POST-PARTITION BOUNDARY RELAXATION FOR REAL-TIME CLUSTER-FIRST LARGE-SCALE VEHICLE ROUTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large-scale capacitated vehicle routing problems (CVRPs) can require substantially more computation than strict real-time budgets allow, motivating cluster-first decomposition into smaller routing subproblems. Yet decomposition silently fixes a modeling choice: how strongly the resulting cluster boundaries should be enforced. We introduce *Post-Partition Boundary Relaxation (PBR)*, a learning-free mechanism that selectively relaxes these post-partition commitments. Given a fixed partition, PBR scores each customer by its assignment margin, the gap between its strongest and second-strongest cluster assignment, and uses these solver-free ambiguity scores to identify boundaries for joint reoptimization. Rather than moving individual customers directly, PBR reopens selected neighboring cluster pairs and lets the routing solver determine the resulting reassignment.

We evaluate PBR on large-scale CVRP instances under strict real-time deadlines. At $N{=}10{,}000$ nodes, monolithic optimization requires approximately 18 seconds, motivating cluster-first decomposition to substantially reduce routing latency. Across 594 instances, PBR shortens routes on 593, reducing mean and median cost by 3.3% relative to strict decomposition. Matched random boundary selection is substantially weaker, showing that the improvement is not explained by additional joint reoptimization alone, but by where reoptimization is applied.

The benefit, however, depends on the computational regime. It disappears when the residual time budget cannot absorb the larger merged subproblems. These results show that decomposition quality is not determined by the partitioner alone: *post-partition commitment is an independent algorithmic design axis*, and inexpensive ambiguity signals can identify where that commitment should be relaxed.

1 INTRODUCTION

Large-scale capacitated vehicle routing problems (CVRPs) can exceed strict real-time response deadlines even when approximate solutions are acceptable. Under such deadlines, computational failure is qualitatively different from approximation error: a solver that overruns the deadline returns no usable solution at all. Algorithm design under real-time constraints therefore couples two questions, *whether a solution can be produced in time* and, conditional on that, *how much performance value is lost in producing it*.

Cluster-first-route-second (CFRS) methods address the first question by partitioning a large instance into K clusters and solving the resulting routing subproblems independently (Zhou et al., 2025). This makes routing computation depend primarily on cluster size rather than the full instance size. Modern CFRS pipelines increasingly construct these clusters with learned or heuristic partitioners (Falkner & Schmidt-Thieme, 2023; Hou et al., 2023; Ye et al., 2024; Chin & Schiffer, 2026).

This decomposition creates a second issue: *commitment*. Once a customer is assigned to a cluster, that assignment becomes a hard boundary between independently optimized subproblems, and no route can connect customers on opposite sides of it (Santini et al., 2023). The routing solver cannot coordinate across that boundary, even when the partition provides evidence that the assignment is ambiguous. A customer that is nearly equally compatible with two neighboring clusters is still committed to exactly one of them.

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.

This observation separates two decisions that strict decomposition couples. The first is *where to partition*; the second is *how strongly to enforce the resulting partition*. Improving the partitioner addresses the first decision but does not change the commitment imposed by the second. We study post-partition commitment as an independent algorithmic design axis: *given a fixed partition, can coordination be selectively restored at the boundaries that appear uncertain?*

We introduce *Post-Partition Boundary Relaxation (PBR)*, a learning-free mechanism for doing so. Given a fixed partition, PBR computes a solver-free ambiguity score for each customer based on its assignment margin, the gap between its strongest and second-strongest cluster assignment. A small margin identifies a customer whose partition assignment is ambiguous, but does not imply that the customer should move. PBR instead aggregates these signals over neighboring cluster pairs to identify contested boundaries. A matching then selects a subset of those boundaries for joint reoptimization: the selected cluster pairs are merged into larger routing subproblems, while all other clusters remain independent. The routing solver determines the resulting reassignment. Thus, the ambiguity signal determines *where coordination is worth restoring*, while optimization determines *what the resulting solution should be*.

