LEARNING TO ENHANCE BRANCH-AND-BOUND SEARCH FOR DEMAND-RESPONSIVE TRANSIT ROUTING WITH HETEROGENEOUS GRAPH NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

The dial-a-ride problem (DARP) provides a mathematical framework for routing and scheduling in demand-responsive transport. However, its tightly coupled routing and service constraints make the search of high-quality feasible solutions computationally challenging. This paper develops a graph learning-guided primal search approach for branch-and-bound to accelerate DARP optimization. A heterogeneous graph is constructed by integrating arc variables, continuous state variables, constraints, passenger requests, and vehicles, enabling graph learning to capture both the algebraic structure of the mathematical formulation and the operational information characteristics of DARP. The learned arc-selection scores are transformed into confidence-aware search guidance and incorporated into variable hints and complementary neighborhood searches. Separate deviation budgets accommodate asymmetric prediction errors, while confidence-weighted local branching directs search toward promising assignments. Feasible candidates strengthen the incumbent of the DARP mixed-integer programming formulation, preserving solver-based feasibility verification and optimality certification. Computational experiments show that the predictor achieves a mean vehicle-aligned recall of 91.4%. Across ten evaluated instances, the prediction-centered parallel configuration obtains feasible solutions in every case and reduces mean time to the first feasible solution by 51.1% relative to cold start. The mean final optimality gap decreases from 5.6% to 3.8%, a relative reduction of 33.1%. These results demonstrate the potential of integrating problem-specific graph learning with confidence-guided primal search to accelerate feasible-solution discovery and improve optimization progress in demand-responsive routing.

1 INTRODUCTION

Demand-responsive transport adapts vehicle assignments and routes to passenger requests. Its routing decisions can be modeled as a dial-a-ride problem (DARP), in which each request must be picked up and delivered by the same vehicle subject to capacity, time windows, precedence, ride-time limits, and route duration (Cordeau, 2006; Cordeau & Laporte, 2007). These coupled requirements make both feasible-solution discovery and objective improvement challenging. Exact mixed-integer programming (MIP) solvers remain useful because they supply feasible incumbents and bounds with which to assess solution quality.

Repeated routing operations also generate related instances and historical solutions. A learned model can use this information to suggest promising arc assignments for a new instance. Such suggestions need not be correct individually, and thresholding them does not enforce the coupled DARP constraints. The relevant question is therefore how to use imperfect predictions to assist primal search while leaving feasibility and certification to the original optimization model.

Graph neural networks provide a representation of relational optimization structure (Gasse et al., 2019; Cappart et al., 2023). A variable–constraint graph preserves the nonzero pattern of an MIP, but does not explicitly distinguish passenger requests and vehicle resources. We augment this al-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

gebraic representation with typed entities and constraint-mediated semantic relations. The resulting heterogeneous GNN predicts arc scores rather than directly constructing certified routes.

The framework has three components. First, a heterogeneous graph combines arc variables, continuous-state variables, constraints, requests, and vehicles. Second, route-based score refinement handles ambiguous assignments among interchangeable vehicles. Third, confidence-weighted neighborhoods allow deviations from a shared prediction center and transfer only verified feasible candidates to the unrestricted solver. Our contributions are the DARP-specific heterogeneous representation, route-based score refinement, and confidence-guided primal-search framework, evaluated through prediction quality and optimization performance.

The evaluation distinguishes arc recovery from optimization performance. Refined predictions rank reference arcs more effectively, but lose some recall and remain infeasible after thresholding. Reported parallel-search runs improve first-feasible time and final gaps on a test instance collection.

2 Related work

Exact DARP optimization. Cordeau (2006) developed a three-index formulation and branch-and-cut algorithm. Stronger formulations and relaxations include branch-cut-and-price with dynamic time windows (Gschwind & Irnich, 2015), restricted-fragment formulations (Rist & Forbes, 2021), and event-based MIP models (Gaul et al., 2022). Our framework augments primal search for a given formulation and is complementary to improvements in mathematical formulations.

Primal heuristics. Local Branching restricts the Hamming distance from an incumbent (Fischetti & Lodi, 2003), while Relaxation Induced Neighborhood Search (RINS) combines incumbent and relaxation information (Danna et al., 2005). The Feasibility Pump alternates rounding and projection to seek an integer-feasible point (Fischetti et al., 2005). These methods motivate our use of neighborhoods and feasibility checks. Our weighted distance and asymmetric error allowances use learned arc preferences to differentiate changes to the prediction center.

Learning for MIP search. Learning can support branching (Gasse et al., 2019), predict variable assignments for primal search (Ding et al., 2020), or guide neighborhood selection (Song et al., 2020; Liu et al., 2022). Neural Diving predicts partial assignments completed by a solver (Nair et al., 2020). More broadly, learning-augmented combinatorial optimization must connect statistical predictions to discrete feasibility (Bengio et al., 2021). We study a typed DARP representation and the use of shared scores across complementary primal mechanisms. The learned scores guide primal heuristics while the solver manages branching and bounding.

3 Problem and optimization interface

Let $P = \{1, \dots, n\}$ and $D = \{n+1, \dots, 2n\}$ denote pickup and delivery nodes. The depot has departure and return copies 0 and $2n+1$, and request i pairs nodes i and $n+i$. A vehicle $k \in K$ has capacity Q_k and route-duration limit T_k. Node i has service duration s_i, load change q_i, and time window $[e_i, \ell_i]$. Each request has a maximum ride time. The allowed arc set excludes self-loops, arrivals at the departure depot, departures from the return depot, and a request's delivery-to-pickup arc.

The binary variable x_{ij}^k indicates whether vehicle k traverses (i, j). Continuous variables encode service-start times, loads after service, and passenger ride times. Let $\mathcal{F}$ be the feasible set of the original DARP formulation, including service, pairing, flow, timing, and resource constraints. We solve

$$\min_{(x,u,w,r)\in\mathcal{F}} f(x), \qquad f(x) = \sum_{k\in K} \sum_{(i,j)\in A} c_{ij}^k x_{ij}^k. \tag{1}$$

Appendix A presents the complete mathematical formulation. In the evaluated homogeneous setting every vehicle follows a nonempty route. Serving $2n$ request nodes with $|K|$ routes therefore selects $2n + |K|$ arcs, or 44 for the evaluated size. The candidate set has 6,400 vehicle-indexed variables.

In branch-and-bound, feasible solutions give upper bounds and relaxations give lower bounds. An early incumbent can improve pruning, but an auxiliary neighborhood's bound is not a global bound

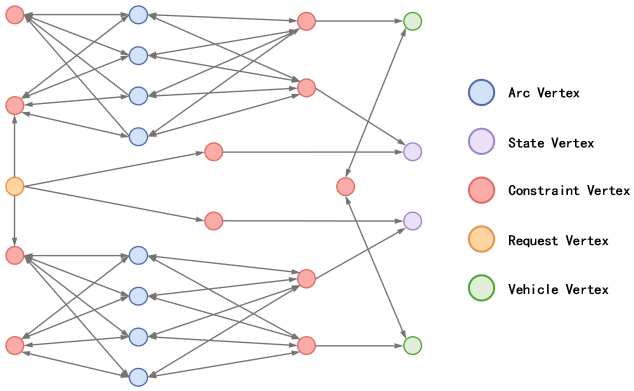

Figure 1: Heterogeneous graph architecture

for the unrestricted problem. We retain the original MIP as the authority for feasibility and certification. Learned hints do not change its feasible region; restricted neighborhoods are auxiliary models whose solutions are checked before transfer. A time-limited run provides only its recorded incumbent and gap, not an unconditional optimality guarantee.

