, and interchange costs, increase it by 1.6%. The average cost reductions on winning instances also exceed the increases on losing instances, 6.4% versus 2.8%. The losses are consistent with differences in how the two criteria account for opening costs. JRC evaluates a path using LP dual prices, which reflect fractional block openings and can understate fixed opening costs, whereas EUC charges the full opening cost of an unopened block to the demand that would open it. The LP objective over the final dual-guided pool is also 3.2% lower on average than that over the EUC pool.

Table 5 Logical paths and integer repair on the small generated instances.

Procedure	Cost/ 10^6	Gain from z^0/%	Paired integer gain/%	Paired LP gain/%	Win/Tie/Loss	Time/s
DG-CGM	18.2	4.9	1.9	3.2	51/84/15	0.8
Transportation cost	18.8	1.9	−1.6	−1.9	4/113/33	0.1
EUC	18.6	3.1	0.0	0.0	–	<0.1

6.2.4. Robustness. We evaluate the robustness of DG-CGM across five random seeds and under parameter variations based on Table EC.1, retaining the 60-second time limit. Across the five runs, the mean gap ranges from 0.10% to 0.16%; each run matches 89 proven optima, and 143 of the 150 instances have identical costs across all runs. The largest coefficient of variation in cost for an instance is 2.3%. Table EC.2 in the electronic companion reports the 14 control variants. Each variant changes the mean gap by at most 0.08 percentage points, comparable to the variation across runs, and leaves costs unchanged on 141 to 150 instances. Among the tested parameters, only the maximum number of pricing batches per *path generation* invocation, I_{PG}, shows statistically significant differences from the default setting at the 5% level. Reducing I_{PG} from its default value of 10 to 5 decreases the mean gap from 0.15% to 0.09%, while increasing it to 20 raises the gap to 0.17%, indicating the robustness of DG-CGM.

6.3. Larger Generated Instances

The 12-demand instances are small enough for ARC MIP to prove optimality on many of them, whereas practical blocking problems involve many more demands. The four larger sets in Table 1 first increase the number of demands on 150-node networks and then increase the network size while keeping 48 demands. DG-CGM uses a 300-second time limit; we run the complete method with three different random seeds and each component variant with the default seed. Since proven optima are unavailable, we evaluate solution quality using the bound gap relative to the DW LP bound. We also run ARC LP and ARC MIP on all sets except N600K48, whose ARC formulation contains about 2.3×10^7 block–arc pairs.

Table 6 summarizes the results. Where the DW bound is available, the mean gap of DG-CGM to this bound is 1.6% on N150K24 and at most 0.6% on N150K48 and N300K48, compared with 24.2% to 78.8% for ARC MIP. ARC MIP returns higher-cost solutions on 28 of the 30 instances. The larger gap on N150K24 likely reflects the bound more than the quality of the feasible plans. On the 12-demand instances, the DW bound alone lies 3.1% below the reference cost (Table 2), and with fewer demands per block, the LP relaxation has greater flexibility to satisfy minimum block loads through fractional block openings. DG-CGM reaches these gaps within 300 seconds, whereas exact computations take considerably longer at this scale. Within two hours, the DW LP returns a bound on 8 of the 10 N300K48 instances and on none of the N600K48 instances, while ARC LP requires 5,985.4 seconds on N300K48 (geometric mean).

The three DG-CGM runs yield nearly identical costs, with a mean coefficient of variation of at most 0.06% in every set. The last three columns show that the contribution of *joint repair* increases with instance

size. Removing *joint repair* raises the mean cost by 0.02% on N150K24 and by 1.39% on N600K48, while removing *path generation* raises the mean cost by up to 0.27% on the three sets with at most 300 nodes. Without *path generation*, runs on these three sets terminate early, after 188 to 249 seconds on average, as on the RAS instances (Section 6.4.2). On N600K48, removing *path generation* lowers the mean cost by 0.05%, and the search then completes 7.9 instead of 5.6 joint MIP calls per run, consistent with *joint repair* making the best use of the 300 seconds at this scale. Removing *local repair* changes the mean cost by at most 0.08%, consistent with each *joint repair* call releasing all 24 or 48 demands, as on the 12-demand instances. The restriction of *joint repair* described in Section 6.2.2 confirms the contribution of *path generation* at this scale. Under this restriction, removing *path generation* raises the mean cost by 0.61%, 0.50%, 0.65%, and 0.86% on N150K24, N150K48, N300K48, and N600K48, respectively, with 27 wins, 11 ties, and two losses across the 40 instances.

Table 6 DG-CGM, DW LP, and ARC MIP on the larger generated instances.

	DW LP		Bound gap/%					Increase without/%		
Set	#Bounds	Time/s	z^0	DG-CGM mean	DG-CGM max.	ARC MIP	Improvement from z^0/%	Path gen.	Local	Joint
N150K24	10	3.4	4.7	1.6	5.9	24.2	3.2	0.27	0.00	0.02
N150K48	10	3.5	3.4	0.5	1.6	34.8	2.9	0.03	0.00	0.31
N300K48	8	86.0	2.2	0.6	0.9	78.8	2.2	0.07	0.04	0.61
N600K48	0	–	–	–	–	–	3.9	−0.05	0.08	1.39

Note. Bound gaps and DW LP times average the instances with a DW bound. Improvement from z^0 and the last three columns, the mean paired cost increase when a component is removed, use seed 1.

6.4. RAS Instances

6.4.1. *Construction* and Complete Search. Table 7 compares the cost z^0 after *construction* with the final cost z^{H} obtained by DG-CGM, whose search uses the remainder of the two-hour time limit. Across the six scenarios, DG-CGM reduces the *construction* cost by 7.2% on average, with reductions ranging from 0.7% on L3×0.5 to 26.4% on L3×2. The *construction* plan is identical across runs, and the three final costs differ by at most 0.6% in every scenario except L3×2. On L3×2, the final costs range from 6,554.2 to 7,412.1, so even the most expensive run reduces the *construction* cost by 22.0%.

The larger spread on L3×2 is consistent with congestion. Its plans use all classification tracks at 26 yards (Table 9), so the final plan depends on which demands the search first admits to these tracks. The smallest reduction, on L3×0.5, is consistent with the design of this scenario. Halving demand volumes leaves minimum block loads unchanged, and L3 demands are already 15.5% smaller on average than L1 demands. Consequently, many demands may be unable to reach a minimum block load even jointly; L3×0.5 keeps the lowest service of all scenarios, 95.9%, after the search (Table 9).

Table 7 *Construction* and complete DG-CGM on the RAS instances.

	Construction		DG-CGM			
Scenario	$z^0/10^6$	Time/s	$z^{\mathrm{H}}/10^6$	z^{H} range/10^6	Improvement/%	Total time/s
L1×0.5	1,631.4	631.7	1,568.9	1,565.9–1,574.9	3.8	7,209.3
L1×1	3,036.4	1,787.2	2,974.5	2,972.9–2,975.5	2.0	7,210.0
L1×2	6,145.4	2,306.5	5,882.4	5,882.1–5,883.1	4.3	7,206.0
L3×0.5	1,787.6	1,022.1	1,775.7	1,775.1–1,776.6	0.7	7,205.3
L3×1	3,327.3	1,768.0	3,123.9	3,121.5–3,126.6	6.1	7,204.9
L3×2	9,498.2	3,715.3	6,989.7	6,554.2–7,412.1	26.4	7,203.7
Mean	–	–	–	–	7.2	–

Note. z^{H}, *construction* time, and total time are means over three runs; the improvement uses the mean z^{H}.

6.4.2. Contributions of the Search Components. To examine the components at full scale, we run DG-CGM and each variant from Section 6.2.2 starting from the *construction* plan of each scenario, allowing 7,200 seconds of search after *construction*. Table 8 reports the final cost of DG-CGM and the relative cost increase of each variant. *Joint repair* again has the largest contribution. Without it, the cost increases by 0.4% to 6.6% on five scenarios and by 37.1% on L3×2, where most of the improvement over *construction* comes from serving more demand. Removing *path generation* increases the cost on every scenario, by 0.3% to 1.7%; without it, the search terminates early, after 341 to 757 seconds, indicating that the paths generated by this component sustain the search over the two-hour time limit. The restriction of *joint repair* described in Section 6.2.2 makes this contribution more pronounced. Under this restriction, the search without *path generation* terminates after 1 to 41 seconds and changes the *construction* cost by less than 0.001% on four of the six scenarios, whereas with *path generation* it runs for the full 7,200 seconds and reduces the *construction* cost by 3.2% on average. Removing *path generation* in this setting increases the cost by 0.7% to 4.3%, exceeding the increase without the restriction on every scenario. *Local repair* matters mainly on L3×2, where its removal increases the cost by 7.9%; elsewhere, the increase is at most 0.3%. This pattern is consistent with the initial demand selection in *destroy*, which prioritizes demands in decreasing order of unserved demand cost. The *construction* plan of L3×2 serves 85.7% of the requested volume, compared with 98.8% for L1×2 and 97.2% for L3×1 (Table 12), leaving many large unserved demands for *local repair* to recover.

6.4.3. Demand Volume and Operating Plans. Table 9 reports total cost, cost per requested unit $z/\sum_k q_k$, volume-weighted service, and the number of open blocks. Table 10 reports the cost components. For each scenario, we analyze the lowest-cost plan among the three runs. We measure congestion by the number of resources operating at capacity, i.e., physical links whose loads equal their capacities, yards whose handling volumes equal their capacities, and yards with all classification tracks in use.

On L1, increasing the demand multiplier from 0.5 to 2 raises the number of open blocks from 759 to 1,249 and reduces the cost per requested unit from 1,002.8 to 941.7, consistent with consolidation over

Table 8 Search components on the RAS instances.

			Cost increase without/%		
Scenario	$z^0/10^6$	$z^H/10^6$	*Path generation*	*Local repair*	*Joint repair*
L1×0.5	1,631.4	1,566.1	1.4	0.3	3.9
L1×1	3,036.4	2,970.0	0.3	0.2	2.1
L1×2	6,145.4	5,880.7	0.3	0.0	2.8
L3×0.5	1,787.6	1,776.1	0.4	0.2	0.4
L3×1	3,327.3	3,118.1	0.3	0.2	6.6
L3×2	9,498.2	6,429.1	1.7	7.9	37.1

larger volumes, while service remains at or above 99.3%. L3 opens more blocks than L1 at every multiplier and serves 95.9% to 98.2% of the requested volume, making unserved demand a larger cost component. At $\alpha = 2$, the cost per requested unit on L3 rises again to 1,046.1, with an unserved demand cost of 755.5 in 10^6 model cost units and 26 yards using all their classification tracks, compared with 28.6 and 19, respectively, on L1×2.

Table 9 Cost, service, and resource use under the six RAS demand scenarios.

					Resources at capacity		
Scenario	Total cost/10^6	Cost/ requested unit	Service/%	Open blocks	Links	Yards (handling)	Yards (tracks)
L1×0.5	1,565.9	1,002.8	99.3	759	0	0	2
L1×1	2,972.9	951.9	99.6	1,001	0	0	7
L1×2	5,882.1	941.7	99.6	1,249	3	0	19
L3×0.5	1,775.1	1,133.3	95.9	789	0	0	7
L3×1	3,121.5	996.4	98.2	1,109	0	0	12
L3×2	6,554.2	1,046.1	96.6	1,293	0	2	26

Table 10 Cost components of the RAS plans.