PBR operates entirely downstream of the partitioner. It neither replaces nor retrains the partitioner, and its selection signal requires no additional solution of the underlying routing problem. It can therefore be applied to different partitioning mechanisms while preserving the computational structure of CFRS. At one extreme, strict decomposition enforces every partition boundary; at the other, full joint optimization removes the decomposition altogether. PBR provides a selective intermediate regime in which only a small number of boundaries are reopened, subject to the available computational budget. The idea is conceptually related to *flexibility* in manufacturing systems, where selectively sharing resources can recover coordination benefits without fully coupling the system (Jordan & Graves, 1995). Our setting differs in both mechanism and purpose: flexibility is introduced only after a routing partition has been constructed, and its role is to determine which partition boundaries should be relaxed. We therefore use this literature as conceptual motivation rather than as a source of algorithmic guarantees.

Decomposition has long been used to scale vehicle-routing optimization, from classical cluster-first methods (Gillett & Miller, 1974; Fisher & Jaikumar, 1981) to decomposition embedded within modern routing heuristics (Santini et al., 2023) and learned large-scale routing methods (Li et al., 2021; Ye et al., 2024; Chin & Schiffer, 2026). Recent work has also adapted the decomposition procedure itself (Karaahmetoglu & Kim, 2026) and reconsidered decisions after decomposition. Li et al. (2021) learn which groups of incumbent routes to reoptimize, while Kerscher & Minner (2025) use fuzzy cluster membership to identify boundary customers eligible for inter-cluster local search. PBR differs in treating the boundary itself as the unit of relaxation: it aggregates assignment ambiguity at cluster boundaries, selects a matching of non-overlapping boundaries, and jointly reoptimizes each merged pair with the routing solver. Its ambiguity score requires no soft clustering and can be computed from any hard partition, and the mechanism is learning-free and designed for strict real-time deadlines. We study PBR in large-scale CVRP under such deadlines through three research questions:

RQ1 How does real-time scale change the computational regime? At what problem sizes and latency budgets does monolithic optimization stop producing a response, and what does decomposition change about the cost profile?

RQ2 Does selective boundary reopening improve solution quality? Does reopening a subset of partition boundaries improve solution quality relative to strict decomposition, and does the improvement survive changes in partition granularity, parameterization, and partition source?

RQ3 Can ambiguity identify where decomposition should be relaxed? Can a solver-free signal derived from a partition identify which cluster boundaries are worth reopening, rather than merely triggering additional reoptimization?

Our experiments establish three main findings. First, the real-time constraint creates a distinct computational regime in which monolithic optimization becomes unusable at scale. At $N = 10{,}000$, the first useful monolithic stopping checkpoint arrives after 18 seconds, whereas decomposition enables the routing computation to remain within the relevant scale regime when paired with a sufficiently fast partitioner. Second, selective boundary reopening improves strict decomposition consistently.

Across 594 instances, PBR shortens routes on 593, reducing mean and median route cost by 3.3%. Matched random boundary selection performs substantially worse, indicating that the improvement is associated with the location of the reopened boundaries rather than simply with performing additional joint reoptimization. The effect also persists across partition sources and on real CVRPLIB instances without modifying or retraining the partitioner. Third, PBR is constrained by the computational regime in which it operates. Its quality benefit disappears when the residual time budget cannot absorb the larger merged subproblems, as it depends on the downstream routing solver's ability to exploit those merged subproblems.

These results establish a complementary perspective on decomposition-based optimization. *Post-partition commitment is itself a design choice.* A partition should not be treated as equally reliable at every boundary. A lightweight ambiguity signal can selectively relax that commitment, preserving the computational advantages of decomposition while restoring coordination where the partition itself indicates that it may be most valuable.

2 METHOD: POST-PARTITION BOUNDARY RELAXATION

Given a partition of N customers into K clusters, strict decomposition solves each cluster independently. PBR selectively restores coordination across cluster boundaries in three steps: *assignment ambiguity*, *boundary selection*, and *joint reoptimization*. It first identifies customers whose cluster assignments are ambiguous, then aggregates these signals to select a sparse set of contested boundaries, and finally jointly reoptimizes the cluster pairs separated by those boundaries.