4 Heterogeneous Prediction and Primal Guidance

4.1 Graph Representation and Learning

The heterogeneous graph combines arc variables V_A, continuous-state variables V_S, constraints V_C, requests V_R, and vehicles V_K. A nonzero coefficient A_{cv} defines a variable–constraint edge. Requests and vehicles connect to a constraint when their indices occur in its nonzero variables; semantic entity–constraint edges are deduplicated. Reverse relations support two-stage propagation from entities to constraints and back. Arc nodes represent vehicle-indexed variables, and state nodes represent individual time, load, or ride-time variables. Figure 1 shows this construction.

Independent encoders map the five node types to an H-dimensional latent space. For relation r at layer l, relation-specific messages are aggregated at each constraint:

$$m_c^{(l,r)} = \mathrm{AGG}_{(v,c)\in E_r}\, \phi_r^{(l)}(h_v^{(l-1)}, e_{vc}^r). \tag{2}$$

The four entity-to-constraint messages remain separate and jointly update the constraint embedding. Updated constraints then send messages to the entity nodes along reverse relations. After L layers, a shared readout yields $p_{ij}^k = \sigma(\mathrm{MLP}_o(h_{ijk}^{(L)}))$. Appendix E gives the update equations; features and training settings appear in Appendices D and C.

For each training instance, a solver-generated feasible solution supplies binary labels $y_a = x_a^*$. The loss combines class-weighted binary cross-entropy with an equality-residual penalty:

$$\mathcal{L} = -\frac{1}{|V_A^L|}\sum_{a\in V_A^L} \alpha_a[y_a \log p_a + (1-y_a)\log(1-p_a)] + \frac{\lambda_{\mathrm{str}}}{|C_{2:5}|}\sum_{c\in C_{2:5}}\left(\sum_{a\in V_A} A_{ca}p_a - b_c\right)^2. \tag{3}$$

Here V_A^L indexes supervised arc variables, class weights use training-label frequencies, and $C_{2:5}$ comprises arc-only service, depot, pairing, and flow equalities. The predicted scores subsequently guide search, whose subproblems enforce the DARP constraints.

4.2 Route-Based Score Refinement

Homogeneous vehicles admit equivalent assignments under relabeling. We map raw predictions to adjusted scores $\widetilde{p}$, a binary search center $\widehat{x}$, and confidence weights γ. Vehicle-invariant support is

$$s_{ij} = \max_{k\in K} p_{ij}^k, \qquad z_{ij} = \mathbf{1}\{s_{ij} \geq \tau\}, \qquad 0 < \tau < 1. \tag{4}$$

The maximum preserves the strongest evidence for each arc under permutations and avoids dilution as the fleet size grows. Pickup and delivery nodes are inserted jointly into routes. Candidate insertions prioritize supported arcs, aggregate support, and then shorter travel distance, using the lexicographic vector

$$Q(R) = \left(\sum_{(i,j)\in A(R)} z_{ij}, \sum_{(i,j)\in A(R)} s_{ij}, -\sum_{(i,j)\in A(R)} d_{ij} \right). \tag{5}$$

Candidate routes satisfy pairing, precedence, capacity, time-window, route-duration, and ride-time restrictions. Temporal feasibility is checked as a system of difference constraints. A limited number of feasible relocation and exchange moves improve the recovered route structure within the construction budget.

Recovered routes are canonically ordered by $(-N(R), \mathrm{sort}(A(R)))$, where $N(R)$ counts non-depot visits. This ordering is applied separately within groups of vehicles with identical attributes and operating constraints. For each arc define $P_{ij} = \{k : p_{ij}^k \geq \tau\}$. When $|P_{ij}| \geq 2$, route membership and then endpoint compatibility determine a target k_{ij}^*. If neither identifies a target, use a vehicle with the highest original score, breaking ties by vehicle index. The adjusted scores are

$$\widetilde{p}_{ij}^k = \begin{cases} s_{ij} + \eta(1 - s_{ij}), & k = k_{ij}^*, \\ (1-\xi)\tau p_{ij}^k / s_{ij}, & k \neq k_{ij}^*, \end{cases} \qquad \eta \in [0,1], \quad \xi \in (0,1]. \tag{6}$$

Arcs with fewer than two predicted-positive vehicles keep their scores. For adjusted groups, $\widetilde{p}_{ij}^{k_{ij}^*} \geq s_{ij} \geq \tau$ and $0 \leq \widetilde{p}_{ij}^k \leq (1-\xi)\tau < \tau$ for competing vehicles. Each duplicated-positive arc thus has a unique positive vehicle assignment, and its strongest support is retained or reinforced. These are search scores, rather than calibrated probabilities. Appendix F gives the complete transformation and confidence-screening equations.

4.3 Confidence and error-tolerant search

Let q_a be the scores supplied to search: $\widetilde{p}_a$ for the refined variant or p_a for raw-GNN guidance. Define $\widehat{x}_a = \mathbf{1}\{q_a \geq \tau\}$. For $0 < \theta \leq 1$ and $0 < \tau < 1$, confidence is

$$\gamma_a = c_{\theta,\tau}(q_a), \qquad c_{\theta,\tau}(u) = \theta + (1-\theta) \begin{cases} ((\tau - u)/\tau)^2, & u < \tau, \\ ((u-\tau)/(1-\tau))^2, & u \geq \tau. \end{cases} \tag{7}$$

Confidence ranges from θ at the threshold to one at the endpoints. It measures decisiveness, not calibrated correctness. The zero derivative at τ reduces sensitivity to small score perturbations near the boundary.

Hints and prediction-centered neighborhoods. Variable hints supply $\widehat{x}_a$ and priorities ordered by γ_a without fixing values. For a confidence cutoff α, define $Z = \{a : \widehat{x}_a = 0, \gamma_a \geq \alpha\}$ and $O = \{a : \widehat{x}_a = 1, \gamma_a \geq \alpha\}$. An auxiliary MIP retains $\mathcal{F}$ and adds

$$\sum_{a\in Z} x_a \leq B_0, \qquad \sum_{a\in O} (1 - x_a) \leq B_1. \tag{8}$$

Separate cutoffs may be used for the two classes. The budgets allow asymmetric errors rather than unconditional fixing. A second auxiliary MIP uses a weighted Local Branching neighborhood:

$$\Delta_\gamma(x, \widehat{x}) = \sum_{a:\widehat{x}_a = 0} \gamma_a x_a + \sum_{a:\widehat{x}_a = 1} \gamma_a (1 - x_a) \leq \kappa. \tag{9}$$

For binary vectors, $\theta d_H(x, \widehat{x}) \leq \Delta_\gamma(x, \widehat{x}) \leq d_H(x, \widehat{x})$. Consequently, the weighted ball contains the ordinary Hamming ball of radius κ and is contained in the ball of radius κ/θ. At $\theta = 1$ this reduces to ordinary Local Branching. The parameters κ and θ therefore control the neighborhood radius and its maximum Hamming reach.