Scenario	Opening	Transportation	Handling	Interchange	Unserved	Total/10^6
L1×0.5	1.9	1,422.0	40.6	52.7	48.7	1,565.9
L1×1	2.5	2,795.0	46.5	101.9	27.0	2,972.9
L1×2	3.1	5,572.4	75.7	202.3	28.6	5,882.1
L3×0.5	2.0	1,474.5	48.6	54.2	195.9	1,775.1
L3×1	2.8	2,808.0	71.0	104.2	135.5	3,121.5
L3×2	3.2	5,474.6	119.7	201.1	755.5	6,554.2

6.4.4. Resource Limits. To relate the operating plans to individual resource limits, we tighten one limit at a time on L1×1 and L3×1 and rerun DG-CGM. We multiply yard handling capacities by 0.75, reduce the number of classification tracks T_i at each yard to $\lceil 0.75T_i \rceil$, multiply physical link capacities by 0.75, or increase minimum block loads by a factor of 1.5 without exceeding the block upper bounds. We also vary the *joint repair* cap from 250 to 100 and 500. Table 11 compares each variant with the unchanged scenario under the same seed.

The minimum block load has the largest effect. Increasing it by 50% raises cost by 3.1% on L1×1 and by 10.4% on L3×1, reduces the number of open blocks by 16% to 20%, and increases unserved demand cost by a factor of about two on L1×1 and three on L3×1. Reducing the number of classification tracks raises cost by 0.7% on L1×1 but by 7.9% on L3×1, where more yards already use all their classification tracks (Table 9). Lower handling and link capacities change the cost of L1×1 by at most 0.5%; on L3×1, reducing link capacities raises the mean cost by 2.4%, mainly due to one run. These two limits rarely bind, as no physical link reaches capacity and no yard reaches its handling capacity in the plans for L1×1 and L3×1 reported in Table 9. The 0.1% cost decrease on L1×1 under tighter link capacities is consistent with the load limits in Section EC.2.5, which tighten as link capacities decrease and can strengthen the restricted MIPs, and with the different physical candidates subsequently generated by *construction*.

The default *joint repair* cap of 250 balances the size of each joint MIP against the time available to solve it. A cap of 100 increases cost by 0.7% to 0.9% in both scenarios, whereas a cap of 500 reduces the cost of L1×1 by 0.3% but leaves L3×1 close to its *construction* cost (3,324.4 versus 3,327.3 in 10^6 model cost units). The latter result indicates that MIPs with up to 500 released demands yield little improvement within the 30 seconds allotted to each *joint repair* invocation (Table EC.1). L1×1, with fewer demands and fewer yards at track capacity, still benefits from the larger cap.

Table 11 Resource limits and the *joint repair* cap on the RAS instances.

Scenario	Variant	$z/10^6$	Change/%	Change by seed/%	Service/%	Open blocks	Unserved/ 10^6
L1×1	Yard handling capacity ×0.75	2,988.0	0.5	0.3 to 0.5	99.7	1,039	43.5
	Classification tracks $\lceil 0.75T_i \rceil$	2,995.2	0.7	0.6 to 0.8	99.5	971	36.0
	Link capacity ×0.75	2,970.5	−0.1	−0.2 to −0.1	99.6	1,002	24.1
	Minimum block load ×1.5	3,067.0	3.1	2.8 to 3.5	99.4	841	64.1
	Joint repair cap 100	2,994.1	0.7	0.5 to 0.8	99.6	1,019	34.3
	Joint repair cap 500	2,966.5	−0.3	−0.4 to −0.2	99.6	1,021	21.5
L3×1	Yard handling capacity ×0.75	3,130.8	0.2	−0.3 to 0.7	98.1	1,111	150.7
	Classification tracks $\lceil 0.75T_i \rceil$	3,371.6	7.9	2.8 to 17.3	96.0	992	444.6
	Link capacity ×0.75	3,199.0	2.4	0.0 to 6.5	97.8	1,103	219.2
	Minimum block load ×1.5	3,448.6	10.4	8.2 to 14.7	96.2	885	399.3
	Joint repair cap 100	3,151.1	0.9	0.8 to 0.9	98.1	1,103	160.5
	Joint repair cap 500	3,324.4	6.4	6.3 to 6.5	97.2	1,105	362.8

Note. Means over three runs. Change compares the mean cost with that of the unchanged scenario; change by seed reports the smallest and largest paired cost changes across the three runs.

6.4.5. Service Recovery on L3×2. The largest reduction, on L3×2, comes mainly from serving demand left unserved by the *construction* plan. Table 12 compares the *construction* plans of L1×2, L3×1, and L3×2 in terms of unserved demand cost, service, and the number of resources operating at capacity, counted as in Table 9. In the L3×2 *construction* plan, ten yards operate at full handling capacity, and the unserved demand cost of 4,433.4 accounts for 46.7% of the total cost, compared with 10.9% for L3×1 and 6.0% for L1×2. DG-CGM serves much of this unserved demand. The resulting L3×2 plan in Table 10

reduces the unserved demand cost from 4,433.4 to 755.5 and has only two yards operating at handling capacity.

Table 12 Service and resource use in the *construction* plans.

				Resources at capacity		
Scenario	Unserved demand cost/10^6	Share of total cost/%	Service/%	Links	Yards (handling)	Yards (tracks)
L1×2	366.6	6.0	98.8	3	2	17
L3×1	363.8	10.9	97.2	0	0	15
L3×2	4,433.4	46.7	85.7	3	10	26

7. Conclusions

We develop DG-CGM for the IRBP, which jointly determines block design, blocking sequences, and physical block routes under minimum-load and shared-resource constraints. We formulate the problem as ARC and derive its exact DW reformulation with a stronger LP relaxation. The proposed JRC enables exact pricing of logical and physical paths without path-specific dual variables. Building on these results, DG-CGM combines dual-guided *path generation* with restricted MIP-based *local and joint repair* to progressively improve feasible blocking plans.

Computational experiments demonstrate the effectiveness of the proposed approach. On 150 generated instances, the DW formulation reduces the mean lower-bound gap from 67.5% for ARC to 3.1%. DG-CGM matches 89 of the 91 proven optima within 60 seconds and reduces the feasible costs obtained by ARC MIP by 3.4% on average, while requiring only 5.2% of its geometric mean runtime. On larger generated instances, the mean gap of DG-CGM to the DW lower bound is at most 1.6% where this bound is available. On six full-scale RAS scenarios with 47,193 physical nodes, DG-CGM reduces the *construction* cost by 7.2% on average and serves 95.9%–99.6% of the requested freight volume within two hours.

These findings highlight the interplay between *path generation* and coordinated optimization in the IRBP. Newly generated logical paths may require multiple demands to jointly satisfy minimum block loads, while changes in demand assignments can affect block openings, physical paths, and shared resource capacities. Conversely, restricted MIP-based repair cannot explore logical paths absent from its candidate pool. Combining *path generation* with *local repair* and *joint repair* allows the search to expand its routing alternatives and coordinate their use in feasible blocking plans.

We conclude with three directions for future research. First, learning-based methods could be incorporated into DG-CGM to improve search efficiency and adapt the algorithm to different network characteristics and demand patterns. Second, the DW formulation and exact pricing algorithms could be extended to develop a branch-and-price algorithm for solving larger IRBP instances to proven optimality. High-quality feasible solutions obtained by DG-CGM could provide initial upper bounds to accelerate the exact search. Finally, the DG-CGM framework could be extended to address other railway planning problems involving time-dependent freight demands and train schedules.

References

Ahuja RK, Jha KC, Liu J (2007) Solving Real-Life Railroad Blocking Problems. *Interfaces* 37(5):404–419.

Arsalani P, Reisi-Nafchi M, Dardashti V, Moslehi G (2023) Two new mixed-integer programming models for the integrated train formation and shipment path optimization problem. *Networks* 81(3):359–377.

Barnhart C, Jin H, Vance PH (2000) Railroad Blocking: A Network Design Application. *Operations Research* 48(4):603–614.

Bodin LD, Golden BL, Schuster AD, Romig W (1980) A model for the blocking of trains. *Transportation Research Part B: Methodological* 14(1-2):115–120.

Buson E, Roberti R, Toth P (2014) A Reduced-Cost Iterated Local Search Heuristic for the Fixed-Charge Transportation Problem. *Operations Research* 62(5):1095–1106.

Crainic T, Ferland JA, Rousseau JM (1984) A Tactical Planning Model for Rail Freight Transportation. *Transportation Science* 18(2):165–184.

CSX Corporation (2024) CSX Investor Day 2024. Technical report, CSX Corporation.

Dantzig GB, Wolfe P (1960) Decomposition Principle for Linear Programs. *Operations Research* 8(1):101–111.

Fang B, Li P, Wei Y, Xia Y (2023) A two-stage, single-tier, restricted blocking plan and train design approach based on candidate blocks. *Journal of Rail Transport Planning & Management* 28:100413.

Gurobi Optimization, LLC (2026) Gurobi optimizer reference manual.

Hewitt M, Nemhauser G, Savelsbergh MWP (2013) Branch-and-Price Guided Search for Integer Programs with an Application to the Multicommodity Fixed-Charge Network Flow Problem. *INFORMS Journal on Computing* 25(2):302–316.

INFORMS Railway Applications Section (2026) The 2026 RAS Problem Solving Competition.

Jaccard P (1912) The Distribution of the Flora in the Alpine Zone. *New Phytologist* 11(2):37–50.

Keaton MH (1989) Designing optimal railroad operating plans: Lagrangian relaxation and heuristic approaches. *Transportation Research Part B: Methodological* 23(6):415–431.

Lübbecke ME, Desrosiers J (2005) Selected Topics in Column Generation. *Operations Research* 53(6):1007–1023.

Lübbecke ME, Witt JT (2018) The strength of Dantzig–Wolfe reformulations for the stable set and related problems. *Discrete Optimization* 30:168–187.

MacQueen J (1967) Some methods for classification and analysis of multivariate observations. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, 281–297 (Berkeley, CA: University of California Press).

Maekawa S, Sasaki Y, Fletcher G, Onizuka M (2023) GenCAT: Generating attributed graphs with controlled relationships between classes, attributes, and topology. *Information Systems* 115:102195.

Newton HN, Barnhart C, Vance PH (1998) Constructing Railroad Blocking Plans to Minimize Handling Costs. *Transportation Science* 32(4):330–345.

Parisi F, Squartini T, Garlaschelli D (2020) A faster horse on a safer trail: Generalized inference for the efficient reconstruction of weighted networks. *New Journal of Physics* 22(5):053053.

Union Pacific (2025) Senior Leaders Reinforce Safety is Job No. 1 During North Platte Field Visit.

Vallarano N, Bruno M, Marchese E, Trapani G, Saracco F, Cimini G, Zanon M, Squartini T (2021) Fast and scalable likelihood maximization for Exponential Random Graph Models with local constraints. *Scientific Reports* 11(1):15227.

Visit North Platte (2020) North Platte Travel Guide 2020.

Voll R, Clausen U (2013) Branch-and-Price for a European variant of the Railroad Blocking Problem. *Electronic Notes in Discrete Mathematics* 41:45–52.

Wilcoxon F (1945) Individual Comparisons by Ranking Methods. *Biometrics Bulletin* 1(6):80–83.

Xin X, Wu X, Wang J (2025) Rail freight transportation service network design with mixed blocking policies: Formulation and solution methods. *Journal of Rail Transport Planning & Management* 33:100495.

Yaghini M, Ahadi HR, Barati E, Saghian Z (2013) Tabu Search Algorithm for the Railroad Blocking Problem. *Journal of Transportation Engineering* 139(2):216–222.