1. Assignment ambiguity. Let $a(i)$ denote the cluster to which customer i is assigned, x_i its coordinate, and μ_k the centroid of cluster k. We define $b(i)$ as the cluster other than $a(i)$ whose centroid is closest to i: $b(i) = \arg\min_{k \neq a(i)} \|x_i - \mu_k\|$. Let $d_a(i) = \|x_i - \mu_{a(i)}\|$, and $d_b(i) = \|x_i - \mu_{b(i)}\|$ denote the distances to the assigned and strongest alternative clusters, respectively. We measure assignment ambiguity using the normalized margin

$$M(i) = \frac{d_b(i) - d_a(i)}{d_b(i) + d_a(i)}. \tag{1}$$

A customer is considered ambiguous when $M(i) < \tau$, where τ controls how similar the two cluster assignments must be to trigger ambiguity. A margin near zero indicates that the customer lies at similar distances from the two centroids, while a larger positive margin indicates stronger geometric support for its assigned cluster. The margin lies in $(-1, 1]$ and becomes negative when the partition assigns a customer to a cluster other than its nearest centroid, as can occur under a capacity-respecting partition. An ambiguous customer identifies the candidate boundary $\{a(i), b(i)\}$. Importantly, ambiguity does not imply that customer i should be reassigned. The signal only identifies a boundary across which coordination may be valuable; any reassignment is determined subsequently by the routing optimization.

2. Boundary selection. PBR next aggregates customer-level ambiguity into boundary-level evidence. Let $G = (V_C, E_C)$ denote the cluster adjacency graph, with one vertex for each cluster and an edge $\{k, \ell\}$ between spatially adjacent clusters. We assign each boundary a weight equal to the number of ambiguous customers that identify its two clusters as their assigned and strongest alternative pair:

$$w(k, \ell) = \left|\{i : M(i) < \tau,\ \{a(i), b(i)\} = \{k, \ell\}\}\right|. \tag{2}$$

Thus, $w(k, \ell)$ measures how many ambiguous assignments contest the boundary between clusters k and ℓ, and is zero when no ambiguous customer identifies that pair.

A cluster can participate in at most one merged subproblem. The selected boundaries must therefore form a matching $S \subseteq E_C$, satisfying $d_S(k) \leq 1$ for every cluster k and consequently $|S| \leq \lfloor K/2 \rfloor$.

Ideally, boundary selection would prioritize the pairs for which joint reoptimization yields the largest route-cost reduction. Let $v(k, \ell)$ denote this improvement. Evaluating $v(k, \ell)$, however, requires solving the merged subproblem for (k, ℓ). PBR therefore uses $w(k, \ell)$ as a proxy for the unknown value of reopening the boundary. This gives the reference selection problem

$$S^\star = \operatorname*{argmax}_{S \subseteq E_C:\ d_S(k) \leq 1\ \forall k} \sum_{(k,\ell) \in S} w(k, \ell). \tag{3}$$

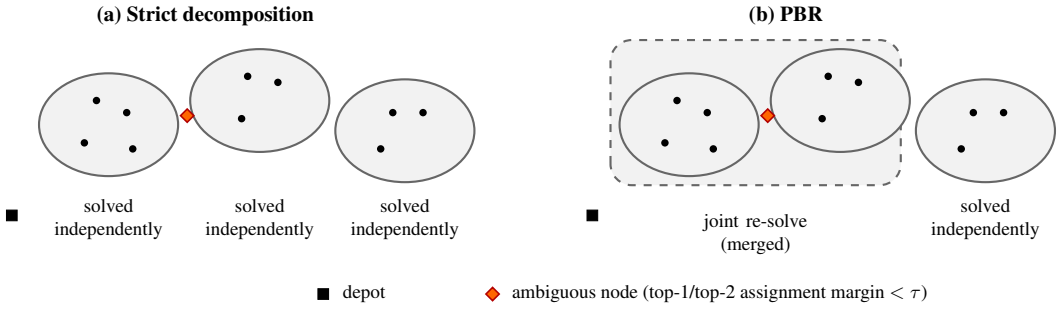

Figure 1: PBR applied to a neural CFRS partition. The diamond marks a node whose top-1/top-2 assignment margin falls below τ (step 1), ambiguous between the two adjacent clusters it sits between. **(a)** Strict decomposition commits this node to one cluster and solves every cluster independently. **(b)** PBR instead reopens that boundary, solving the two adjacent clusters as a single merged sub-CVRP (step 3). The third cluster, not part of any selected pair, is solved independently exactly as in (a).