Incumbent-guided neighborhoods. Once a feasible incumbent x^I is available, an agreement neighborhood limits changes where $\widehat{x}_a = x^I_a$ and confidence is high. A weighted incumbent-centered alternative uses

$$\omega_a = \begin{cases} (1+\gamma_a)/2, & x^I_a = \widehat{x}_a, \\ (\theta+\gamma_a)/2, & x^I_a \neq \widehat{x}_a, \end{cases} \qquad \Delta_\omega(x, x^I) \leq \kappa_I. \tag{10}$$

For a fixed confidence, agreement is more expensive to change. Appendix G.1 describes a prediction-initialized Feasibility Pump as an additional search mechanism.

Solver integration. Prediction-centered subproblems search model copies. The incumbent-guided configuration additionally runs the unrestricted model and invokes incumbent-centered searches once, on its first incumbent. Auxiliary searches share a deadline and retain their original reference centers. Verified candidates are compared using $f(x)$ and the best available candidate is transferred to the unrestricted MIP. Neighborhood restrictions are not retained in the final model. The unrestricted original MIP then continues with the best available incumbent and the remaining time budget.

5 Experiments

5.1 Experimental setup

The dataset contains 51 synthetic instances with 20 requests and four homogeneous vehicles, partitioned into 33 training, 8 validation, and 10 test instances using seed 0. All graph and solution records for an instance remain in one subset. Preliminary MIP solves provide feasible solutions for binary supervision. The generation procedure and parameters are given in Appendix B.

The HGNN uses four message-passing layers and hidden dimension 256. We restore the checkpoint with the lowest validation loss. Confidence cutoffs and neighborhood budgets are selected using validation data and fixed before testing. The operating thresholds are 0.57 for GNN only and Proposed, and 0.50 for Root LP. Training settings appear in Appendix C.

Each test instance receives a 3,600 s wall-clock budget, including feature extraction, inference, projection, auxiliary search, and continued optimization. Where applicable, the auxiliary phase is limited to 600 s within that budget. Prediction-centered searches start after root-node processing and prediction preparation; incumbent-guided searches are triggered once by the original solver's first feasible incumbent. Later-triggered searches receive the remaining auxiliary time. At the auxiliary deadline, candidates and the original incumbent are compared under the DARP objective. The unrestricted solver continues for the remaining total time. Cold start uses the same formulation, total limit, and resource allocation. Global bounds and gaps come from the unrestricted model; offline generation and training costs are separate.

Experiments use Python, PyTorch 2.8.0 with CUDA 12.9 support, and Gurobi 12.0.3 on Windows with an AMD Ryzen Threadripper PRO 5995wX (64 cores, 2.70 GHz), 256 GB RAM, and an NVIDIA GeForce RTX 4090 (47.5 GB VRAM).

5.2 Prediction quality

Proposed denotes the complete prediction scheme; GNN only removes vehicle symmetry elimination; Root LP directly uses the root-node relaxation values. The first comparison assesses the symmetry treatment, and the second assesses the complete scheme relative to relaxation information. Each of the ten test instances has 6,400 vehicle-indexed variables. The lowest-objective feasible pool solution defines a binary reference with 44 selected arcs (0.6875%). Metrics measure recovery of that reference.

For each instance and source, one global vehicle permutation maximizes overlap at the operating threshold and is retained for ranking evaluations. This evaluation relabeling is distinct from changing scores through symmetry elimination. Precision, recall, F_1, and average precision (AP) are unweighted instance means; predicted-positive and true-positive counts are pooled. AP summarizes precision–recall performance across thresholds.

Table 1: Prediction performance

Method	Positives	TP	Precision (%)	Recall (%)	F_1 (%)	AP
Root LP	463	141	36.38	32.05	31.71	.2039
GNN only	7,180	402	5.60	91.36	10.56	.1264
Proposed	1,909	361	18.90	82.05	30.71	.4141

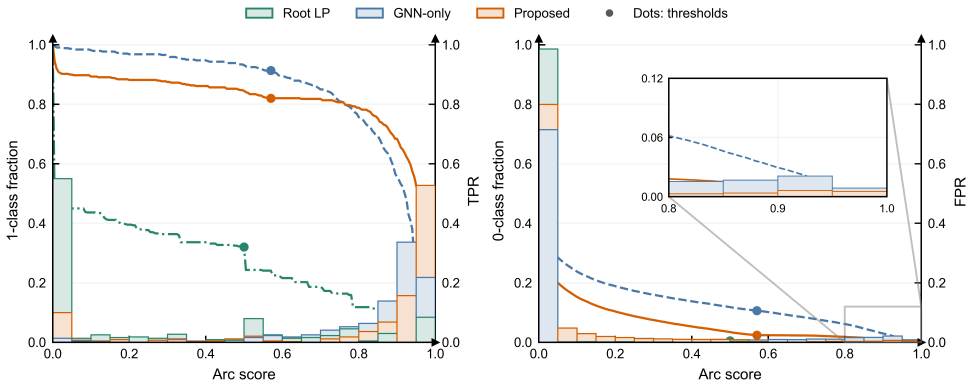

Figure 2: Arc-score distributions

Table 1 shows that symmetry elimination reduces positive predictions by 73.4% (7,180 to 1,909), while matched reference arcs decrease from 402 to 361. Precision rises from 5.60% to 18.90%, and AP from 0.1264 to 0.4141, at the cost of recall declining from 91.36% to 82.05%. Root LP has higher precision and slightly higher F_1 than Proposed, but lower recall and AP.

Figure 2 separates score distributions for reference-positive and reference-negative arcs. Histograms show class fractions; curves on the right axes give TPR and FPR as thresholds vary, with operating thresholds marked by dots. The inset enlarges the high-score negative region. Proposed reduces high scores on negative arcs, while assigning some positive arcs low scores. Root LP concentrates most negatives near zero, but also gives near-zero scores to many positives. These patterns explain the threshold-level trade-offs.

For a candidate budget k, reference-arc coverage is

$$C(k) = \frac{|\operatorname{Top}_k(p) \cap Y^+|}{|Y^+|}, \tag{11}$$

where Y^+ is the reference positive set and $\operatorname{Top}_k(p)$ contains the highest-scoring variables. A budget cutting tied scores uses expected coverage under uniform selection. Figure 3 uses normalized budget $k/|Y^+|$: curves are means, bands are instance interquartile ranges, and markers are extrema at budgets 1, 2, and 4.

With 44, 88, and 176 candidates, Proposed covers 45.91%, 64.55%, and 81.36% of reference arcs. GNN only covers 14.77%, 29.77%, and 52.05%; Root LP covers 30.38%, 39.76%, and 45.30%. Proposed reaches 80% mean coverage at normalized budget 3.68, about 162 candidates. Neither baseline reaches this level within the displayed range of 352 candidates. At 176 candidates, Proposed's instance coverage ranges from 50.00% to 93.18%.