Yaghini M, Mirghavami M, Zare Andaryan A (2021) A mathematical formulation and an LP-based neighborhood search matheuristic solution method for the integrated train blocking and shipment path problem. *Networks* 78(4):523–542.

Yaghini M, Seyedabadi M, Khoshraftar MM (2012) A population-based algorithm for the railroad blocking problem. *Journal of Industrial Engineering International* 8(1):8.

You Z, Yang Y (2026) RouteOpt: An Open-Source Modular Exact Solver for Vehicle Routing Problems. *INFORMS Journal on Computing*.

Zhu E, Crainic TG, Gendreau M (2014) Scheduled Service Network Design for Freight Rail Transportation. *Operations Research* 62(2):383–400.

Electronic Companion

EC.1. Proofs

EC.1.1. Proof of Theorem 1

Proof. We partition the constraints of ARC into three groups. The *demand subsystem* of demand k consists of Eqs. (1b)–(1g) for k, the bounds on ℓ^{D}_{ki} in Eq. (1u), and the binary domains of x^{D}_{k} and x^{D}_{kb}. Its variables other than the order variables form the vector $u^{\mathrm{D}}_{k} = (x^{\mathrm{D}}_{k}, (x^{\mathrm{D}}_{kb})_{b\in\mathcal{B}_k}, v_k, (g_{kb})_{b\in\mathcal{B}_k})$. The *block subsystem* of block b consists of Eqs. (1i)–(1k) and (1m)–(1p) for b, the bounds on ℓ^{B}_{bn}, and the binary domains of y_b and x^{B}_{ba}, with $u^{\mathrm{B}}_{b} = (y_b, (x^{\mathrm{B}}_{ba})_{a\in\mathcal{A}}, f_b, (w_{ba})_{a\in\mathcal{A}})$. The remaining Eqs. (1h), (1l), and (1q)–(1t) are the *coupling constraints*. For $p \in \mathcal{P}_k$ and $r \in \mathcal{R}_b$, we write $\iota_p = (\iota_{bp})_{b\in\mathcal{B}_k}$ and $\delta_r = (\delta_{ar})_{a\in\mathcal{A}}$.

Step 1. We show that the set $\mathcal{X}^{\mathrm{D}}_{k}$ of vectors u^{D}_{k} for which some order vector $(\ell^{\mathrm{D}}_{ki})_{i\in\mathcal{Y}}$ satisfies the demand subsystem of k is

$$\mathcal{X}^{\mathrm{D}}_{k} = \{0\} \cup \bigcup_{p\in\mathcal{P}_k} \big\{(1, \iota_p, t, t\iota_p) : 0 \le t \le q_k\big\}. \tag{EC.1}$$

Let $u^{\mathrm{D}}_{k} \in \mathcal{X}^{\mathrm{D}}_{k}$. If $x^{\mathrm{D}}_{k} = 0$, then, since every $b \in \mathcal{B}_k$ belongs to $\mathcal{B}^{+}_{k}(i_b)$, Eq. (1d) at $i = i_b$ gives $x^{\mathrm{D}}_{kb} = 0$, and Eqs. (1f) and (1g) give $g_{kb} = 0$ and $v_k = 0$. Hence $u^{\mathrm{D}}_{k} = 0$.

Now let $x^{\mathrm{D}}_{k} = 1$, and call the blocks $b \in \mathcal{B}_k$ with $x^{\mathrm{D}}_{kb} = 1$ *selected*. By Eq. (1d), at most one selected block leaves each yard and at most one enters it. Since $o_k \neq d_k$, Eq. (1b) states that the number of selected blocks leaving a yard minus the number entering it is one at o_k, minus one at d_k, and zero elsewhere. Together with these upper bounds, it implies that exactly one selected block leaves o_k and none enters it, that exactly one enters d_k and none leaves it, and that every other yard is left by a selected block if and only if it is entered by one. Starting at o_k, we repeatedly follow the selected block that leaves the current yard. Every yard reached in this way other than d_k is entered by a selected block and is therefore also left by one, so the walk can stop only at d_k. The walk never returns to a yard. A return to o_k would require a selected block entering o_k. At the first return to another yard, that yard would be entered by two distinct selected blocks, because the blocks used for its two entries leave two different yards, as no yard is repeated before this return. Because $\mathcal{Y}$ is finite, the walk reaches d_k and traces an elementary logical path $p \in \mathcal{P}_k$ whose blocks are selected.

We next show that no other block is selected. Each yard of p is left by at most one selected block and entered by at most one, and these blocks lie on p, because no selected block enters o_k or leaves d_k. Hence every selected block off p leaves and enters yards off p, and such a yard is left by a selected block off p if and only if it is entered by one. If some selected block lies off p, following selected blocks off p from the tail of any of them therefore never stops, so some yard is eventually reached a second time. The blocks traversed between the two visits of the first such yard form a directed cycle C of selected blocks in which every yard is the head of exactly one block and the tail of exactly one block. Summing Eq. (1e) over $b \in C$, for which $x^{\mathrm{D}}_{kb} = 1$, gives $\sum_{b\in C}(\ell^{\mathrm{D}}_{kj_b} - \ell^{\mathrm{D}}_{ki_b}) \ge |C| \ge 1$, whereas the left side equals zero. Hence the selected blocks are exactly the blocks of p, that is, $(x^{\mathrm{D}}_{kb})_{b\in\mathcal{B}_k} = \iota_p$.

For the volumes, Eq. (1f) gives $g_{kb} = 0$ for $b \notin p$. Let $b_1, \dots, b_m$ be the blocks of p in order, so that $i_{b_1} = o_k$, $j_{b_s} = i_{b_{s+1}}$ for $s < m$, and $j_{b_m} = d_k$. Eq. (1c) at o_k gives $g_{kb_1} = v_k$, because b_1 is the only block of p incident to o_k, and at j_{b_s} with $s < m$ it gives $g_{kb_{s+1}} = g_{kb_s}$. Hence $g_{kb} = v_k \iota_{bp}$ for every $b \in \mathcal{B}_k$, and Eq. (1g) gives $0 \le v_k \le q_k$. Thus $u^{\mathrm{D}}_k = (1, \iota_p, v_k, v_k \iota_p)$ belongs to the right side of Eq. (EC.1).

Conversely, the zero vector satisfies the demand subsystem with $\ell^{\mathrm{D}}_{ki} = 0$ for all i, since Eq. (1e) then reads $0 \ge 1 - |\mathcal{Y}|$. For $p \in \mathcal{P}_k$ with blocks $b_1, \dots, b_m$ and $t \in [0, q_k]$, set $\ell^{\mathrm{D}}_{ko_k} = 0$, $\ell^{\mathrm{D}}_{kj_{b_s}} = s$ for $s = 1, \dots, m$, and $\ell^{\mathrm{D}}_{ki} = 0$ at the yards off p. These positions lie in $[0, |\mathcal{Y}| - 1]$ because p visits $m + 1 \le |\mathcal{Y}|$ distinct yards. Eq. (1e) holds with equality for the blocks of p, and for every other block it reads $\ell^{\mathrm{D}}_{kj_b} - \ell^{\mathrm{D}}_{ki_b} \ge 1 - |\mathcal{Y}|$, which holds because both positions lie in $[0, |\mathcal{Y}| - 1]$. Eqs. (1b) and (1d) hold because p is an elementary path from o_k to d_k, and Eqs. (1c), (1f), and (1g) hold for $(1, \iota_p, t, t\iota_p)$ because $0 \le t \le q_k$. The two inclusions prove Eq. (EC.1).

Step 2. Similarly, the set $\mathcal{X}^{\mathrm{B}}_b$ of vectors u^{B}_b for which some order vector $(\ell^{\mathrm{B}}_{bn})_{n \in \mathcal{N}}$ satisfies the block subsystem of b is

$$\mathcal{X}^{\mathrm{B}}_b = \{0\} \cup \bigcup_{r \in \mathcal{R}_b} \{(1, \delta_r, t, t\delta_r) : \underline{f}_b \le t \le \overline{f}_b\}. \tag{EC.2}$$

If $y_b = 0$, Eq. (1j) at $n = i_a$ gives $x^{\mathrm{B}}_{ba} = 0$ for every arc a, Eq. (1p) gives $f_b = 0$, and Eq. (1n) gives $w_{ba} = 0$. If $y_b = 1$, Eq. (1p) gives $\underline{f}_b \le f_b \le \overline{f}_b$, so $\overline{f}_b \ge \underline{f}_b$. The argument of Step 1 applies to the arcs a with $x^{\mathrm{B}}_{ba} = 1$, with nodes in place of yards, Eqs. (1i), (1j), and (1m) in place of Eqs. (1b), (1d), and (1e), and $i_b \ne j_b$ in place of $o_k \ne d_k$. Hence the selected arcs form an elementary physical path r from i_b to j_b, and $(x^{\mathrm{B}}_{ba})_{a \in \mathcal{A}} = \delta_r$. Eq. (1k) gives $\sum_{a \in \mathcal{A}} D_a \delta_{ar} \le \overline{D}_b$, so $r \in \mathcal{R}_b$ because $\overline{f}_b \ge \underline{f}_b$. For $a \notin r$, Eq. (1n) gives $0 \le w_{ba} \le 0$. For $a \in r$, it gives $w_{ba} \le f_b$, while Eq. (1o) gives $w_{ba} \ge f_b - \overline{f}_b(1 - 1) = f_b$. Thus $u^{\mathrm{B}}_b = (1, \delta_r, f_b, f_b \delta_r)$. Conversely, the zero vector satisfies the block subsystem with zero positions, since Eq. (1m) then reads $0 \ge 1 - |\mathcal{N}|$ and Eq. (1o) reads $0 \ge 0$. For $r \in \mathcal{R}_b$ and $t \in [\underline{f}_b, \overline{f}_b]$, positions $0, 1, \dots$ along r and zero elsewhere lie in $[0, |\mathcal{N}| - 1]$ and satisfy Eq. (1m) as in Step 1, Eqs. (1i) and (1j) hold because r is an elementary path from i_b to j_b, Eq. (1k) holds because $r \in \mathcal{R}_b$, and Eqs. (1n)–(1p) hold for $(1, \delta_r, t, t\delta_r)$; in particular, Eq. (1o) reads $t \ge t$ for $a \in r$ and $0 \ge t - \overline{f}_b$ for $a \notin r$. When $\overline{f}_b < \underline{f}_b$, Eq. (1p) excludes $y_b = 1$, in agreement with $\mathcal{R}_b = \emptyset$.