The formulation selects a set of non-overlapping boundaries carrying the greatest approximate aggregate ambiguity. Our deployed implementation approximates this selection to reduce its computational overhead.

The deployed implementation restricts the candidate graph to a chain and performs a single greedy matching pass. Each cluster is first represented by an anchor seed, by default its member farthest from the depot. A nearest-neighbor tour over the K seeds induces an ordering of the clusters, and only the $K-1$ consecutive pairs in this ordering remain candidate boundaries. PBR then traverses the chain once, selecting a boundary whenever $w(k,\ell) > 0$ and neither endpoint has already been selected. Once a cluster is selected, it cannot participate in another merged pair.

This procedure approximates Equation 3 in two ways: it restricts the candidate edge set to the chain and replaces maximum-weight matching with a greedy pass. The resulting selection is therefore a *maximal* rather than maximum-weight matching and depends on the anchor used to construct the chain. Alternative anchor definitions are evaluated in the supplementary material.

3. Joint reoptimization. For each selected boundary $\{k,\ell\} \in S$, PBR replaces the two independent cluster subproblems with a single sub-CVRP containing all customers assigned to either k or ℓ. The merged problem retains the same depot and capacity constraints as the original routing problem. Its vehicle count remains endogenous, so the routing solver determines both the routes and any reassignment of customers between the two clusters. All unselected clusters are solved independently, exactly as under strict decomposition. Thus, PBR changes only which partition boundaries are jointly optimized; it does not alter the underlying routing model or constrain how customers are reassigned within a merged pair. Selected pairs remain independent of one another.

Figure 1 summarizes the procedure. Starting from the same partition, strict decomposition solves every cluster independently, whereas PBR uses assignment ambiguity to identify contested boundaries and jointly reoptimizes a sparse subset of neighboring cluster pairs. The ambiguity signal determines *where coordination is restored*; the routing solver determines *how that coordination is used*.

3 Experimental Setup

We evaluate PBR on controlled synthetic instances and the 100-instance CVRPLIB X-set (Uchoa et al., 2017). Synthetic coordinates are sampled uniformly from $[0,1)^2$ with demands normalized to fleet capacity; the $N{=}100$ setting reproduces the instances of Kool et al. (2018). Two partition sources cover learned and classical mechanisms: the neural optimal-transport head of Chin & Schiffer (2026) and capacitated k-means (Geetha et al., 2009).

Unless otherwise stated, experiments use $K{=}24$ clusters and ambiguity threshold $\tau{=}0.15$, fixed a priori rather than tuned. Strict decomposition and PBR share the same partition, routing solver

(PyVRP (Wouda et al., 2024) with `MaxRuntime`), constraints, and computational budget; only the joint reoptimization of selected pairs differs. Quality comparisons are paired at the instance level and use two-sided sign tests.

4 RESULTS

4.1 RQ1: HOW DOES DECOMPOSITION AFFECT COMPUTATIONAL SCALABILITY AT LARGE SCALE?

We first characterize the computational cost of monolithic and decomposed optimization as problem size grows. Table 1 reports both solution quality and measured end-to-end runtime.

At N=10,000, monolithic PyVRP needs a mean of 18.417s to reach its first useful stopping checkpoint (range 15.4–30.3s) across 594 instances, having already required approximately 1s at N=500, so it returns nothing usable within a near-second budget for larger instances. Fast construction alone does not resolve this trade-off. A classical polar-angle sweep heuristic (Gillett & Miller, 1974), implemented as a single greedy capacity-respecting pass with no local search, produces feasible solutions in approximately 0.03s. At N=10,000, however, its mean route distance is 2095.85, compared with 85.095 for monolithic optimization. Thus, the two computational extremes are both inadequate: monolithic optimization preserves solution quality but cannot respond within the real-time budget, whereas a purely constructive method easily satisfies the computational requirement but sacrifices most of the solution quality.