Figure 4 shows paired instance changes: recall on the horizontal axis, AP on the vertical axis, and precision in the shared color scale. Changes are percentage points (AP multiplied by 100); diamonds are means and marginal ticks are instance values. Relative to GNN only, precision and AP improve on all ten instances, by means of 13.29 and 28.77 points. Recall falls on seven instances and is unchanged on three, with a mean loss of 9.32 points. The largest losses are 29.55, 25.00, and 18.18 points on Hard17, Hard8, and Hard55. Relative to Root LP, recall and AP improve on every instance, by means of 50.00 and 21.02 points; precision falls on nine and rises on one, with a mean decrease of 17.48 points. Improved ranking coexists with omissions, motivating neighborhoods that permit prediction errors.

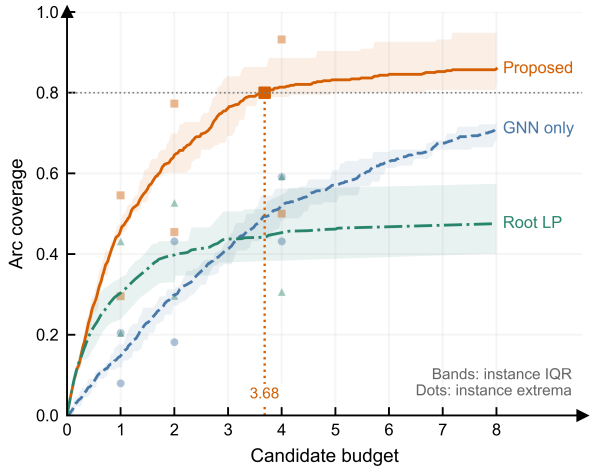

Figure 3: Reference-arc coverage

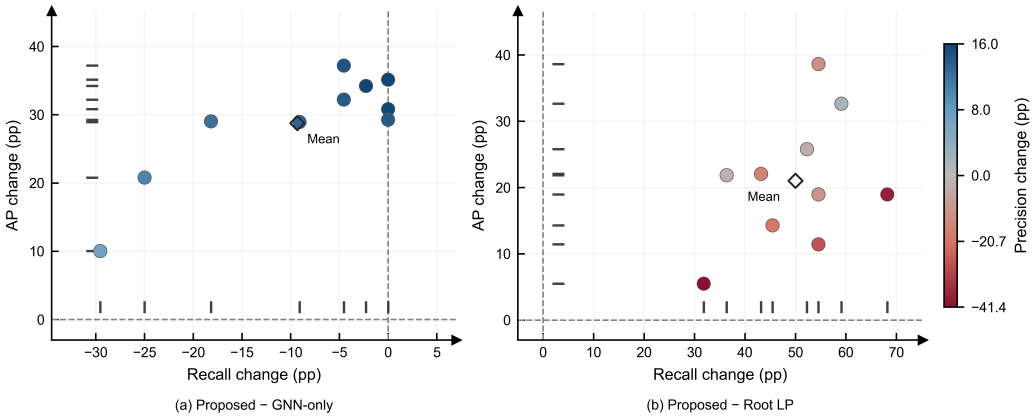

Figure 4: Instance-level prediction trade-offs

5.3 Optimization performance

Table 2 compares cold start, hints, prediction-centered parallel search, and incumbent-guided parallel search on the same ten instances. Each configuration is evaluated independently under the same 3,600 s total limit. Objective and gap means use available values, first-feasible means use available attainment times, and runtime means include all ten runs. Paired objective comparisons use common successful subsets.

Parallel search and cold start both have feasible objectives for all ten instances. Their mean recorded first-feasible times differ by 371.3 seconds, a 51.1% reduction. Mean final gap decreases by 1.854 percentage points (33.1% relative). Figure 5 shows cumulative first-feasible attainment; all ten runs remain in the denominator.

The largest objective improvement is on Hard17: 975.982 to 939.873 (3.70%). On Hard8, cold start is better: 907.904 versus 922.383. The remaining eight paired objectives agree to two decimal places. The mean of the ten per-instance relative improvements is 0.210%; it differs from the percentage change in the aggregate mean objective. Thus the main observed benefits concern first-feasible time and gaps, rather than broad objective improvement.

Parallel search has a lower displayed final gap on nine instances and the same displayed gap on Hard15. Hard17 decreases from 12.28% to 8.45%, and Hard47 from 10.80% to 6.81%. On Hard10 the objective is approximately 999.80 in both configurations, while the gap changes from 4.81% to 2.97%, consistent with a tighter final bound. Gap plots are provided in Appendix H.

Table 2: Optimization results. Objective/gap means use available values; first-feasible means use available attainment times. Runtime averages use all ten records. "Parallel" denotes prediction-centered parallel search.

Configuration	Feasible records	Mean objective	Mean gap (%)	Runtime (s)	First feasible (s)
Cold start	10/10	901.010	5.608	3,548.5	726.2
Hint	9/10	900.016	3.553	3,042.4	444.1
Parallel	10/10	898.847	3.754	3,156.7	354.9
Incumbent parallel	9/10	894.288	3.892	3,283.1	468.6

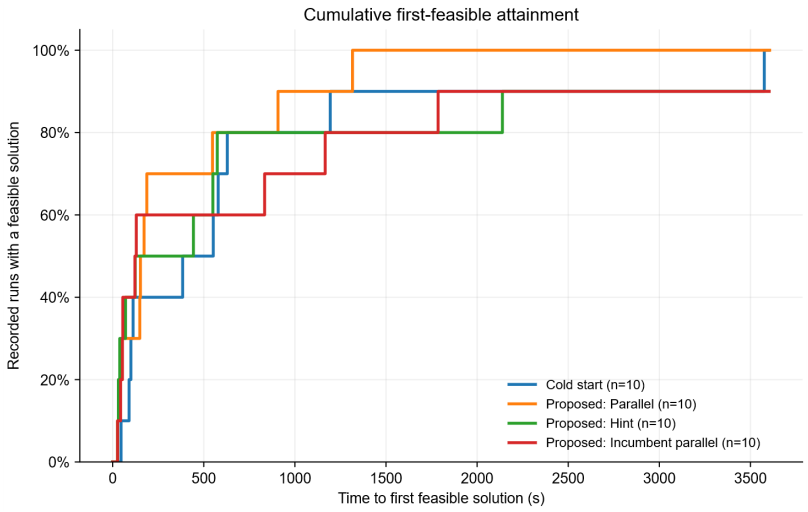

Figure 5: First-feasible-solution attainment

Hint lacks an available feasible objective on Hard21; incumbent parallel lacks one on Hard17. Their mean paired objective improvements on the respective common successful subsets are 0.160% and -0.177%. The comparison therefore reports both feasible-solution coverage and paired objective changes.

6 Limitations and Conclusion

The framework connects a typed MIP representation to error-tolerant primal search while keeping feasibility checks and global certification in the unrestricted solver. The prediction results show improved ranking after refinement, with a measurable recall cost. Separately, the recorded parallel-search results show earlier feasible-solution discovery and smaller final gaps on ten instances.

Further validation on broader and more diverse datasets, spanning different request volumes, fleet configurations, and demand patterns, is needed to assess the generality of the framework.

AI USE STATEMENT

Generative AI (OpenAI Codex) was used for LaTeX preparation, text organization and language editing, and mathematical notation transcription.

REFERENCES

Yoshua Bengio, Andrea Lodi, and Antoine Prouvost. Machine learning for combinatorial optimization: A methodological tour d'horizon. *European Journal of Operational Research*, 290(2): 405–421, 2021.