Step 3. We relate the path variables of DW to these sets. Let $\xi = (x^{\mathrm{D}}, x^{\mathrm{B}}, y, v, f, g, w, v^{\mathrm{H}})$ collect the ARC variables other than the order variables. The linear map Φ from the DW variables (x, v) to ξ is

$$\begin{aligned} x^{\mathrm{D}}_k &= \sum_{p\in\mathcal{P}_k} x^{\mathrm{D}}_{kp}, & y_b &= \sum_{r\in\mathcal{R}_b} x^{\mathrm{B}}_{br}, \\ x^{\mathrm{D}}_{kb} &= \sum_{p\in\mathcal{P}_k} \iota_{bp} x^{\mathrm{D}}_{kp}, & x^{\mathrm{B}}_{ba} &= \sum_{r\in\mathcal{R}_b} \delta_{ar} x^{\mathrm{B}}_{br}, \\ v_k &= \sum_{p\in\mathcal{P}_k} v^{\mathrm{D}}_{kp}, & f_b &= \sum_{r\in\mathcal{R}_b} v^{\mathrm{B}}_{br}, \\ g_{kb} &= \sum_{p\in\mathcal{P}_k} \iota_{bp} v^{\mathrm{D}}_{kp}, & w_{ba} &= \sum_{r\in\mathcal{R}_b} \delta_{ar} v^{\mathrm{B}}_{br}, \\ v^{\mathrm{H}}_i &= \sum_{k\in\mathcal{K}} \sum_{p\in\mathcal{P}_k} \eta_{ip} v^{\mathrm{D}}_{kp}. \end{aligned} \tag{EC.3}$$

Let $\Phi^{\mathrm{D}}_k(x,v) = \sum_{p\in\mathcal{P}_k}(x^{\mathrm{D}}_{kp}, x^{\mathrm{D}}_{kp}\iota_p, v^{\mathrm{D}}_{kp}, v^{\mathrm{D}}_{kp}\iota_p)$ and $\Phi^{\mathrm{B}}_b(x,v) = \sum_{r\in\mathcal{R}_b}(x^{\mathrm{B}}_{br}, x^{\mathrm{B}}_{br}\delta_r, v^{\mathrm{B}}_{br}, v^{\mathrm{B}}_{br}\delta_r)$ denote the components of $\Phi(x,v)$ that form u^{D}_k and u^{B}_b. The *local constraints* of demand k in the DW LP relaxation are Eqs. (2f) and (2h) for k together with $x^{\mathrm{D}}_{kp} \geq 0$, that is, $x^{\mathrm{D}}_{kp} \geq 0$, $0 \leq v^{\mathrm{D}}_{kp} \leq q_k x^{\mathrm{D}}_{kp}$, and $\sum_p x^{\mathrm{D}}_{kp} \leq 1$. Those of block b, from Eqs. (2g) and (2i), are $x^{\mathrm{B}}_{br} \geq 0$, $\underline{f}_b x^{\mathrm{B}}_{br} \leq v^{\mathrm{B}}_{br} \leq \overline{f}_b x^{\mathrm{B}}_{br}$, and $\sum_r x^{\mathrm{B}}_{br} \leq 1$. We show two properties.

(a) If the local constraints of k hold and every x^{D}_{kp} is binary, then $\Phi^{\mathrm{D}}_k(x,v) \in \mathcal{X}^{\mathrm{D}}_k$, and every point of $\mathcal{X}^{\mathrm{D}}_k$ arises in this way. The same holds for blocks. The convexity constraint allows at most one path p with $x^{\mathrm{D}}_{kp} = 1$, and $v^{\mathrm{D}}_{kp'} \leq q_k x^{\mathrm{D}}_{kp'} = 0$ for every other path p'. Hence $\Phi^{\mathrm{D}}_k(x,v)$ is either zero or $(1, \iota_p, v^{\mathrm{D}}_{kp}, v^{\mathrm{D}}_{kp}\iota_p)$ with $0 \leq v^{\mathrm{D}}_{kp} \leq q_k$, which belongs to $\mathcal{X}^{\mathrm{D}}_k$ by Eq. (EC.1). Conversely, zero arises from zero path variables, and $(1, \iota_p, t, t\iota_p)$ arises from $x^{\mathrm{D}}_{kp} = 1$, $v^{\mathrm{D}}_{kp} = t$, and zero for all other paths. For a block, the same argument uses $\underline{f}_b \leq v^{\mathrm{B}}_{br} \leq \overline{f}_b$ when $x^{\mathrm{B}}_{br} = 1$ and Eq. (EC.2); when $\mathcal{R}_b = \emptyset$, both $\Phi^{\mathrm{B}}_b(x,v)$ and the only point of $\mathcal{X}^{\mathrm{B}}_b$ are zero.

(b) If the local constraints of k hold, then $\Phi^{\mathrm{D}}_k(x,v) \in \operatorname{conv} \mathcal{X}^{\mathrm{D}}_k$, and every point of $\operatorname{conv} \mathcal{X}^{\mathrm{D}}_k$ arises in this way. The same holds for blocks. For the first part, under the local constraints,

$$\Phi^{\mathrm{D}}_k(x,v) = \Big(1 - \sum_{p\in\mathcal{P}_k} x^{\mathrm{D}}_{kp}\Big) \cdot 0 + \sum_{p\in\mathcal{P}_k} \Big[\Big(x^{\mathrm{D}}_{kp} - \frac{v^{\mathrm{D}}_{kp}}{q_k}\Big)(1,\iota_p,0,0) + \frac{v^{\mathrm{D}}_{kp}}{q_k}(1,\iota_p,q_k,q_k\iota_p)\Big],$$

where the weights are nonnegative by the local constraints and sum to one, and the points belong to $\mathcal{X}^{\mathrm{D}}_k$. For a block with $\underline{f}_b < \overline{f}_b$, the corresponding representation places weight $(\overline{f}_b x^{\mathrm{B}}_{br} - v^{\mathrm{B}}_{br})/(\overline{f}_b - \underline{f}_b)$ on $(1, \delta_r, \underline{f}_b, \underline{f}_b\delta_r)$, weight $(v^{\mathrm{B}}_{br} - \underline{f}_b x^{\mathrm{B}}_{br})/(\overline{f}_b - \underline{f}_b)$ on $(1, \delta_r, \overline{f}_b, \overline{f}_b\delta_r)$, and weight $1 - \sum_r x^{\mathrm{B}}_{br}$ on zero. These weights are nonnegative, and those of path r sum to x^{B}_{br} and reproduce the volume v^{B}_{br}. When $\underline{f}_b = \overline{f}_b$, the local constraints give $v^{\mathrm{B}}_{br} = \underline{f}_b x^{\mathrm{B}}_{br}$, and the weight x^{B}_{br} on $(1, \delta_r, \underline{f}_b, \underline{f}_b\delta_r)$ suffices. When $\overline{f}_b < \underline{f}_b$, $\mathcal{R}_b = \emptyset$ and $\Phi^{\mathrm{B}}_b(x,v) = 0 \in \mathcal{X}^{\mathrm{B}}_b$. For the second part, the path variables that satisfy the local constraints form a convex set and Φ^{D}_k is linear, so their images form a convex set, which contains $\mathcal{X}^{\mathrm{D}}_k$ by (a) and hence contains $\operatorname{conv} \mathcal{X}^{\mathrm{D}}_k$; the same argument applies to blocks.

Step 4. We show that, for $\xi = \Phi(x,v)$, the coupling constraints and the objective of ARC coincide with the remaining constraints and the objective of DW. Eq. (1l) becomes $\sum_r v^{\mathrm{B}}_{br} = \sum_{k\in\mathcal{K}_b} \sum_p \iota_{bp} v^{\mathrm{D}}_{kp}$, which is

Eq. (2b). Eq. (1h) becomes $\sum_p \iota_{bp} x^{\mathrm{D}}_{kp} \le \sum_r x^{\mathrm{B}}_{br}$, which is Eq. (2j). Eq. (1r) becomes Eq. (2e), and, since $\sum_{a\in\mathcal{A}_e} w_{ba} = \sum_r (\sum_{a\in\mathcal{A}_e} \delta_{ar}) v^{\mathrm{B}}_{br}$, Eq. (1s) becomes Eq. (2c). For handling, an elementary path $p \in \mathcal{P}_k$ leaves each yard at most once, so $\sum_{b\in\mathcal{B}^+_k(i)} \iota_{bp}$ equals one if p leaves yard i and zero otherwise. Since $o_k \neq d_k$, every $p \in \mathcal{P}_k$ leaves o_k, and therefore $\sum_{b\in\mathcal{B}^+_k(i)} \iota_{bp} - \mathbf{1}\{i = o_k\} = \eta_{ip}$ for every yard i. Substituting $g_{kb} = \sum_p \iota_{bp} v^{\mathrm{D}}_{kp}$ and $v_k = \sum_p v^{\mathrm{D}}_{kp}$ into the right side of Eq. (1q) thus gives $\sum_k \sum_p \eta_{ip} v^{\mathrm{D}}_{kp}$, the value of v^{H}_i in Eq. (EC.3). Hence Eq. (1q) holds, and Eq. (1t) becomes Eq. (2d). Finally, substituting Eq. (EC.3) into Objective (1a) gives

$$\begin{aligned}
&\sum_b c^{\mathrm{O}}_b \sum_r x^{\mathrm{B}}_{br} + \sum_{b,r} \Big(\sum_a c^{\mathrm{T}}_{ba} \delta_{ar}\Big) v^{\mathrm{B}}_{br} + \sum_{k,p} \Big(\sum_i c^{\mathrm{H}}_i \eta_{ip}\Big) v^{\mathrm{D}}_{kp} + \sum_b c^{\mathrm{I}}_b \sum_r v^{\mathrm{B}}_{br} + \sum_k c^{\mathrm{U}}_k \Big(q_k - \sum_p v^{\mathrm{D}}_{kp}\Big) \\
&\quad = \sum_k q_k c^{\mathrm{U}}_k + \sum_{k,p} c^{\mathrm{D}}_{kp} v^{\mathrm{D}}_{kp} + \sum_{b,r} \big(c^{\mathrm{O}}_b x^{\mathrm{B}}_{br} + c^{\mathrm{B}}_{br} v^{\mathrm{B}}_{br}\big)
\end{aligned}$$

by the definitions of c^{D}_{kp} and c^{B}_{br} in Section 4.1, and the right side is Objective (2a).

Step 5. We prove $z^*_{\mathrm{arc}} = z^*_{\mathrm{dw}}$. Let (x, v) be feasible for DW. By Step 3(a), $\Phi^{\mathrm{D}}_k(x, v) \in \mathcal{X}^{\mathrm{D}}_k$ and $\Phi^{\mathrm{B}}_b(x, v) \in \mathcal{X}^{\mathrm{B}}_b$, so order vectors exist that satisfy every demand and block subsystem. By Step 4, $\Phi(x, v)$ satisfies the coupling constraints and has the DW objective value. Hence $\Phi(x, v)$, together with these order vectors, is feasible for ARC with the same cost, and $z^*_{\mathrm{arc}} \le z^*_{\mathrm{dw}}$. Conversely, let ξ and its order vectors be feasible for ARC. By Eqs. (EC.1) and (EC.2), each u^{D}_k is zero or equals $(1, \iota_p, t, t\iota_p)$ for a path $p \in \mathcal{P}_k$, which is unique because it consists of the blocks with $x^{\mathrm{D}}_{kb} = 1$, and each u^{B}_b is zero or equals $(1, \delta_r, t, t\delta_r)$ for a unique $r \in \mathcal{R}_b$. Setting the selection and volume variables of these paths to one and t, and all other path variables to zero, gives binary path variables that satisfy the local constraints and whose image under Φ is ξ, where v^{H} agrees by Step 4. By Step 4, these path variables also satisfy the remaining DW constraints and have the cost of ξ, so $z^*_{\mathrm{dw}} \le z^*_{\mathrm{arc}}$. Therefore $z^*_{\mathrm{arc}} = z^*_{\mathrm{dw}}$.

Steps 3 and 4 also identify DW as a Dantzig–Wolfe reformulation of ARC. Under the linear map Φ, the local constraints of the DW LP relaxation describe exactly $\operatorname{conv} \mathcal{X}^{\mathrm{D}}_k$ and $\operatorname{conv} \mathcal{X}^{\mathrm{B}}_b$ by Step 3(b), and the coupling constraints and objective of ARC become the remaining constraints and objective of DW by Step 4. The DW LP relaxation is therefore the master problem obtained by convexifying every demand and block subsystem while retaining the coupling constraints, and requiring binary path selections recovers exactly $\mathcal{X}^{\mathrm{D}}_k$ and $\mathcal{X}^{\mathrm{B}}_b$ by Step 3(a), which makes the reformulation exact. Formally, Φ maps the feasible set of DW onto the set of vectors ξ that, together with some order vectors, are feasible for ARC, and it preserves the objective value.