Cluster-first decomposition occupies the regime between these extremes. By replacing the full instance with smaller routing subproblems, it retains optimization within each cluster while substantially reducing the computational scale presented to the routing solver. At N=10,000, strict decomposition and PBR produce feasible solutions on all 594 instances, with mean distances of 96.615 and 93.443, respectively.

The computational benefit of decomposition also depends on partition construction being inexpensive. The neural partitioner requires 0.036s on average, whereas capacitated k-means requires a mean of 7.80s across 30 instances (range 5.75–8.25s). A computationally efficient partitioner is therefore essential: decomposition does not remove the computational bottleneck if partition construction is itself too expensive.

Table 1: Solution quality and measured end-to-end runtime at N=10,000 and K=24.

Scheme	mean distance	mean run time (s)
Monolithic (full-joint)	85.095	18.417
Decomp. strict	96.615	1.310
Decomp. PBR	93.443	1.509
Sweep (naive construction)	2095.85	$\sim$ 0.03

Together, these results establish the computational role of decomposition: it reduces the scale of the routing problems passed to the solver, provided that the partition itself can be constructed efficiently. Having established the computational motivation for decomposition at this scale, we next ask whether selectively relaxing its boundaries can recover some of the resulting quality loss.

4.2 RQ2: DOES SELECTIVE BOUNDARY REOPENING IMPROVE SOLUTION QUALITY?

We first compare PBR with strict decomposition using the same partition, routing solver, and per-subproblem time allocation. Across the 594 instances, PBR shortens routes on 593 and lengthens them on one, with a mean paired improvement of 3.29% (two-sided sign test, p=1.84×10^{-176}). Selectively reopening partition boundaries therefore consistently recovers part of the quality lost under strict decomposition without increasing the time allocated to each routing subproblem.

Because reopening boundaries merges cluster pairs into larger routing subproblems, a separate question is whether PBR simply behaves like a coarser decomposition. Starting from $K = 24$, PBR

reopens 11.2 boundaries on average, leaving approximately 13 independently solved groups; we therefore compare it with a direct strict $K = 12$ decomposition. Under capacitated k-means, direct $K = 12$ is 0.570% worse than strict $K = 24$, whereas PBR improves over strict $K = 24$. Thus, PBR's benefit is not reproduced by simply constructing a coarser partition with a comparable number of solved groups.

The benefit of PBR persists across partition granularities and becomes larger as the partition becomes finer. Holding all other settings fixed, PBR improves over strict decomposition by 1.59% at K=26, 2.81% at K=28, 3.62% at K=30, and 4.75% at K=32, with significant improvements at every value tested. The gain is therefore not specific to a single choice of K and is largest among the finer decompositions tested. The result is similarly insensitive to the other design choices tested. Varying the ambiguity threshold τ over a sixfold range changes the improvement by only 0.09 percentage points. The boundary-selection traversal is also robust to how clusters are represented when constructing the chain: using either cluster centroids or medoids as anchors yields comparable gains. Furthermore, the result is stable across the learned partitioner, with six checkpoints spanning three training seeds producing improvements between 3.18% and 3.35%. Together, these results show that the observed gain is robust to the ambiguity threshold, boundary-selection construction, and neural checkpoint.

In an additional numerical study, we evaluate PBR with two partitioning methods: the neural optimal-transport partitioner and capacitated k-means at N=100, 500, and 1000, using generous per-subproblem budgets. PBR improves over strict decomposition on all 235 paired instances, with improvements ranging from 6.9% to 11.3% across the six configurations ($p < 10^{-4}$ throughout). PBR improves strict decomposition under both the learned and capacitated k-means partitioners.

We next test whether the effect transfers from the controlled synthetic instances to the 100-instance CVRPLIB X-set. When capacitated k-means partitioning time is charged against a 0.9s computational budget, partition construction exhausts the budget on 38 instances, leaving 62 with positive routing time. PBR improves 51 of these 62 instances, with a median improvement of 3.88% (p=2.8×10^{-7}; Table 2).

The CVRPLIB results also reveal a computational limit to the quality gain. When at least 0.5s remains after partitioning, PBR improves 38 of 40 instances (p=1.5×10^{-9}). With less than 0.5s remaining, it improves 13 of 22 instances, which is not distinguishable from chance (p=0.52). In a companion experiment in which partition time is not charged against the routing budget, all 100 instances solve and PBR improves 99 of them, with a median improvement of 4.96% (p=1.6×10^{-28}).

