REDUCE-THEN-SOLVE: ITERATIVE SEARCH VIA REDUCTION FOR COMBINATORIAL OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural combinatorial optimization (NCO) has emerged as a promising approach to combinatorial optimization problems (COPs). Existing Learning-to-Construct (L2C) methods have complementary limitations: global prediction (GP) provides scalable guidance but struggles with complex constraints, while local construction (LC) handles constraints well but incurs sequential solving overhead. We therefore propose Reduce-then-Solve (RS), a general framework that iteratively retains promising structures from an incumbent, reduces the COP to a residual subproblem, and re-solves it. For NCO, we instantiate RS as COReformer, a Learning-to-Search (L2S) method that couples GP and LC through problem reduction, with Perturbed Rectified Flow (PRF) as the global predictor and SymNCO as the local constructor. At each iteration, PRF identifies promising incumbent structures to induce a size-controllable LinkGraph, which SymNCO then solves to refine the incumbent. PRF adapts rectified flow to iterative search by transporting over perturbed-solution distributions rather than a single optimum. Extensive Experiments on TSP, CVRP, and CVRPTW show that COReformer achieves SOTA among learning-based solvers across prior L2C, L2S, and D&C comparisons. A learning-free RS instantiation on MCut further shows that the framework extends beyond neural routing.

1 INTRODUCTION

In recent years, Neural Combinatorial Optimization (NCO) (Pan et al., 2025; Sun & Yang, 2023; Wang et al., 2024) has emerged as a promising paradigm for solving Combinatorial Optimization Problems (COPs). Rather than relying on handcrafted heuristics and problem-specific algorithms (Vidal et al., 2012; Helsgaun, 2017; Applegate & Cook, 2006; Nagata & Kobayashi, 2013), NCO leverages data-driven learning to develop effective optimization policies. Existing NCO methods can be broadly categorized into Learning-to-Construct (L2C), Learning-to-Search (L2S), and Divide-and-Conquer (D&C). Among them, L2C has been extensively studied and can be further divided into Global Prediction (GP), Local Construction (LC), and Adaptive Expansion (AE) (Ma et al., 2025a;b).

To better understand the design space of L2C, we briefly examine the strengths and limitations of these three paradigms. **GP methods** (Joshi et al., 2019; Fu et al., 2021) predict a global solution structure through one or a few neural inference steps. Typically trained with supervised objectives, they scale favorably with problem size and underpin many recent generative NCO methods (Li et al., 2023; 2024b). GP is particularly effective for problems with relatively simple constraints, such as TSP, offering efficient inference even at large scales. However, its reliance on global prediction becomes challenging for problems with complex constraints such as CVRP, where imperfect heatmap predictions can lead to *prediction conflict* (Ma et al., 2025a) during heuristic decoding, degrading the quality of the final solution. **LC methods** (Kool et al., 2019; Kwon et al., 2020; Kim et al., 2022; Drakulic et al., 2023; 2025) instead formulate solution construction as a Markov decision process (MDP) and builds solutions autoregressively, enabling dynamic feasibility control through step-wise state updates. However, the sequential construction process requires repeated neural inference and therefore incurs substantial solving overhead, particularly as the problem size increases. Moreover, representative RL-based LC methods (Berto et al., 2023) face additional scalability challenges due to long-horizon decision making with sparse and delayed rewards, which makes training and generalization increasingly difficult as the problem size grows. **AE methods** (Ma et al., 2025a; Wang et al., 2026) attempt to combine the two by adaptively expanding partial solutions, but still rely heavily on a global predictor and thus inherit its limitations under complex constraints.

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.

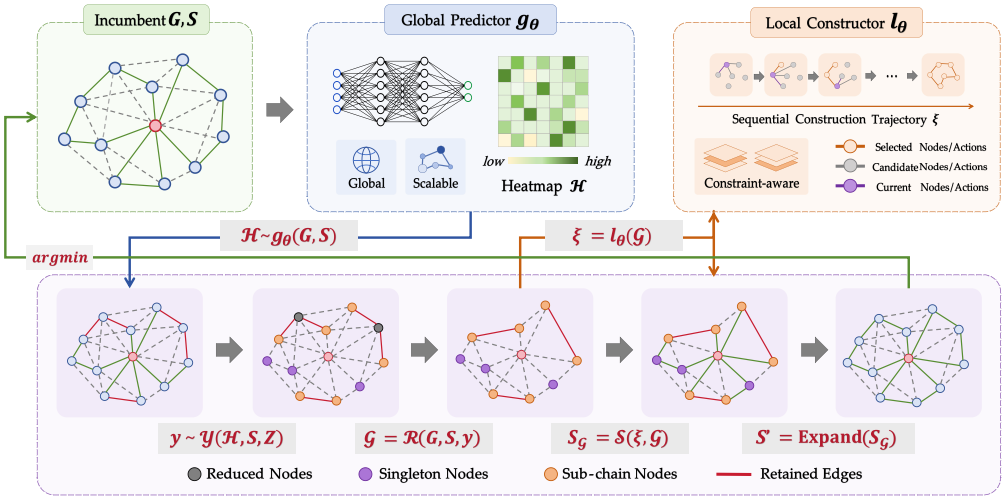

Figure 1: Overview of the iterative Reduce-then-Solve (RS) framework instantiated as COReformer. Given an incumbent (G, S), the global predictor g_θ produces a scalable retention heatmap $\mathcal{H}$, while the local constructor l_θ performs constraint-aware sequential construction. Progressive reduction samples a retention indicator $\mathbf{y} \sim \mathcal{Y}(\mathcal{H}, S, Z)$ and forms a LinkGraph $\mathcal{G} = \mathcal{R}(G, S, \mathbf{y})$ of residual size Z. A trajectory $\xi \sim l_\theta(\mathcal{G})$ yields $S_\mathcal{G} = \mathcal{S}(\xi, \mathcal{G})$, which is mapped back to update the incumbent.