Step 6. We prove $z^{\mathrm{lp}}_{\mathrm{arc}} \le z^{\mathrm{lp}}_{\mathrm{dw}}$. Let (x, v) be feasible for the DW LP relaxation. By Step 3(b), $\Phi^{\mathrm{D}}_k(x, v)$ is a convex combination of points of $\mathcal{X}^{\mathrm{D}}_k$, and by Step 1, each of these points satisfies the demand subsystem together with some order vector. Once the binary domains are relaxed to $[0, 1]$, every constraint of the demand subsystem is linear, so the same convex combination of these points and their order vectors satisfies the LP relaxation of the demand subsystem, and its components other than the order vector equal $\Phi^{\mathrm{D}}_k(x, v)$. The same argument applies to every block. By Step 4, $\Phi(x, v)$ also satisfies the coupling constraints and has

the DW objective value. Hence every feasible solution of the DW LP relaxation maps to a feasible solution of the ARC LP relaxation with the same objective value, and $z^{\mathrm{lp}}_{\mathrm{arc}} \le z^{\mathrm{lp}}_{\mathrm{dw}}$.

Step 7. We show that the inequality can be strict. Consider the instance with yards $\mathcal{Y} = \{o, d\}$, physical nodes $\mathcal{N} = \{o, n_1, n_2, d\}$, and four one-way links of capacity 1, each consisting of a single arc. Arcs (o, n_1) and (n_1, d) have length 1 and unit transportation cost 5, and arcs (o, n_2) and (n_2, d) have length 2 and unit transportation cost 1. There is one block b from o to d with $\overline{D}_b = 3$, $\underline{f}_b = 1$, $\kappa_b = 1$, $c^{\mathrm{O}}_b = 1$, and $c^{\mathrm{I}}_b = 0$, and one demand k from o to d with $q_k = 1$, $\mathcal{B}_k = \{b\}$, and $c^{\mathrm{U}}_k = 100$, so that $\overline{f}_b = q_k = 1$. Both yards have $T_i = H_i = 1$ and $c^{\mathrm{H}}_i = 0$. The path $o \to n_1 \to d$ has length 2 and the path $o \to n_2 \to d$ has length 4, so $\mathcal{R}_b$ contains only the first.

Consider any feasible solution of the ARC LP relaxation. Since $\mathcal{B}_k = \{b\}$, Eq. (1c) at o gives $g_{kb} = v_k$, Eq. (1l) gives $f_b = g_{kb}$, and Eq. (1p) with $\underline{f}_b = \overline{f}_b = 1$ gives $f_b = y_b$, so $v_k = y_b$. For every arc a, Eq. (1n) gives $w_{ba} \le x^{\mathrm{B}}_{ba}$, and Eq. (1o) gives $w_{ba} \ge f_b - (y_b - x^{\mathrm{B}}_{ba}) = x^{\mathrm{B}}_{ba}$, so $w_{ba} = x^{\mathrm{B}}_{ba}$. Eq. (1i) at n_1, n_2, and o gives $x^{\mathrm{B}}_{b,(o,n_1)} = x^{\mathrm{B}}_{b,(n_1,d)} =: \chi_1$, $x^{\mathrm{B}}_{b,(o,n_2)} = x^{\mathrm{B}}_{b,(n_2,d)} =: \chi_2$, and $\chi_1 + \chi_2 = y_b$. Eq. (1k) gives $2\chi_1 + 4\chi_2 \le 3y_b$, and substituting $\chi_2 = y_b - \chi_1$ yields $\chi_1 \ge y_b/2$. Since $c^{\mathrm{H}}_i = c^{\mathrm{I}}_b = 0$, the objective equals $y_b + 10\chi_1 + 2\chi_2 + 100(1 - v_k) = 100 - 97y_b + 8\chi_1 \ge 100 - 93y_b \ge 7$, where the last step uses $y_b \le 1$. The value 7 is attained by $x^{\mathrm{D}}_k = x^{\mathrm{D}}_{kb} = y_b = v_k = g_{kb} = f_b = 1$ and $x^{\mathrm{B}}_{ba} = w_{ba} = 1/2$ on all four arcs, with logical positions $\ell^{\mathrm{D}}_{ko} = 0$ and $\ell^{\mathrm{D}}_{kd} = 1$, physical positions $\ell^{\mathrm{B}}_{bn} = 0$ for every node n, and $v^{\mathrm{H}}_o = v^{\mathrm{H}}_d = 0$. For this point, Eqs. (1i) and (1j) hold because each of the two paths carries $1/2$, Eq. (1k) holds because $(1+1)/2 + (2+2)/2 = 3$, Eq. (1m) holds because its right side equals $1 - 4(1 - 1/2) = -1$, Eq. (1n) holds because $w_{ba} = 1/2 = \overline{f}_b x^{\mathrm{B}}_{ba} \le f_b$, and Eq. (1o) holds because its right side equals $1 - (1 - 1/2) = 1/2$. Eqs. (1b)–(1h) hold because the single block carries the whole demand from o to d with positions 0 and 1, Eqs. (1l) and (1p) hold because $f_b = g_{kb} = 1$, each link carries $1/2 \le 1$, the track constraint at o reads $1 \le 1$, and Eq. (1q) gives $v^{\mathrm{H}}_o = g_{kb} - v_k = 0$ and $v^{\mathrm{H}}_d = 0$, within $H_o = H_d = 1$. All positions lie within the bounds in Eq. (1u), since $|\mathcal{Y}| - 1 = 1$ and $|\mathcal{N}| - 1 = 3$, and all relaxed binary variables lie in $[0, 1]$. Hence $z^{\mathrm{lp}}_{\mathrm{arc}} = 7$.

In the DW LP relaxation, $\mathcal{P}_k$ contains the single path $p = (b)$ and $\mathcal{R}_b$ contains the single path $r = o \to n_1 \to d$, with $c^{\mathrm{D}}_{kp} = -100$ and $c^{\mathrm{B}}_{br} = 10$. Eq. (2g) with $\underline{f}_b = \overline{f}_b = 1$ gives $v^{\mathrm{B}}_{br} = x^{\mathrm{B}}_{br}$, and Eq. (2b) gives $v^{\mathrm{D}}_{kp} = v^{\mathrm{B}}_{br}$. The objective therefore equals $100 - 100x^{\mathrm{B}}_{br} + x^{\mathrm{B}}_{br} + 10x^{\mathrm{B}}_{br} = 100 - 89x^{\mathrm{B}}_{br} \ge 11$, because $x^{\mathrm{B}}_{br} \le 1$ by Eq. (2i). Setting all four path variables to one attains 11 and satisfies every DW constraint. Eqs. (2b) and (2f)–(2j) hold with all four values equal to one, the two links of r carry $1 \le 1$ and the other two links carry $0 \le 1$, the track constraint at o reads $1 \le 1$, and the handling constraints read $0 \le 1$ because $\eta_{ip} = 0$ for the direct path p. Hence $z^{\mathrm{lp}}_{\mathrm{dw}} = 11 > 7 = z^{\mathrm{lp}}_{\mathrm{arc}}$. □

EC.1.2. Proofs of Propositions 1 and 2

Proof. We first verify that label costs equal path JRCs. For $p \in \mathcal{P}_k$, substituting $c^{\mathrm{D}}_{kp} = \sum_i c^{\mathrm{H}}_i \eta_{ip} - c^{\mathrm{U}}_k$ into Eqs. (3) and (5) gives

$$\widetilde{c}^{\mathrm{D}}_{kp} = -q_k c^{\mathrm{U}}_k - \pi_k + \sum_{b \in \mathcal{B}_k} \iota_{bp}\big(q_k \sigma_b - \gamma_{kb}\big) + q_k \sum_{i \in \mathcal{Y}} \big(c^{\mathrm{H}}_i - \mu^{\mathrm{H}}_i\big)\eta_{ip}.$$

Each block $b \in \mathcal{B}_k$ belongs to exactly one set $\mathcal{B}_k^+(i)$, the one with $i = i_b$, so the definition of η_{ip} gives $\sum_i (c_i^{\mathrm{H}} - \mu_i^{\mathrm{H}})\eta_{ip} = \sum_b \iota_{bp} \mathbf{1}\{i_b \neq o_k\}(c_{i_b}^{\mathrm{H}} - \mu_{i_b}^{\mathrm{H}})$. Hence $\widetilde{c}_{kp}^{\mathrm{D}} = \widetilde{c}_k^{\mathrm{D,init}} + \sum_b \iota_{bp} \widetilde{c}_{kb}^{\mathrm{D}}$, with the extension costs in Eq. (6). Similarly, substituting $c_{br}^{\mathrm{B}} = c_b^{\mathrm{I}} + \sum_a c_{ba}^{\mathrm{T}} \delta_{ar}$ into Eqs. (7) and (8) gives $\widetilde{c}_{br}^{\mathrm{B}}(t) = \widetilde{c}_{bt}^{\mathrm{B,init}} + \sum_a \delta_{ar} \widetilde{c}_{bta}^{\mathrm{B}}$, with the extension costs in Eq. (9). By the extension rules in Sections 4.2.1 and 4.2.2, each logical label represents an elementary prefix from o_k that uses compatible blocks, each physical label represents an elementary prefix from i_b of the recorded length within the length budget, and the cost of a label equals the initial cost plus the extension costs along its prefix. Physical labels are generated only for blocks with $\overline{f}_b \geq \underline{f}_b$ (Section 4.2.2), so a complete physical path within the length budget belongs to $\mathcal{R}_b$. A *completion* of a label is a possibly empty sequence of blocks or arcs whose concatenation with the prefix is a path in $\mathcal{P}_k$ or $\mathcal{R}_b$, respectively. A label at the destination has only the empty completion, since an elementary path ends there. If $\Delta(Q)$ denotes the sum of the extension costs along a completion Q, the decompositions above show that the completed path has JRC $c + \Delta(Q)$, where c is the cost of the label. Let $c^\star(L)$ be the minimum of $c + \Delta(Q)$ over the completions Q of label L. Here and below, a minimum over the empty set equals $+\infty$.

Now let $L_1 \preceq L_2$ be two logical labels at yard i, as in Proposition 1, and let Q be a completion of L_2. Because the path completed from L_2 is elementary, each yard of Q other than i lies outside S_2, and therefore outside $S_1 \subseteq S_2$. Appending Q to the prefix of L_1 thus yields an elementary path from o_k to d_k with compatible blocks, so Q is also a completion of L_1. The extension costs of Q depend only on k and the blocks of Q, so the two completed paths have JRCs $c_1 + \Delta(Q) \leq c_2 + \Delta(Q)$, whatever the sign of $\Delta(Q)$. Hence $c^\star(L_1) \leq c^\star(L_2)$, which is the sense in which L_1 dominates L_2. For two physical labels $L_1 \preceq L_2$ at node n, as in Proposition 2, the same argument applies with nodes in place of yards and with extension costs that depend only on b, t, and the arcs of Q. In addition, the path completed from L_1 has length $d_1 + D(Q) \leq d_2 + D(Q) \leq \overline{D}_b$, where $D(Q)$ is the length of Q. Hence Q is a completion of L_1 within the length budget, and again $c^\star(L_1) \leq c^\star(L_2)$.