Taken together, these results show that selective boundary reopening improves strict decomposition across partition granularities, parameter choices, partition sources, and both synthetic and CVRPLIB instances. The benefit is not unconditional, however: reopening creates larger routing subproblems, and the advantage disappears when too little computation remains to exploit the restored coordination. PBR therefore benefits from a computationally efficient partitioner that preserves sufficient routing time for the reopened subproblems.

Table 2: PBR against strict decomposition on all 100 CVRPLIB X-set instances, capacitated k-means partition, 0.9s budget with partition time charged against it. Gaps are medians to the best-known solution; p is a two-sided sign test on paired instances.

Outcome / stratum	n	strict gap	PBR gap	better/worse (p)
Partitioner exhausts deadline	38	—	—	—
Solved, ≥ 0.5s left	40	$+17.83\%$	$+10.84\%$	38/2 (p=1.5×10^{-9})
Solved, < 0.5s left	22	$+83.47\%$	$+53.77\%$	13/9 (p=0.52)
All solved	62	$+23.56\%$	$+17.60\%$	51/11 (p=2.8×10^{-7})
Unlimited partition time	100	$+20.84\%$	$+13.48\%$	99/1 (p=1.6×10^{-28})

4.3 RQ3: CAN AMBIGUITY IDENTIFY WHERE DECOMPOSITION SHOULD BE RELAXED?

RQ2 establishes that selectively reopening boundaries can improve solution quality. We now test whether the ambiguity signal identifies *which* boundaries are worth reopening. The key comparison holds the amount of relaxation fixed: PBR and the baseline reopen the same number of boundaries under the same matching constraints, but PBR uses ambiguity to choose them whereas the baseline chooses them randomly. We then solve the resulting merged subproblems and compare the final route costs. This isolates the value of *where* PBR restores coordination from the value of performing additional joint reoptimization itself.

For each instance, we compare PBR with 20 such random boundary selections. We report PBR's percentile rank among the resulting route costs, where lower values are better and 50% represents the median random selection. The experiment uses 206 held-out synthetic instances at $N{=}10{,}000$ and $K{=}24$ under the main experimental configuration, excluding the 100 instances used to train the merge-value model of Section 4.4 and instances for which the deployed selector reopens either zero boundaries or all 12 possible pairs, since no matched random comparison exists in those cases. Across 206 instances, PBR achieves a median rank of 30%, with a rank below 50% on 147 instances (Wilcoxon $p{=}8.2 \times 10^{-12}$). The result replicates on a disjoint instance set. Thus, when the amount of boundary reopening is held fixed, ambiguity-guided selection produces lower route costs than random boundary selection.

This comparison still combines two components of PBR: identifying contested boundaries from customer ambiguity and ranking those boundaries by their ambiguity weights. We therefore conduct a stricter control that holds the candidate set fixed as well. Both methods choose only among the same contested boundaries, but PBR ranks them using the ambiguity score whereas the baseline selects among them randomly. PBR achieves median percentile ranks of 30% at $N{=}2{,}000$, 20% at $N{=}5{,}000$, 15% at $N{=}10{,}000$, and 22.5% at $N{=}20{,}000$, with significant differences at every scale tested ($p \leq 7.5 \times 10^{-8}$ throughout). The advantage therefore persists even when the candidate boundaries are held fixed, showing that the ambiguity score contains information about which contested boundaries are more valuable to reopen.

Finally, we test whether customer ambiguity should be interpreted as predicting which individual customers will be reassigned after a boundary is reopened. Within reopened boundaries, ambiguous and non-ambiguous customers are reassigned at nearly identical rates, 86.8% and 86.4%, respectively. The ambiguity signal therefore identifies *where* coordination should be restored, rather than *which customers* should move.