Quentin Cappart, Didier Chételat, Elias B. Khalil, Andrea Lodi, Christopher Morris, and Petar Veličković. Combinatorial optimization and reasoning with graph neural networks. *Journal of Machine Learning Research*, 24(130):1–61, 2023.

Jean-François Cordeau. A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research*, 54(3):573–586, 2006.

Jean-François Cordeau and Gilbert Laporte. The dial-a-ride problem: Models and algorithms. *Annals of Operations Research*, 153:29–46, 2007.

Emilie Danna, Edward Rothberg, and Claude Le Pape. Exploring relaxation induced neighborhoods to improve MIP solutions. *Mathematical Programming*, 102:71–90, 2005.

Jian-Ya Ding, Chao Zhang, Lei Shen, Shengyin Li, Bing Wang, Yinghui Xu, and Le Song. Accelerating primal solution findings for mixed integer programs based on solution prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 1452–1459, 2020.

Matteo Fischetti and Andrea Lodi. Local branching. *Mathematical Programming*, 98:23–47, 2003.

Matteo Fischetti, Fred Glover, and Andrea Lodi. The feasibility pump. *Mathematical Programming*, 104:91–104, 2005.

Maxime Gasse, Didier Chételat, Nicola Ferroni, Laurent Charlin, and Andrea Lodi. Exact combinatorial optimization with graph convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Daniela Gaul, Kathrin Klamroth, and Michael Stiglmayr. Event-based MILP models for ridepooling applications. *European Journal of Operational Research*, 301(3):1048–1063, 2022.

Timo Gschwind and Stefan Irnich. Effective handling of dynamic time windows and its application to solving the dial-a-ride problem. *Transportation Science*, 49(2):335–354, 2015.

Defeng Liu, Matteo Fischetti, and Andrea Lodi. Learning to search in local branching. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 3796–3803, 2022.

Vinod Nair, Sergey Bartunov, Félix Gimeno, Ingrid von Glehn, Pawel Lichocki, Ivan Lobov, Brendan O'Donoghue, et al. Solving mixed integer programs using neural networks. arXiv:2012.13349, 2020.

Yannik Rist and Michael A. Forbes. A new formulation for the dial-a-ride problem. *Transportation Science*, 55(5):1113–1135, 2021.

Jialin Song, Ravi Lanka, Yisong Yue, and Bistra Dilkina. A general large neighborhood search framework for solving integer linear programs. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

A DARP FORMULATION AND NOTATION

The formulation uses s_i consistently for service duration and d_{ij} for travel distance. Let $o = 0$, $d = 2n + 1$, and $N = P \cup D$. The arc set is

$$A = (\{o\} \times P) \cup \{(i, j) \in N^2 : i \neq j,\ i \neq n + j\} \cup (D \times \{d\}). \tag{12}$$

For each request $i \in P$, $q_i > 0$ and $q_{n+i} = -q_i$; depot loads and service durations are zero. Define $\delta^+(i) = \{j : (i, j) \in A\}$ and $\delta^-(i) = \{j : (j, i) \in A\}$. The original objective is Equation 1. Its service and flow constraints are

$$\sum_{k \in K} \sum_{j \in \delta^+(i)} x_{ij}^k = 1 \qquad i \in P, \tag{13}$$

$$\sum_{j \in \delta^+(o)} x_{oj}^k = \sum_{i \in \delta^-(d)} x_{id}^k = 1 \qquad k \in K, \tag{14}$$

$$\sum_{j \in \delta^+(i)} x_{ij}^k - \sum_{j \in \delta^+(n+i)} x_{n+i,j}^k = 0 \qquad i \in P,\ k \in K, \tag{15}$$

$$\sum_{j \in \delta^+(i)} x_{ij}^k - \sum_{j \in \delta^-(i)} x_{ji}^k = 0 \qquad i \in N,\ k \in K. \tag{16}$$

These are the C2–C5 constraint families used in the structural loss. The continuous-resource constraints are

$$u_j^k \geq u_i^k + s_i + t_{ij} - M^u(1 - x_{ij}^k) \qquad (i, j) \in A,\ k \in K, \tag{17}$$

$$w_j^k \geq w_i^k + q_j - M^w(1 - x_{ij}^k) \qquad (i, j) \in A,\ k \in K, \tag{18}$$

$$w_j^k \leq w_i^k + q_j + M^w(1 - x_{ij}^k) \qquad (i, j) \in A,\ k \in K, \tag{19}$$

$$r_i^k \geq u_{n+i}^k - u_i^k + s_i \qquad i \in P,\ k \in K, \tag{20}$$

$$u_d^k - u_o^k \leq T_k \qquad k \in K, \tag{21}$$

$$e_i \leq u_i^k \leq \ell_i \qquad i \in V,\ k \in K, \tag{22}$$

$$t_{i,n+i} \leq r_i^k \leq L^{\max} \qquad i \in P,\ k \in K, \tag{23}$$

$$\max(0, q_i) \leq w_i^k \leq \min(Q_k, Q_k + q_i) \qquad i \in V,\ k \in K, \tag{24}$$

$$w_o^k = w_d^k = 0 \qquad k \in K, \tag{25}$$

$$x_{ij}^k \in \{0, 1\} \qquad (i, j) \in A,\ k \in K. \tag{26}$$

The constants M^u, M^w relax inactive arcs over the declared variable bounds.

B INSTANCE GENERATION PARAMETERS

Instance generation uses a grid over a square region and a central depot. Pickup–delivery pairing favors short trips, with Euclidean distance determining travel time at speed one. Demand is obtained by adding a normal and a uniform random variable, rounding, and truncating. Time windows are built around service schedules and then perturbed. Coordinates and time are expressed in model units.

Endpoint reuse and partial temporal dependence between successive generated candidates are allowed. The reuse-mode probabilities are 0.10 (both endpoints), 0.20 (pickup), 0.20 (delivery), and 0.50 (neither), with fallback when reuse is unavailable. These are generation settings, not measured reuse frequencies. Time-window centers for reused endpoints can mix new and previous centers. The dataset contains 51 instances.

C LEARNING DETAILS

A solver-generated feasible solution provides binary labels $y_a = x_a^*$. Selected arcs receive label one and all other arcs receive label zero.

Table 3: Instance generation parameters.

Parameter	Value
Requests / homogeneous vehicles	20 / 4
Region / grid spacing	$[0, 100] \times [0, 100]$ / 5
Initial location pool / depot	40 / $(50, 50)$
Pickup–delivery sampling scale	8
Vehicle capacity / speed	20 / 1
Maximum route duration / depot window	750 / $[0, 750]$
Maximum ride time / node service time	100 / 5
Demand components	$\mathcal{N}(4, 2^2) + \mathrm{Uniform}(0, 2)$
Integer demand range	1–8
Nominal window width / width-noise SD	100 / 10
Local center-noise SD	$0.15w_i$, for sampled width w_i

Let $N_+ = \sum_a y_a$ and $N_- = \sum_a (1 - y_a)$ across training arcs. The inverse-frequency class weights are

$$w_+ = \frac{N_+ + N_-}{2N_+}, \qquad w_- = \frac{N_+ + N_-}{2N_-}, \tag{27}$$

with empty-class masses lower-bounded by one. The implementation uses logits-based BCE for numerical stability. Equation 3 is its probability-space expression.