It remains to show that discarding dominated labels preserves the minimum JRC computed by the labeling algorithm. A label at the destination is stored and not extended. Every other label is eventually extended unless it is discarded, and a label L is discarded only when a stored label $L_1 \preceq L$ exists that has not been discarded, whether or not L_1 has been extended. The algorithm stops because every extension enlarges the visited set, which is contained in $\mathcal{Y}$ or $\mathcal{N}$. Let $\mathcal{D}$ be the set of labels at the destination that are never discarded. Each label in $\mathcal{D}$ represents a path in $\mathcal{P}_k$ or $\mathcal{R}_b$, so its cost is at least the minimum JRC. We show, by induction on the number of blocks or arcs of Q, that for every generated label L and every completion Q of L, some label in $\mathcal{D}$ has cost at most $c + \Delta(Q)$, where c is the cost of L. Applied to L_0, whose completions are all paths in $\mathcal{P}_k$ or $\mathcal{R}_b$, this claim shows that the minimum cost over $\mathcal{D}$ equals the minimum JRC.

Assume the claim for all completions shorter than Q. Suppose first that L is never discarded. If Q is empty, then L lies at the destination and belongs to $\mathcal{D}$. Otherwise, L is not at the destination and is therefore extended. Its extension L' along the first block or arc of Q is generated, because the head of that block or

arc lies outside the visited set of L and, for a physical label, the extended prefix stays within the length budget, since the remaining arcs of Q have positive lengths. The rest Q' of Q is a completion of L', the cost c' of L' satisfies $c' + \Delta(Q') = c + \Delta(Q)$, and Q' is shorter than Q, so the induction hypothesis gives the claim. Suppose next that L is discarded. At that time, some label $L_1 \preceq L$ exists that has not been discarded. If L_1 is discarded later, some label $L_2' \preceq L_1$ exists that has not been discarded, and so on. The discard times increase strictly along this chain, so the chain ends at a label L^* that is never discarded. By transitivity of $\preceq$, $L^* \preceq L$, so the preceding paragraph shows that Q is a completion of L^* with cost $c^* + \Delta(Q) \le c + \Delta(Q)$, where c^* is the cost of L^*. The first case, applied to L^* and Q, gives the claim. □

EC.1.3. Proof of Proposition 3

Proof. Fix a block b with $\overline{f}_b \ge \underline{f}_b$. Every completion Q of a physical label $L = (n, S, d, c)$ is a directed path in G^{P} from n to j_b, possibly empty, so its length satisfies $D(Q) \ge \underline{D}_b(n)$, the shortest length from n to j_b without the visited-node restriction. If no directed path from n to j_b exists, then L has no completion and $\underline{D}_b(n) = +\infty$. The distances $\underline{D}_b(n)$ are therefore lower bounds on the lengths of all completions. Dijkstra's algorithm from j_b on the reversed graph computes $\underline{D}_b(n)$ for every node n, because reversing the arcs maps the directed paths from n to j_b onto the directed paths from j_b to n with the same lengths, and all lengths D_a are positive. These distances depend only on G^{P}, the arc lengths, and j_b, so they can be reused for every load and every set of dual prices.

If $d + \underline{D}_b(n) > \overline{D}_b$, every completion Q of L would give a complete path of length $d + D(Q) \ge d + \underline{D}_b(n) > \overline{D}_b$, which lies outside $\mathcal{R}_b$, so L has no completion. Similarly, the extension of L along an arc $a \in \mathcal{A}^+(n)$ is a label at j_a of length $d + D_a$, and each of its completions has length at least $\underline{D}_b(j_a)$. If $d + D_a + \underline{D}_b(j_a) > \overline{D}_b$, this extension therefore has no completion. Both tests thus remove only labels without completions. In the induction at the end of Section EC.1.2, the claim is vacuous for such labels, and the extension L' used there is never rejected, because it has the completion Q'. The induction therefore holds with the tests, and the labeling algorithm still returns the minimum physical path JRC. □

EC.1.4. Proof of Proposition 4

Proof. We use LP duality for the DW LP relaxation. We call its constraints other than the path volume bounds in Eqs. (2f) and (2g) the *master rows*, i.e., Eqs. (2b)–(2e) and (2h)–(2j). The RMP contains the master rows, restricted to its paths, and the volume bounds of its paths. When rows of Eq. (2j) are separated as in Section 6.1, the argument below applies once no omitted row is violated, because the omitted rows can then be added with zero dual variables without changing the optimal primal and dual solutions or their objective values. In a minimization LP, the dual variable of a $\le$ constraint is nonpositive, that of an equality is free, and the reduced cost of a variable equals its objective coefficient minus the sum of its constraint coefficients weighted by the dual variables. The dual objective is the sum of the right-hand sides weighted by the dual variables, plus the objective constant $\sum_k q_k c_k^{\mathrm{U}}$.

Step 1. We derive the reduced costs. The selection variable x^{D}_{kp} has objective coefficient zero, coefficient one in Eq. (2h) for k, coefficient ι_{bp} in Eq. (2j) for each pair (k,b), and coefficient $-q_k$ in its row $v^{\mathrm{D}}_{kp} - q_k x^{\mathrm{D}}_{kp} \le 0$ of Eq. (2f), whose dual variable is $\lambda_{kp} \le 0$. The volume variable v^{D}_{kp} has objective coefficient c^{D}_{kp}, coefficient $-\iota_{bp}$ in Eq. (2b) for each block b, coefficient η_{ip} in Eq. (2d) for each yard i, and coefficient one in the same bound row. Subtracting the dual-weighted coefficients from the objective coefficients gives the reduced costs in Eq. (4), with the coefficients in Eq. (3). For a physical path $r \in \mathcal{R}_b$, we write the bounds in Eq. (2g) as $\underline{f}_b x^{\mathrm{B}}_{br} - v^{\mathrm{B}}_{br} \le 0$ and $v^{\mathrm{B}}_{br} - \overline{f}_b x^{\mathrm{B}}_{br} \le 0$, with dual variables $\underline{\lambda}_{br}, \overline{\lambda}_{br} \le 0$. The variable x^{B}_{br} has objective coefficient c^{O}_b, coefficient one in Eq. (2i) for b, coefficient κ_b in Eq. (2e) for i_b, coefficient -1 in Eq. (2j) for each $k \in \mathcal{K}_b$, and coefficients $\underline{f}_b$ and $-\overline{f}_b$ in its two bound rows. The variable v^{B}_{br} has objective coefficient c^{B}_{br}, coefficient one in Eq. (2b) for b, coefficient $\sum_{a \in \mathcal{A}_e} \delta_{ar}$ in Eq. (2c) for each link e, and coefficients -1 and 1 in the two bound rows. Since each arc belongs to exactly one link, $\sum_e \mu_e \sum_{a \in \mathcal{A}_e} \delta_{ar} = \sum_a \mu_{e_a} \delta_{ar}$, and the reduced costs are

$$\mathrm{rc}(x^{\mathrm{B}}_{br}) = \bar{c}^{\mathrm{B},x}_{br} - \underline{f}_b \underline{\lambda}_{br} + \overline{f}_b \overline{\lambda}_{br}, \qquad \mathrm{rc}(v^{\mathrm{B}}_{br}) = \bar{c}^{\mathrm{B},v}_{br} + \underline{\lambda}_{br} - \overline{\lambda}_{br},$$

with the coefficients in Eq. (7).

Step 2. We fix the dual variables of the master rows and characterize when the bound rows of a single path admit dual variables that make the reduced costs of both of its variables nonnegative. For a logical path, such a $\lambda_{kp} \le 0$ exists if and only if $\bar{c}^{\mathrm{D},x}_{kp} \ge 0$ and $\widetilde{c}^{\mathrm{D}}_{kp} \ge 0$. If such a λ_{kp} exists, then $\bar{c}^{\mathrm{D},x}_{kp} = \mathrm{rc}(x^{\mathrm{D}}_{kp}) - q_k \lambda_{kp} \ge 0$ and $\widetilde{c}^{\mathrm{D}}_{kp} = \mathrm{rc}(x^{\mathrm{D}}_{kp}) + q_k \,\mathrm{rc}(v^{\mathrm{D}}_{kp}) \ge 0$. Conversely, $\lambda_{kp} = -\bar{c}^{\mathrm{D},x}_{kp}/q_k \le 0$ gives $\mathrm{rc}(x^{\mathrm{D}}_{kp}) = 0$ and $\mathrm{rc}(v^{\mathrm{D}}_{kp}) = \widetilde{c}^{\mathrm{D}}_{kp}/q_k \ge 0$. Moreover, $\bar{c}^{\mathrm{D},x}_{kp} = -\pi_k - \sum_b \gamma_{kb}\iota_{bp} \ge 0$ holds for every logical path because $\pi_k, \gamma_{kb} \le 0$, so the condition reduces to $\widetilde{c}^{\mathrm{D}}_{kp} \ge 0$. For a physical path, such $\underline{\lambda}_{br}, \overline{\lambda}_{br} \le 0$ exist if and only if $\widetilde{c}^{\mathrm{B}}_{br}(\underline{f}_b) \ge 0$ and $\widetilde{c}^{\mathrm{B}}_{br}(\overline{f}_b) \ge 0$. By Step 1, for every t,

$$\mathrm{rc}(x^{\mathrm{B}}_{br}) + t\,\mathrm{rc}(v^{\mathrm{B}}_{br}) = \widetilde{c}^{\mathrm{B}}_{br}(t) + (t - \underline{f}_b)\underline{\lambda}_{br} + (\overline{f}_b - t)\overline{\lambda}_{br}.$$

If both reduced costs are nonnegative, then for $t \in \{\underline{f}_b, \overline{f}_b\}$ the left side is nonnegative and the last two terms are nonpositive, so $\widetilde{c}^{\mathrm{B}}_{br}(t) \ge 0$. Conversely, if $\bar{c}^{\mathrm{B},v}_{br} \ge 0$, the choice $\underline{\lambda}_{br} = -\bar{c}^{\mathrm{B},v}_{br}$ and $\overline{\lambda}_{br} = 0$ gives $\mathrm{rc}(v^{\mathrm{B}}_{br}) = 0$ and $\mathrm{rc}(x^{\mathrm{B}}_{br}) = \widetilde{c}^{\mathrm{B}}_{br}(\underline{f}_b) \ge 0$. If $\bar{c}^{\mathrm{B},v}_{br} < 0$, the choice $\underline{\lambda}_{br} = 0$ and $\overline{\lambda}_{br} = \bar{c}^{\mathrm{B},v}_{br}$ gives $\mathrm{rc}(v^{\mathrm{B}}_{br}) = 0$ and $\mathrm{rc}(x^{\mathrm{B}}_{br}) = \widetilde{c}^{\mathrm{B}}_{br}(\overline{f}_b) \ge 0$.