4.4 CAN BOUNDARY SELECTION BE IMPROVED FURTHER?

The deployed PBR rule prioritizes low selection overhead: it restricts candidate boundaries to the chain and constructs a greedy maximal matching using the ambiguity weight w. We first test whether solving the resulting score-based selection problem more deliberately can improve route quality. Replacing the greedy traversal with exact score-based matching, while keeping the same ambiguity weights and candidate boundaries, reduces route cost by a median of 0.61–0.77% across the tested scales. Selection itself remains inexpensive and is faster in our implementation (4.99ms versus 12.12ms). The exact rule is not a direct replacement for the deployed method, however, because it requires the number of boundaries to reopen to be specified in advance, whereas the deployed traversal determines this implicitly.

Motivated by prior work on learning which routing subproblems to reoptimize (Li et al., 2021), we next ask whether the hand-crafted ambiguity weight can be improved by learning the value of reopening a boundary directly. A supervised merge-value model can improve boundary ranking, but requires labels obtained by solving candidate boundaries both separately and jointly, approximately 47 solves per training instance at $K{=}24$. Better ranking also does not translate into a consistent improvement in final route cost. Under a 0.86s solve budget, the learned selector changes median route distance relative to the ambiguity-based selector by -0.131%, -0.087%, -0.006%, and $+0.030\%$ at $N{=}2{,}000$, $5{,}000$, $10{,}000$, and $20{,}000$, respectively. Thus, the learned selector provides at most a small median routing improvement at the first three scales and is slightly worse at the largest scale, despite its additional training cost.

These experiments reveal two different opportunities for improving boundary selection. Better optimization of the existing ambiguity score yields a measurable routing benefit, whereas replacing the solver-free score with a learned estimate of merge value improves ranking without producing a consistent corresponding improvement in route quality. We therefore retain the ambiguity-based rule as the deployed selector: it requires neither solver-generated training labels nor an additional learned model, while leaving measurable room for improvements in how its scores are converted into a matching.

5 Discussion and Conclusion

The central implication of our results is that decomposition involves two distinct design decisions: *where to partition* and *how strongly to enforce the resulting partition*. Cluster-first methods often treat the resulting partition as the computational structure passed to downstream routing, although prior work has also introduced post-decomposition improvement across clusters (Li et al., 2021; Kerscher & Minner, 2025). PBR makes the degree of post-partition commitment explicit: it leaves the partition unchanged but selectively removes some of the boundaries that the partition creates. This perspective places strict decomposition and monolithic optimization at opposite ends of a continuum: the former preserves all partition boundaries, the latter preserves none, and PBR selectively restores coordination between them.

This view also exposes a fundamental trade-off. Reopening a boundary restores decisions that strict decomposition had artificially separated, but it simultaneously creates a larger subproblem that must be solved within the available computational budget. Post-partition relaxation is therefore valuable only while the benefit of restored coordination exceeds the computational cost of the larger solve. The residual-budget results in Section 4.2 illustrate this boundary directly. More generally, the appropriate degree of decomposition cannot be determined from problem size alone; it depends jointly on partition granularity, partitioning cost, the remaining routing budget, and the downstream solver.

The role of ambiguity should likewise be understood as allocating computational coordination rather than modifying the partition directly. PBR does not infer a corrected customer assignment. Instead, ambiguity identifies locations at which the original decomposition is comparatively weak, and optimization is then allowed to reconsider the decisions on both sides of that boundary. This separation is useful because the screening decision is inexpensive and solver-free, while the combinatorial decision of how routes should change remains with the routing solver.

PBR inherits several limitations from this computational trade-off. First, the deployed boundary selector is deliberately approximate: it restricts candidate boundaries to a chain and uses a greedy maximal matching. More expensive selection procedures can improve boundary choice, but introduce additional requirements such as a prescribed number of reopened boundaries or solver-generated training labels. Second, the current experiments establish the mechanism for CVRP and spatial cluster-first decomposition; whether post-partition ambiguity provides a useful relaxation signal for other decomposition structures remains open. These limitations suggest that post-partition relaxation should not be viewed as a universally beneficial correction to decomposition. Rather, it introduces an additional control available to decomposition-based algorithms: when sufficient computation remains, selected commitments can be relaxed to recover coordination without abandoning decomposition altogether.