Table 4: GNN training configuration.

Setting	Value
Hidden dimension / message-passing layers	256 / 4
Dropout / maximum epochs / batch size	0.25 / 80 / 1
Optimizer / learning rate	AdamW / 5×10^{-4}
Weight decay / structural-loss weight	10^{-4} / 10^{-3}
Gradient-norm limit	5.0
Validation interval	Every 2 training epochs
Early-stopping patience	12 training epochs
Minimum validation improvement	10^{-4}

The checkpoint with the lowest validation loss is restored for evaluation. CUDA automatic mixed precision and dynamic gradient scaling are used during training.

D Feature and relation specification

Arc and state variables share 40 algebraic/root-LP features. For variable v, let $C(v) = \{c : A_{cv} \neq 0\}$ and let d_c be the row nonzero count. The row-level features include LP activity ℓ_c^{LP}, a right-hand LP quantity b_c^{LP}, and row dual π_c. The notation b_c denotes the original right-hand-side constant used in the structural loss.

Standard deviations are computed as population statistics. Empty groups, missing/non-finite values, and ratios with numerically zero denominators are set to zero. Let f_v^{common} denote these 40 features. Arc variables add objective coefficient c_v, $|c_v|$, $-|c_v|$, a bound-fixed indicator, and a root-LP-integrality indicator (45 total). State variables add model lower/upper bounds, a three-dimensional one-hot time/load/ride-time type, and root global lower/upper bounds (47 total). Infinite bounds are serialized as such but mapped to zero at model input.

Constraint nodes use the following 13-dimensional vector:

$$f_c^C = \left[n_c^{\mathrm{nz}}, n_c^+, n_c^-, \ell_c^{\mathrm{LP}}, b_c^{\mathrm{LP}}, \pi_c, \sum_v |A_{cv}|, \sum_{A_{cv}>0} A_{cv}, \sum_{A_{cv}<0} A_{cv}, \mu_c, \sigma_c, A_c^{\min}, A_c^{\max} \right]. \tag{28}$$

Table 5: Shared variable features.

Group	Description	Dim.
Degree	$\lvert C(v)\rvert$	1
Root LP	LP value, distances to adjacent integers, reduced cost	4
Adjacent row size	Mean, SD, minimum, maximum of d_c	4
Activity ratios	Min/max of $\ell_c^{\mathrm{LP}}/b_c^{\mathrm{LP}}$, grouped by coefficient sign	4
Signed coefficients	Count, min, max, SD, mean, sum for each sign	12
Coefficient sequences	Sum, mean, SD, min, max for each of A_{cv}, $A_{cv}\pi_c$, and row-normalized coefficients	15
Total		40

Here the first three entries count all, positive, and negative coefficients, and the last four are their mean, population SD, minimum, and maximum. The representation excludes constraint-family one-hot encodings, constraint direction, and the original RHS as direct neural inputs; row families and RHS remain available to the loss.

For request q with pickup p_q and delivery d_q, the nine features are

$$f_q^R = [e_{p_q}, \ell_{p_q}, e_{d_q}, \ell_{d_q}, q_{p_q}, q_{d_q}, s_{p_q}, s_{d_q}, t_{p_q d_q}^{\min}]. \tag{29}$$

Both signed demand changes and both service durations are retained. Vehicle features are $[Q_k, T_k^{\max}, L_k^{\max}]$. Vehicle ID is not a numerical input. In particular, duplicate vehicle attributes alone do not break homogeneous-vehicle symmetry.

Each type-specific input encoder has the form

$$h_v^0 = \mathrm{LN}_H(W_{\tau,2}\,\mathrm{Dropout}(\mathrm{GELU}(W_{\tau,1}\,\mathrm{LN}_{d_\tau}(f_v) + b_{\tau,1})) + b_{\tau,2})\,. \tag{30}$$

This and the scalar arc-output head use independent learnable maps; the latter applies GELU and dropout between two affine transformations.

For a coefficient edge (v, c), the four features are

$$e_{vc} = \left[A_{cv},\ \mathrm{sgn}(A_{cv}),\ \log(1 + |A_{cv}|),\ \frac{A_{cv}}{\sum_{v'} |A_{cv'}|}\right]. \tag{31}$$

Semantic edges carry scalar one. Reverse edges reuse these attributes but have independent learnable maps. Raw inputs use type-specific LayerNorm rather than per-instance column normalization. Constraint updates have input dimension $5H$, hidden dimension $2H$, and output dimension H; entity updates have input dimension $2H$ and hidden/output dimension H.

E Heterogeneous prediction details

E.1 Heterogeneous graph representation

For a DARP graph, construct a heterogeneous graph $\mathcal{G}^H = (\mathcal{V}, \mathcal{E}, \mathcal{R})$.The vertex set is $\mathcal{V} = \mathcal{V}_A \cup \mathcal{V}_S \cup \mathcal{V}_C \cup \mathcal{V}_R \cup \mathcal{V}_K$, where $\mathcal{V}_A$ represents arc variable nodes corresponding to the binary variables x_{ij}^k; $\mathcal{V}_S$ represents state variable nodes, including service time u_i^k, vehicle load w_i^k, and ride time r_i^k; $\mathcal{V}_C$ represents MIP constraint nodes; $\mathcal{V}_R$ represents request nodes, used to represent pickup-and-delivery pairs and request attributes; and $\mathcal{V}_K$ represents vehicle nodes, used to represent vehicle resource attributes.

The edge connection type in the graph is defined as$\mathcal{R} = \{A-C, S-C, R-C, K-C\}$. Among these, $A-C$ and $S-C$ directly inherit the non-zero structure of the MIP constraint matrix. Let A_{cv} denote the coefficient of variable v in constraint c; then:

$$\mathcal{E}_{AC} = \{(v, c) : v \in \mathcal{V}_A, A_{cv} \neq 0\}, \mathcal{E}_{SC} = \{(v, c) : v \in \mathcal{V}_S, A_{cv} \neq 0\}. \tag{32}$$

The $R-C$ and $K-C$ relationships are automatically constructed based on the indices of variables actually present in the constraints: if the index of a specific request or vehicle appears in any

non-zero variable within constraint c, the corresponding entity is linked to that constraint. Only a single semantic edge is retained for any given entity–constraint pair. Reverse message-passing edges are established for all relationships, thereby enabling bidirectional information exchange between entities and constraints.

This representation preserves the complete MIP incidence structure while allowing request attributes and vehicle resources to influence the arc variable representation through shared constraints. Specific construction rules for node features, edge attributes, and semantic edges are provided in Appendix D.