Step 3. We prove the proposition. Take an optimal primal solution of the RMP and the optimal RMP dual solution whose master-row dual variables define the JRCs. By hypothesis and Step 2, every path in $\mathcal{P}_k$ or $\mathcal{R}_b$, whether or not it belongs to the RMP, admits bound dual variables that make the reduced costs of its two variables nonnegative under these master-row dual variables. Together, the master-row dual variables and these bound dual variables form a feasible dual solution of the DW LP relaxation. Its objective value equals that of the optimal RMP dual solution, because the master rows coincide and the bound rows have zero right-hand sides, and therefore equals the optimal RMP value by LP duality. Extending the RMP primal solution with zero values for all paths outside the RMP gives a feasible solution of the DW LP relaxation

with the same objective value, because the added variables contribute nothing to the objective or to the master rows and satisfy their bound rows as $0 \le 0$. By weak duality, the extended solution is optimal for the DW LP relaxation, so the RMP value equals $z^{\mathrm{lp}}_{\mathrm{dw}}$. Since the DW LP relaxation relaxes the binary domains of DW, $z^{\mathrm{lp}}_{\mathrm{dw}} \le z^{*}_{\mathrm{dw}}$. $\square$

Algorithm 1 terminates, and by Proposition 4 its final RMP solution is optimal for the DW LP relaxation. Every RMP is feasible, since setting all path variables to zero satisfies its rows, the capacities being nonnegative, and Eqs. (2f)–(2i) bound all variables, so every RMP has an optimal solution. For a path in the RMP, the optimal RMP dual solution provides bound dual variables that make the reduced costs of its two variables nonnegative, so Step 2 shows that its JRC is nonnegative, at both load bounds for a physical path. Exact pricing returns only paths with negative JRC, so every iteration that does not stop adds paths outside the RMP. Since every $\mathcal{P}_k$ and $\mathcal{R}_b$ is finite, exact pricing eventually returns no path. By Propositions 1–3 and the single search for both load bounds in Section 4.2.2, exact pricing minimizes the logical JRC over every $\mathcal{P}_k$ and the physical JRC over every $\mathcal{R}_b$ at both load bounds, so an empty return means that every JRC is nonnegative, and Proposition 4 applies. If rows of Eq. (2j) are separated, each RMP solve adds violated rows until none remains, which takes finitely many solves because these rows are finite in number.

EC.2. Heuristic Details

EC.2.1. *Construction* Details

The initial restricted block domain in Section 5.1 uses compatible direct connections and at most eight hub proposals per demand. Hubs are ranked by compatible volume support divided by summed path length budgets. The full physical graph, original capacities, demands, and individual compatibility are retained.

The physical candidates $\mathcal{R}^{\mathrm{C}}_b \subseteq \mathcal{R}_b$ of a block b are generated by shortest-path searches when b is first considered. Let $q^{\max}_b = \max\{\underline{f}_b, \max_{k\in\mathcal{K}_b} q_k\}$. For each load level $t \in \{\underline{f}_b, \min\{\overline{f}_b, q^{\max}_b\}, \min\{\overline{f}_b, 2q^{\max}_b\}\}$, in increasing order, we remove the arcs of the links with $C_e < t$ and run Dijkstra's algorithm from i_b to j_b with each of the arc weights D_a, c^{T}_{ba}, $c^{\mathrm{T}}_{ba} + D_a$, and $c^{\mathrm{T}}_{ba} + 4D_a$. If no path exists or the minimum-length path exceeds $\overline{D}_b$, no path within the length budget exists at this or any higher level, and the searches for b stop. If the minimum-cost path exceeds $\overline{D}_b$, we also examine the two cheapest elementary paths with respect to c^{T}_{ba} (Table EC.1). When the unit transportation costs of b are proportional to the arc lengths, all four weights yield the same path, and only the length search is run. A path joins $\mathcal{R}^{\mathrm{C}}_b$ if it is new, satisfies the length budget, and can carry the minimum block load, i.e., $\overline{f}_{br} \ge \underline{f}_b$ in Eq. (EC.5).

When SERVE evaluates a candidate assignment (Section 5.1.3), the open blocks keep their incumbent physical paths, and the newly opened blocks receive paths from their candidates one at a time, in increasing block index. SERVE compares two selection rules, both of which account for the paths selected earlier. The first selects the cheapest candidate that keeps every link within capacity at the smallest value of θ allowed by the block load bounds and handling capacities. The second selects the candidate that keeps the largest value of θ within the link capacities, breaking ties by unit cost. For each rule, SERVE computes $[\theta_-, \theta_+]$ with the selected paths and applies Eq. (13), and it keeps the cheaper of the two resulting plans.

EC.2.2. Search Schedule

After *construction*, DG-CGM repeats search cycles until the time limit, using the controls in Table EC.1. Each cycle starts with a local search stage. It then performs pricing rounds, each consisting of one *path generation* invocation followed by a local search stage, until two consecutive rounds each fail to lower the incumbent cost by more than $\varepsilon_{\mathrm{H}} \max\{1, |z|\}$, where z is the incumbent cost at the start of the round. *Joint repair* then runs once, and the next cycle starts.

A local search stage alternates *destroy* and *local repair* for at most $|\mathcal{K}|$ calls within 30 seconds, and each local MIP has a time limit of one second. A call is improving if it lowers the incumbent cost by more than $\varepsilon_{\mathrm{H}} \max\{1, |z|\}$, where z is the incumbent cost before the call. Every strict feasible cost decrease is retained. In each stage, the group size limit N_{D} of Section 5.2 starts at eight, rises to twelve after two consecutive nonimproving calls, and returns to eight after an improving call. The stage ends after three consecutive nonimproving calls.

A *path generation* invocation is skipped when the incumbent and the candidate path pools coincide with those at the start of the most recent invocation that ended with an optimal LP and a complete pricing sweep, since it would return the same paths. Its round then consists of the local search stage alone. The 30-second allowance of a *joint repair* invocation starts before the candidate searches of Section 5.5, so the joint MIP receives the remaining time. The search for alternative physical paths visits the open blocks in turn and stops after 0.1 seconds in total. For each retained physical path of a block and each node of this path, it keeps the prefix up to the node, removes the prefix nodes and the arcs by which retained paths with the same prefix continue, and computes a minimum-cost suffix with arc costs c^{T}_{ba} over the links with $C_e \geq \underline{f}_b$. If the resulting path violates the length budget, revisits a node, or cannot carry $\underline{f}_b$, the suffix search is repeated with arc costs $c^{\mathrm{T}}_{ba} + D_a$, $c^{\mathrm{T}}_{ba} + 4D_a$, and D_a. The two cheapest new paths of each block join its pool. Section EC.2.3 describes the search for partner chains. The joint MIP is skipped when it coincides with the most recent joint MIP whose search was completed, that is, which was solved to optimality or shown to contain no plan cheaper than the incumbent.

The search stops at the time limit or after a cycle in which every *path generation* invocation and the joint MIP are skipped and the candidate searches before *joint repair* add no path.

EC.2.3. Minimum-Load Partner Chains

Partner chains are generated before each joint MIP from the incumbent and the shared logical pools $\widetilde{\mathcal{P}}_j$. Let $\mathcal{B}^{\mathrm{O}}$ be the set of open blocks, $c^{\min}_b = \min_{r \in \widetilde{\mathcal{R}}_b} c^{\mathrm{B}}_{br}$, and $\bar{c}_j$ the incumbent unit cost of demand j in Eq. (11). For a path $p \in \widetilde{\mathcal{P}}_j$, let $\mathcal{D}_{jp}$ contain its blocks $b \notin \mathcal{B}^{\mathrm{O}}$ with $q_j < \underline{f}_b$, which demand j cannot open alone, and let

$$\Delta_{jp} = \bar{v}_j \bar{c}_j + (q_j - \bar{v}_j) c^{\mathrm{U}}_j + \sum_{b \in \mathcal{B}^{\mathrm{excl}}_j \setminus p} c^{\mathrm{O}}_b - q_j \sum_{b \in p} \big(c^{\min}_b + \mathbf{1}\{i_b \neq o_j\} c^{\mathrm{H}}_{i_b} \big) - \sum_{b \in p \setminus \mathcal{B}^{\mathrm{O}}} c^{\mathrm{O}}_b$$

be the estimated saving from moving all of q_j to p, where $\mathcal{B}_j^{\text{excl}}$ contains the blocks of $\bar{p}_j$ that carry no other served demand. For a demand k and a block $b \in \mathcal{B}_k$ with $\widetilde{\mathcal{R}}_b \neq \emptyset$, the reward

$$R_{kb} = \max\Big(\{0\} \cup \Big\{ \frac{\Delta_{jp}}{|\mathcal{D}_{jp}|} : j \neq k,\ p \in \widetilde{\mathcal{P}}_j,\ \Delta_{jp} > 0,\ b \in \mathcal{D}_{jp},\ q_j + q_k \geq \underline{f}_b \Big\}\Big) \tag{EC.4}$$

assigns to b the largest share of such a saving among the paths of other demands that need b and can open it together with k. For each demand k with a positive reward in Eq. (EC.4) for some b, a labeling search finds the cheapest elementary paths from o_k to d_k over the compatible blocks with retained physical paths, where block b costs

$$q_k\big(c_b^{\min} + \mathbf{1}\{i_b \neq o_k\} c_{i_b}^{\mathrm{H}}\big) - R_{kb} + \mathbf{1}\{b \notin \mathcal{B}^{\mathrm{O}},\ R_{kb} = 0\} c_b^{\mathrm{O}}.$$

An unopened block is charged its opening cost only when it carries no reward, because a positive reward is computed after the opening costs in Δ_{jp} have been deducted. The three cheapest paths that are not yet in $\widetilde{\mathcal{P}}_k$ join the pool, and the searches for all demands stop after 0.1 seconds in total. The rewards only guide this search, and the joint MIP evaluates the resulting paths with the original objective and constraints.

EC.2.4. Arc Options in *Joint Repair*

The arc option of Section 5.5 gives each released demand k a copy $(x_k^{\mathrm{D,arc}}, x_{kb}^{\mathrm{D,arc}}, g_{kb}^{\mathrm{arc}}, v_k^{\mathrm{arc}}, \ell_{ki}^{\mathrm{D,arc}})$ of its logical variables in ARC. The copy satisfies Eqs. (1b)–(1g) over the compatible blocks with retained physical paths, and $x_{kb}^{\mathrm{D,arc}} \leq \sum_{r \in \widetilde{\mathcal{R}}_b} x_{br}^{\mathrm{B}}$ replaces Eq. (1h). The constraints $\sum_{p \in \widetilde{\mathcal{P}}_k} x_{kp}^{\mathrm{D}} + x_k^{\mathrm{D,arc}} \leq 1$ for each released demand and $\sum_{k \in \mathcal{K}_{\text{free}}} x_k^{\mathrm{D,arc}} \leq 1$ allow each demand at most one of its options and at most one demand the arc option. The copy enters the other constraints and the objective as a logical path does. The volume g_{kb}^{arc} is added to the volumes of the released demands on block b in the block balance of Eq. (15), the reclassified volume $\sum_{b \in \mathcal{B}_k^+(i)} g_{kb}^{\mathrm{arc}} - \mathbf{1}\{i = o_k\} v_k^{\mathrm{arc}}$ is added at each yard i to the handling capacity and the handling cost, and the served volume of k becomes $\sum_{p \in \widetilde{\mathcal{P}}_k} v_{kp}^{\mathrm{D}} + v_k^{\mathrm{arc}}$ in the unserved demand cost. The size estimate s_{arc} is the number of variables of the augmented MIP plus an upper bound on its number of constraints, computed before the model is built.

EC.2.5. Physical Path Load Limits

An elementary physical path visits no node twice, so it traverses each link at most once, that is, $\sum_{a \in \mathcal{A}_e} \delta_{ar} \in \{0, 1\}$. A path r selected for block b therefore carries the full block load on every link that it traverses, and the restricted MIPs of *local repair* and *joint repair* use the load limit

$$\overline{f}_{br} = \min\Big\{ \overline{f}_b, \min_{e \in \mathcal{E}:\, \sum_{a \in \mathcal{A}_e} \delta_{ar} = 1} C_e \Big\} \tag{EC.5}$$

in place of $\overline{f}_b$ in Eq. (2g), while the physical candidate sets exclude paths with $\overline{f}_{br} < \underline{f}_b$. These changes remove no integer feasible solution of a restricted MIP. If such a solution selects path r for block b, then for every link e traversed by r, the link row of Eq. (15), with the fixed loads moved back to its left side, contains the term v_{br}^{B} and further nonnegative terms, so $v_{br}^{\mathrm{B}} \leq C_e$. Together with $v_{br}^{\mathrm{B}} \leq \overline{f}_b$, this gives $v_{br}^{\mathrm{B}} \leq \overline{f}_{br}$. If the solution does not select r, then $v_{br}^{\mathrm{B}} = 0$. A path with $\overline{f}_{br} < \underline{f}_b$ can therefore never be selected, because

Table EC.1 Default search controls.