Taken together, these results suggest that the output of a decomposition method need not be treated as an all-or-nothing commitment. A partition can remain computationally useful while selected boundaries are reopened when there is sufficient evidence and computational capacity to do so. PBR demonstrates this principle with a particularly lightweight mechanism: a solver-free ambiguity signal identifies where coordination should be restored, while the routing solver determines how to use that coordination. More broadly, the results identify post-partition commitment as a design dimension distinct from partition construction itself, opening a complementary direction for decomposition methods that focuses not only on constructing better partitions, but also on deciding which of their boundaries should remain binding.

AI USE STATEMENT

We used generative AI tools for code generation and drafting throughout; all reported results were independently verified before inclusion.

REPRODUCIBILITY STATEMENT

Code, exact seeds, and the decode/budget protocol will be released in a public git repository upon acceptance.

REFERENCES

Samuel J. K. Chin and Maximilian Schiffer. Neural cluster first, route second: One-shot capacitated vehicle routing via differentiable optimal transport. *arXiv preprint arXiv:2605.09301*, 2026.

Jonas K. Falkner and Lars Schmidt-Thieme. Neural capacitated clustering. *arXiv preprint arXiv:2302.05134*, 2023.

Marshall L Fisher and Ramchandran Jaikumar. A generalized assignment heuristic for vehicle routing. *Networks*, 11(2):109–124, 1981.

S. Geetha, G. Poonthalir, and P. T. Vanathi. Improved k-means algorithm for capacitated clustering problem. *INFOCOMP Journal of Computer Science*, 8(4):52–59, 2009.

Billy E. Gillett and Leland R. Miller. A heuristic algorithm for the vehicle-dispatch problem. *Operations Research*, 22(2):340–349, 1974. doi: 10.1287/opre.22.2.340.

Qingchun Hou, Jingwei Yang, Yiqiang Su, Xiaoqing Wang, and Yuming Deng. Generalize learned heuristics to solve large-scale vehicle routing problems in real-time. In *International Conference on Learning Representations*, 2023.

William C. Jordan and Stephen C. Graves. Principles on the benefits of manufacturing process flexibility. *Management Science*, 41(4):577–594, 1995. doi: 10.1287/mnsc.41.4.577.

Oguzhan Karaahmetoglu and Hyong Kim. Adaptive cluster-first route-second decomposition for industrial-scale vehicle routing. *arXiv preprint arXiv:2606.31820*, 2026.

Christoph Kerscher and Stefan Minner. Decompose-route-improve framework for solving large-scale vehicle routing problems with time windows. *Transportation Research Part E: Logistics and Transportation Review*, 204:104409, 2025.

Wouter Kool, Herke Van Hoof, and Max Welling. Attention, learn to solve routing problems! *arXiv preprint arXiv:1803.08475*, 2018.

Sirui Li, Zhongxia Yan, and Cathy Wu. Learning to delegate for large-scale vehicle routing. *Advances in neural information processing systems*, 34:26198–26211, 2021.

Alberto Santini, Michael Schneider, Thibaut Vidal, and Daniele Vigo. Decomposition strategies for vehicle routing heuristics. *INFORMS Journal on Computing*, 35(3):543–559, 2023.

Eduardo Uchoa, Diego Pecin, Artur Pessoa, Marcus Poggi, Thibaut Vidal, and Anand Subramanian. New benchmark instances for the capacitated vehicle routing problem. *European Journal of Operational Research*, 257(3):845–858, 2017. doi: 10.1016/j.ejor.2016.08.012.

Niels A. Wouda, Leon Lan, and Wouter Kool. PyVRP: A high-performance VRP solver package. *INFORMS Journal on Computing*, 36(4):943–955, 2024. doi: 10.1287/ijoc.2023.0055.

Haoran Ye, Jiarui Wang, Helan Liang, Zhiguang Cao, Yong Li, and Fanzhang Li. GLOP: Learning global partition and local construction for solving large-scale routing problems in real-time. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 20284–20292, 2024.

Fangting Zhou, Attila Lischka, Balázs Kulcsár, Jiaming Wu, Morteza Haghir Chehreghani, and Gilbert Laporte. Learning for routing: A guided review of recent developments and future directions. *Transportation Research Part E: Logistics and Transportation Review*, 202:104278, 2025.