E.2 Constraint-mediated message passing

Different types of nodes are first mapped to a unified H-dimensional latent space via independent encoders. Subsequently, each message-passing layer executes two phases:

$$\{\mathcal{V}_A, \mathcal{V}_S, \mathcal{V}_R, \mathcal{V}_K\} \to \mathcal{V}_C \to \{\mathcal{V}_A, \mathcal{V}_S, \mathcal{V}_R, \mathcal{V}_K\}. \tag{33}$$

For a relation r, source node representations and edge attributes are first transformed via a relation-specific message function and aggregated at the target constraint node:

$$\mathbf{m}_c^{(l,r)} = \mathrm{AGG}_{(v,c)\in\mathcal{E}_r}\, \phi_r^{(l)}\left(\mathbf{h}_v^{(l-1)}, \mathbf{e}_{vc}^{r}\right). \tag{34}$$

Messages from the four entity types to the constraint remain separate during the aggregation phase and are used—alongside the constraint node's original representation—to update the constraint embedding:

$$\mathbf{h}_c^{(l)} = \mathrm{UPDATE}_C^{(l)}\left(\mathbf{h}_c^{(l-1)}, \mathbf{m}_{c,A}^{(l)}, \mathbf{m}_{c,S}^{(l)}, \mathbf{m}_{c,R}^{(l)}, \mathbf{m}_{c,K}^{(l)}\right). \tag{35}$$

The updated constraint representation is then propagated back to the various entity nodes along the inverse relations:

$$\mathbf{h}_v^{(l)} = \mathrm{UPDATE}_{\tau(v)}^{(l)}\left(\mathbf{h}_v^{(l-1)}, \overline{\mathbf{m}}_{v,C}^{(l,\tau(v))}\right). \tag{36}$$

After L layers of propagation, arc variable nodes can synthesize information regarding candidate arc competition relationships, continuous state feasibility, request attributes, and vehicle resources through paths such as $A \to C \to A$, $S \to C \to A$, $R \to C \to A$, and $K \to C \to A$. Details regarding the specific node encoders, edge gating functions, relation-specific mappings, residual updates, and network dimensions employed in this paper are provided in Appendix C.

E.3 Arc prediction and learning objective

After L layers of message passing, the shared readout network maps the arc variable nodes to scalars and applies a Sigmoid function to obtain the arc selection probabilities:

$$p_{ij}^k = \sigma\left(\mathrm{MLP}_o\left(\mathbf{h}_{ijk}^{(L)}\right)\right) = \Pr\left(x_{ij}^k = 1 \mid \mathcal{G}^H\right). \tag{37}$$

For each training instance, a feasible solution x^* generated by a solver provides binary supervision labels for the arc variables, i.e.,

$$y_{ijk} = x_{ijk}^* \in \{0, 1\}. \tag{38}$$

Since the number of selected arcs in a feasible path is far smaller than that of unselected arcs, class-weighted binary cross-entropy is employed for training. Let $\mathcal{V}_A^L$ denote the set of arc variable nodes with supervision labels; then,

$$\mathcal{L}_{\text{pred}} = -\frac{1}{|\mathcal{V}_A^L|} \sum_{a \in \mathcal{V}_A^L} \alpha_a \left[y_a \log p_a + (1 - y_a) \log(1 - p_a) \right], \tag{39}$$

where the class weights α_a are calculated based on the frequencies of positive and negative samples in the training set (see Appendix C for the specific definition).

In addition to arc-wise supervision, this paper further utilizes structural information from the DARP model to constrain the prediction results. For the equality constraints C2–C5 involving only arc variables, the binary variables are replaced by predicted probabilities, yielding the predicted activity value:

$$\hat{a}_c = \sum_{v \in \mathcal{V}_A} A_{cv} p_v. \tag{40}$$

Let $\mathcal{C}_{2:5}$ denote the set of these constraints and b_c their original right-hand side values; the structural consistency loss is then defined as

$$\mathcal{L}_{\text{str}} = \frac{1}{|\mathcal{C}_{2:5}|} \sum_{c \in \mathcal{C}_{2:5}} (\hat{a}_c - b_c)^2. \tag{41}$$

The final training objective is

$$\mathcal{L} = \mathcal{L}_{\text{pred}} + \lambda_{\text{str}} \mathcal{L}_{\text{str}}. \tag{42}$$

The resulting arc probabilities p_{ij}^k do not directly fix the MIP variables; instead, they undergo further path-structure-based post-processing (described in Section 4.2) and are converted into adjusted arc scores, binary search centers, and confidence weights for use in the prediction-guided search detailed in Section 4.3.

F Route recovery and score concentration

For homogeneous vehicles, equivalent route sets may differ only by vehicle labels, which can make the vehicle-indexed probabilities p_{ijk} inconsistent across symmetric assignments. We therefore transform the raw predictions into route-adjusted search guidance,

$$\mathcal{T} : p \to (\tilde{p}, \hat{x}, \gamma), \tag{43}$$

where $\tilde{p}$ denotes adjusted arc scores, $\hat{x}$ is the binary search center, and γ contains margin-based confidence weights.

F.1 Vehicle-invariant aggregation and route recovery

Let $p_{ijk} \in [0, 1]$ be the predicted probability for arc (i, j) and vehicle $k \in \mathcal{K}$, and let $\tau \in (0, 1)$ be the classification threshold. Define the vehicle-invariant arc support as

$$s_{ij} = \max_{k \in \mathcal{K}} p_{ijk}, \quad z_{ij} = \mathbb{I}\{s_{ij} \geq \tau\}. \tag{44}$$

The maximum preserves the strongest arc evidence under vehicle-label permutations and avoids dilution as $|\mathcal{K}|$ increases.

For a route R, define

$$Q(R) = \left(\sum_{(i,j) \in \mathcal{A}(R)} z_{ij}, \sum_{(i,j) \in \mathcal{A}(R)} s_{ij}, - \sum_{(i,j) \in \mathcal{A}(R)} d_{ij} \right), \tag{45}$$

where $\mathcal{A}(R)$ is the arc set of R and d_{ij} is the travel distance. Route construction therefore prioritizes supported arcs, aggregate prediction support, and travel distance in lexicographic order.

Pickup and delivery nodes are inserted jointly. Candidate routes satisfy pairing, precedence, capacity, time-window, route-duration, and ride-time restrictions. For a fixed route $R = (v_0, \dots, v_h)$,

$$T_{v_{r+1}} \geq T_{v_r} + s_{v_r} + t_{v_r v_{r+1}}, \quad r = 0, \dots, h-1, \tag{46}$$

$$e_{v_r} \leq T_{v_r} \leq \ell_{v_r}, \quad r = 0, \dots, h, \tag{47}$$

and

$$T_{v_h} - T_{v_0} \leq T_{\max}, \quad T_{d_i} - T_{p_i} - s_{p_i} \leq L_i. \tag{48}$$

The temporal restrictions are checked as a system of difference constraints. A limited number of feasible relocation and exchange moves are used to improve the recovered route structure within the construction budget.

F.2 Canonical vehicle assignment

Recovered routes are mapped to canonical vehicle indices by

$$\chi(R) = \left(-N(R), \mathrm{sort}(\mathcal{A}(R))\right), \tag{49}$$

where $N(R)$ is the number of visited non-depot nodes. Routes are ordered first by decreasing $N(R)$ and then lexicographically by their sorted arc lists. Canonicalization is performed separately within groups of vehicles sharing identical attributes and operating constraints.

For each arc, define

$$\mathcal{P}_{ij} = \{k \in \mathcal{K} : p_{ijk} \geq \tau\}. \tag{50}$$

When $|\mathcal{P}_{ij}| \geq 2$, a target vehicle k^*_{ij} is determined from the canonical route structure. Direct route membership has priority, followed by endpoint compatibility with the recovered routes. If the route structure does not identify a target vehicle,

$$k^*_{ij} \in \arg\max_{k \in \mathcal{K}} p_{ijk}, \tag{51}$$

with ties broken by vehicle index.