Control	Default
Related-demand score weights $(\omega_0, \omega_1, \omega_2)$	$(10^4, 10^3, 500)$
Initial demand-group size limit	8
Consecutive nonimprovements triggering expansion	2
Expanded demand-group size limit	12
Stage-control relative-improvement threshold ε_{H}	10^{-4}
Consecutive nonimprovements ending a local search stage	3
Maximum number of local MIP calls per local search stage	$\lvert\mathcal{K}\rvert$
Time allowance per local search stage	30 s
Nonimproving completed pricing rounds triggering *joint repair*	2
Time limit per local MIP	1 s
Cheapest retained physical paths per released block	3
Alternative physical paths per open block before *joint repair*	2
Time limit for this physical path search	0.1 s
Partner chains per demand before *joint repair*	3
Time limit for the partner chain search	0.1 s
Regenerated lowest-EUC logical paths per demand in each repair	16
Construction stopping tolerance ε_{C}	10^{-6}
Hub yards for two-hub groups N_{hub}	16
Candidate blocks above which *construction* restricts the block set	10,000
New blocks permitted in an expanded logical chain	2
Initial logical-path limit per demand N_{LP}	4
Return cap per demand N_{add}	12
Mixed-selection JRC quota N_{JRC}	8
Mixed-selection diversity quota N_{div}	2
Mixed-selection EUC quota N_{EUC}	2
Successor-yard neighborhood size N_{nbr}	32
Soft time allowance per RMP iteration τ_{PG}	10 s
New logical paths per LP batch N_{batch}	1,000
Maximum pricing batches per invocation I_{PG}	10
LP plateau window W_{PG}	3
Relative LP plateau tolerance $\varepsilon_{\mathrm{PG}}$	10^{-5}
Pricing-batch share of remaining overall time	0.2
Maximum demands released in *joint repair* N_{J}	250
Joint MIP size limit for arc options N_{arc}	10^5
Cost-ranked shortest paths examined after an initial length violation	2
Time allowance per *joint repair* invocation	30 s

it would require $\underline{f}_b \le v^{\mathrm{B}}_{br} \le \overline{f}_{br}$. Since $\overline{f}_{br} \le \overline{f}_b$, the load limits can only shrink the LP relaxations of the restricted MIPs, and $\overline{f}_{br}$ can only decrease when link capacities are reduced.

In each *local repair* call, the candidate set $\widetilde{\mathcal{R}}_b$ of a block $b \in \mathcal{B}_{\mathrm{free}}$ contains up to three of its cheapest retained physical paths, a retained path with the largest $\overline{f}_{br}$, and its incumbent path.

EC.2.6. Default Search Controls

Table EC.1 lists the default controls of DG-CGM, which all experiments use unless stated otherwise.

EC.2.7. Sensitivity to Search Controls

Table EC.2 reports the one-at-a-time variants of Section 6.2.4. Each variant changes one control of Table EC.1 and keeps all others, the 60-second time limit, one solver thread, and seed 1 on the 150 generated instances. Diff. is the mean paired relative cost difference from the default configuration, W/T/L counts the instances on which the variant is cheaper, equal, or more expensive, and p is the two-sided Wilcoxon signed-rank p-value, reported when at least six instances differ.

Table EC.2 One-at-a-time sensitivity of DG-CGM to its search controls.

Control	Default	Value	Mean gap/%	Max. gap/%	#Opt.	Time/s	Diff./%	W/T/L	p
N_{LP}	4	2	0.10	5.8	89	49.3	−0.05	5/142/3	0.148
N_{LP}	4	8	0.10	5.8	89	51.1	−0.04	4/144/2	0.219
$(N_{\mathrm{add}}, N_{\mathrm{JRC}}, N_{\mathrm{div}}, N_{\mathrm{EUC}})$	(12,8,2,2)	(6,4,1,1)	0.06	2.8	89	50.7	−0.08	6/142/2	0.078
$(N_{\mathrm{add}}, N_{\mathrm{JRC}}, N_{\mathrm{div}}, N_{\mathrm{EUC}})$	(12,8,2,2)	(24,16,4,4)	0.09	4.5	89	50.6	−0.06	5/144/1	0.094
N_{nbr}	32	16	0.12	5.8	89	50.7	−0.03	4/143/3	0.297
N_{nbr}	32	64	0.15	5.8	89	50.8	0.00	0/150/0	–
I_{PG}	10	5	0.09	5.8	91	51.4	−0.05	6/142/2	0.039
I_{PG}	10	20	0.17	5.8	88	50.7	0.03	1/142/7	0.016
τ_{PG}/s	10	5	0.12	4.2	88	50.7	−0.03	1/148/1	–
τ_{PG}/s	10	20	0.15	5.8	89	51.0	0.00	0/150/0	–
Group size limits	(8,12)	(4,8)	0.14	5.8	89	50.1	−0.01	5/141/4	0.820
Group size limits	(8,12)	(12,16)	0.14	5.8	88	48.1	−0.01	1/147/2	–
ε_{H}	10^{-4}	10^{-3}	0.14	5.8	89	50.5	−0.01	1/148/1	–
ε_{H}	10^{-4}	10^{-5}	0.14	5.8	88	50.7	−0.01	1/147/2	–
Default	–	–	0.15	5.8	89	50.8	–	–	–

EC.3. Details of Instance Generation

The generator takes a random seed and the target numbers of nodes and demands as input. It estimates all distributions from data set L1 (Section 6.1) and keeps the network-wide parameters of L1, such as the cost coefficients, unchanged. Each L1 node is a yard, a gate, or a micro node, each yard has one of three levels, and every L1 demand connects yards of levels 1 and 2. We treat the L1 network as an undirected simple graph that keeps the shortest of any parallel links, so that the degree of a node is its number of neighbors. In its largest connected component, we replace every maximal path whose interior nodes are micro nodes of degree two by one link of the same total length, again keeping the shortest of any parallel links, and retain all other nodes. In the resulting simplified real network, we partition the nodes once, for all instances, into six regions by k-means (MacQueen 1967) on their coordinates in miles under an equirectangular projection. Region g has $\widehat{n}_g$ nodes, and $\widehat{m}_{gh}$ and $\widehat{s}_{gh}$ denote the mean and standard deviation, over these nodes, of the fraction of a node's neighbors that lie in region h. The demands are apportioned by the largest-remainder method among commodities and then among pairs of origin and destination yard levels, in proportion to the numbers of L1 demand records, and each commodity and level pair with demands forms a demand category.

Each generated node joins region g with probability $\widehat{n}_g / \sum_h \widehat{n}_h$, and a node i in region g draws a preference $\varphi_{ih} \sim \mathrm{Normal}(\widehat{m}_{gh}, \widehat{s}^2_{gh})$ for links to each region h, clipped to $[0, 1]$ and normalized to sum to one. If

n_g generated nodes lie in region g, the nodes of this region in the simplified real network, sorted by degree, supply reference records at positions $1+\lfloor(t-\frac{1}{2})\widehat{n}_g/n_g\rfloor$, $t=1,\dots,n_g$, which are assigned to the generated nodes of the region in random order. GenCAT (Maekawa et al. 2023) then calibrates the preferences of each region to $\widehat{m}_{gh}$ and adds links according to them, up to a target degree for each node equal to the degree of its reference record. Each link can be traversed in both directions.

The reference record of a generated node also supplies its node type and coordinates, with each coordinate perturbed by independent Gaussian noise whose standard deviation is 0.035 times that of the coordinate over the region in the simplified real network; $\mathbf{r}_i$ denotes the result for node i. The nodes whose reference records are yards form $\mathcal{Y}$. To preserve the observed combinations of yard characteristics, we sample $|\mathcal{Y}|$ complete records of yards of the simplified real network with replacement within each level. The numbers of records per level are apportioned in the same way by the level shares of that network, after which levels 1 and 2 are enlarged at the expense of level 3 until the n records of each of these levels give at least as many distinct ordered pairs, $n(n-1)$, as any category with both endpoints at that level has demands. A minimum-cost one-to-one matching assigns these records to the generated yards, where assigning the record of yard j to generated yard i costs $3|\log[(1+\deg(i))/(1+\deg(j))]|+\|\mathbf{r}_i-\mathbf{r}_j^{\mathrm{ref}}\|_2/1{,}500$, with degrees in the respective networks and the coordinates $\mathbf{r}_j^{\mathrm{ref}}$ of yard j in miles. The assigned record provides the operator, level, classification track slots T_i, handling capacity H_i, and handling cost c_i^{H} of yard i.

The length of each arc a is first set to $D_a=\max\{0.01,\|\mathbf{r}_{i_a}-\mathbf{r}_{j_a}\|_2\}$ miles. All arc lengths are then multiplied by $\widehat{D}/D$, where $\widehat{D}$ and D are the median shortest-path distances over unordered pairs of distinct yards in the simplified real network and in the generated network, so that the two medians coincide. As in L1, every link has capacity $C_e=700{,}000$.

For each demand k, we sample an L1 demand record uniformly from its category. This record supplies an initial volume $q_k^{(0)}$, the shortest-path distance $\widehat{D}_k$ between its origin and destination in the simplified real network, and whether these two yards have the same operator. Among the ordered pairs (i,j) of distinct generated yards, with i at the origin level and j at the destination level of the category, that no earlier demand of the same commodity uses, we draw (o_k,d_k) with probability proportional to $\exp\{-3|\log(\max\{\mathrm{dist}(i,j),1\}/\max\{\widehat{D}_k,1\})|\}$, multiplied by $\exp(-1.5)$ if the pair and the record differ in whether their two yards have the same operator. Here, $\mathrm{dist}(i,j)$ is the shortest-path distance in miles in the generated network.

Within each commodity, the initial volumes divided by their mean define an outgoing and an incoming total at each yard. The conditional reconstruction method (CReMa) of Parisi et al. (2020), implemented in NEMtropy (Vallarano et al. 2021), fits yard parameters ν_i^+ and ν_i^- such that independent samples $X_k\sim\mathrm{Exp}(\nu_{o_k}^+ + \nu_{d_k}^-)$ of the demands, where Exp is parameterized by its rate, match these totals in expectation; every absolute deviation is at most 10^{-5} times the largest total. Let $\mathcal{K}'$ list the demands of the commodity in increasing order of X_k, and let $\widehat{Q}$ be the sample quantile function of the L1 volumes of the commodity,

interpolated linearly between order statistics. The t-th demand k in $\mathcal{K}'$ receives the requested volume $q_k = \max\{1, \lfloor \widehat{Q}((t-\frac{1}{2})/|\mathcal{K}'|) \rceil\}$, where $\lfloor \cdot \rceil$ rounds to the nearest integer, with ties to even. The final volumes thus follow the L1 volume distribution of the commodity, and the samples X_k determine only their order. An instance is valid if the network generated by GenCAT is connected, so that its largest connected component contains all generated nodes, and if it has at least eight yards, level 3 can supply the enlargement of levels 1 and 2, every category has at least as many ordered pairs of distinct yards at its levels as demands, and every fit meets the tolerance above.