F.3 Route-guided score concentration

For arcs with $|\mathcal{P}_{ij}| \geq 2$, define

$$\tilde{p}_{ijk} = \begin{cases} s_{ij} + \eta(1 - s_{ij}), & k = k^*_{ij}, \\ (1-\xi)\tau \dfrac{p_{ijk}}{s_{ij}}, & k \neq k^*_{ij}, \end{cases} \tag{52}$$

where $\eta \in [0, 1]$ controls reinforcement of the selected vehicle and $\xi \in (0, 1]$ controls suppression of competing assignments. Arcs with fewer than two predicted-positive vehicles retain their original scores.

For every adjusted arc group,

$$\tilde{p}_{ij,k^*_{ij}} \geq s_{ij} \geq \tau, \tag{53}$$

whereas

$$0 \le \tilde{p}_{ijk} \le (1-\xi)\tau < \tau, \quad k \ne k^*_{ij}. \tag{54}$$

Hence, each duplicated-positive arc is mapped to a unique positive vehicle assignment while preserving its strongest vehicle-invariant support. The transformed quantities $\tilde{p}_{ijk}$ are treated as search scores rather than calibrated probabilities.

F.4 Binary search center and margin-based confidence

The binary search center is

$$\hat{x}_{ijk} = \mathbb{I}\{\tilde{p}_{ijk} \ge \tau\}. \tag{55}$$

For a minimum weight $\theta \in (0, 1]$, define

$$c_{\theta,\tau}(u) = \begin{cases} \theta + (1-\theta)\left(\dfrac{\tau - u}{\tau}\right)^2, & 0 \le u < \tau, \\ \theta + (1-\theta)\left(\dfrac{u - \tau}{1-\tau}\right)^2, & \tau \le u \le 1, \end{cases} \tag{56}$$

and

$$\gamma_{ijk} = c_{\theta,\tau}(\tilde{p}_{ijk}). \tag{57}$$

The function attains its minimum θ at τ and increases toward one as the adjusted score moves away from the classification boundary.

For $\theta < 1$ and $\alpha \in [\theta, 1]$,

$$c_{\theta,\tau}(u) \ge \alpha \iff u \le \tau(1 - r_\alpha) \text{ or } u \ge \tau + (1-\tau)r_\alpha, \tag{58}$$

where

$$r_\alpha = \sqrt{\frac{\alpha - \theta}{1 - \theta}}. \tag{59}$$

The procedure therefore returns $\tilde{p}$, $\hat{x}$, and γ as the common inputs to the prediction-guided search strategies in Section 4.3.

G Search specification

The hints and auxiliary searches share scores and confidence but are distinct configurations. The prediction-centered neighborhood and weighted Local Branching models retain the original objective and constraints. Incumbent-guided search adds an agreement neighborhood

$$Z^I = \{a : \gamma_a \ge \alpha,\ x^I_a = \widehat{x}_a\}, \qquad \sum_{a \in Z^I : x^I_a = 0} x_a + \sum_{a \in Z^I : x^I_a = 1} (1 - x_a) \le B_I. \tag{60}$$

When $B_I = 0$ the agreement coordinates are fixed; positive budgets permit correction. Equation 10 defines the second incumbent-guided neighborhood. The reference incumbent is fixed at invocation, rather than continuously updated.

A shared-deadline allocation is expressed as

$$T_{\text{remaining}} = \max\{0, 3600 - (t_{\text{now}} - t_{\text{start}})\}, \tag{61}$$

with t_{start} before online preprocessing. The auxiliary phase is capped at 600 seconds and also at the global deadline. Global lower bounds are obtained from the unrestricted model.

G.1 Prediction-guided Feasibility Pump

Let $\mathcal{P}$ be the full LP relaxation, including continuous state variables. Initialize the rounded vector from the prediction center. After rounding an LP point x^t to b^t, fix the arc variables and reoptimize the continuous variables in the original model. If no feasible completion exists, solve

$$\min_{(x,u,w,r)\in\mathcal{P}} \sum_{a\in\mathcal{B}} \gamma_a |x_a - b_a^t|. \tag{62}$$

For binary b^t, the objective is linear: $\sum_{a:b_a^t=0} \gamma_a x_a + \sum_{a:b_a^t=1} \gamma_a (1 - x_a)$. The next LP-feasible point is rounded and checked again. Repeated rounded points may be perturbed on low-confidence coordinates. Confidence remains fixed throughout. The procedure terminates when a feasible completion is found or its search budget is exhausted.

H Supplementary optimization results

Figure 6 gives final relative MIP gaps by instance and configuration. Values are percentages; unavailable values are shown in gray.

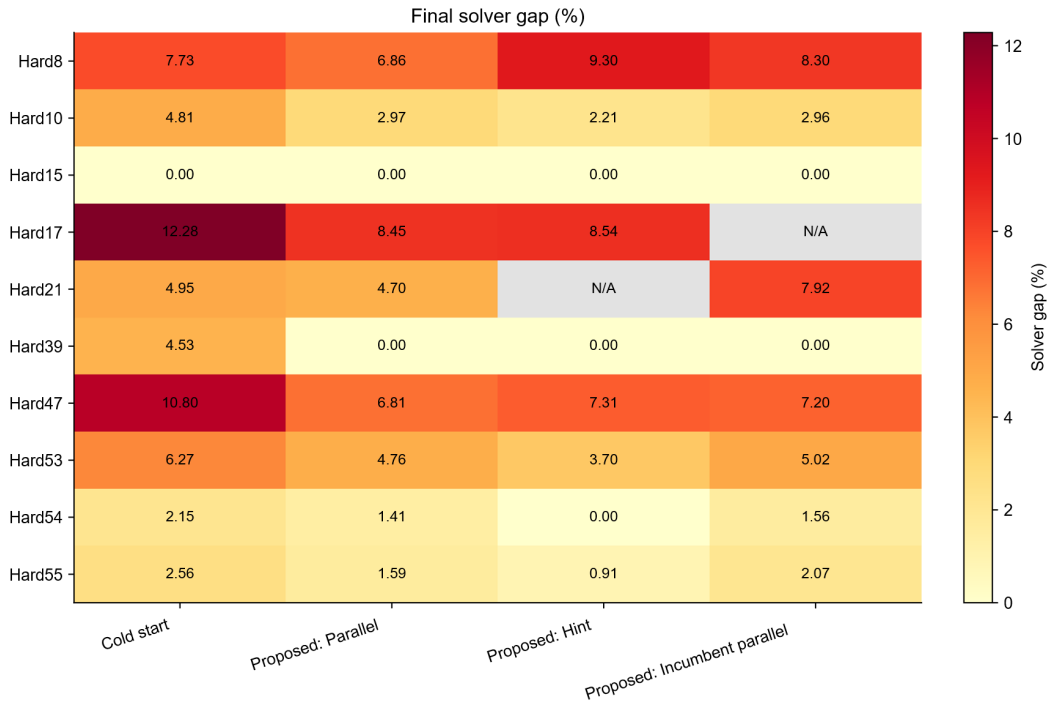

Figure 6: Final MIP gaps