The above analysis reveals complementary limitations of existing L2C paradigms. GP provides scalable global guidance but struggles with complex constraints, whereas LC offers effective constraint handling at the cost of sequential solving overhead. RL-based LC approaches, such as RL4CO (Berto et al., 2023), further face scalability challenges on large instances. While AE takes an initial step toward combining GP and LC through adaptive partial-solution expansion, its heuristic expansion procedure and GP-dominated learning module still limit its ability to handle complex constraints. Nevertheless, AE suggests that jointly leveraging GP and LC is a promising direction: rather than treating them as alternative L2C construction paradigms, we generalize this insight into an iterative optimization process, leading to our proposed **Reduce-then-Solve (RS)** framework. RS iteratively retains promising structures from an incumbent, reduces the original problem to a residual subproblem, and re-solves it to update the incumbent. The *reduce* stage enables global reorganization by selectively preserving promising assignments, while the *solve* stage refines the residual task under its constraints. RS is not tied to any particular solver or learning paradigm, allowing both learned and learning-free components to be incorporated into the framework.

To instantiate RS for NCO, we develop **COReformer**, an L2S method that realizes the two stages with GP and LC, as illustrated in Fig. 1. COReformer employs Perturbed Rectified Flow (PRF) as the global predictor and SymNCO (Kim et al., 2022) as the local constructor. PRF adapts rectified flow (Liu et al., 2022) to iterative search by transporting between distributions of perturbed feasible solutions rather than a single optimum, yielding a non-degenerate target distribution while allowing the source distribution to exploit incumbent information during search. Conditioned on the incumbent, PRF produces a heatmap that scores the likelihood of retaining each decision variable, which guides the contraction of retained incumbent fragments into a compact LinkGraph of controllable size, whose nodes represent unfixed variables and contracted fragments. SymNCO then solves the residual task, and the reconstructed solution is mapped back to update the incumbent. The highlights include:

- We propose Reduce-then-Solve (RS), a general optimization framework that transforms incumbent-guided structure preservation into a controllable residual search process, supporting global reorganization and constraint-aware refinement in an iterative loop.
- We develop COReformer as the NCO instantiation of RS, where PRF, LinkGraph-based reduction, and SymNCO respectively provide global retention guidance, compact residual construction, and local constraint-aware solving. PRF further adapts RF with feasibility-preserving perturbations to avoid degenerate transport targets and better match incumbent-conditioned inference.
- Experiments on TSP, CVRP, and CVRPTW demonstrate that COReformer achieves SOTA performance among learning-based routing solvers. We further validate RS with a learning-free MCut instantiation, showing that the framework applies beyond neural routing.

2 Related Work

L2C Solvers. 1) GP methods learn global solution structures through supervised (Joshi et al., 2019; Fu et al., 2021), unsupervised (Min et al., 2023), or meta-reinforced learning (Qiu et al., 2022). Recent advances have increasingly adopted generative models, including diffusion-based methods (Sun & Yang, 2023; Sanokowski et al., 2024; Li et al., 2023), consistency and flow-based models (Li et al., 2024b; 2025). **2)** LC methods instead learn sequential decision policies that autoregressively construct solutions while conditioning on the evolving partial solution, with reinforcement learning as the predominant training paradigm (Kool et al., 2019; Kwon et al., 2020). Supervised and self-improved training have also been introduced to guide sequential decisions and improve scalability (Drakulic et al., 2023; 2025; Luo et al., 2023; 2025). **3)** AE methods (Ma et al., 2025a; Wang et al., 2026) further explore the interaction between GP and LC by adaptively expanding partial solutions.

L2S Solvers. L2S methods iteratively refine an incumbent solution rather than constructing one from scratch. Early efforts have explored learning classical local search operators, such as learning 2-opt heuristics through deep reinforcement learning (d O Costa et al., 2020), while NeuOpt-GIRE (Ma et al., 2023) incorporates Tabu Search (Glover, 1989) into neural optimization to enable broader solution exploration. LEHD and SIL (Luo et al., 2023; 2025) improve incumbents via iterative reconstruction at inference time. More recently, GenSCO (Li et al., 2025) combines solution perturbation with a flow-based generative model for iterative refinement. These approaches primarily improve solutions through direct local modification, partial reconstruction, or solution-level generation.

D&C Solvers. D&C methods typically scale neural solvers by decomposing large instances into smaller subproblems that are solved independently and then merged. TAM-LKH (Hou et al., 2023) learns a dividing policy that partitions a large VRP into capacity-feasible sub-TSPs, subsequently solved in parallel by LKH-3 (Helsgaun, 2017). GLOP (Ye et al., 2024) hybridizes a non-autoregressive global policy for coarse partitioning with an autoregressive local policy for fine-grained construction. UDC (Zheng et al., 2024) further unifies this paradigm via Divide-Conquer-Reunion training, pairing a GNN-based divider with a fixed-length sub-path solver across diverse COPs.

Positioning of RS and COReformer. Relative to these lines of work, RS is not tied to a specific NCO category, but provides a general framework that reduces an incumbent into a residual task and re-optimizes it; its scoring, reduction, and solving components can be learned or hand-designed. COReformer instantiates this framework for NCO and is essentially an L2S method, using GP for global retention guidance and LC for constraint-aware residual solving. RS shares with LNS (Shaw, 1998) the idea of fixing part of the incumbent, and with D&C the intuition of solving a smaller task, but differs in residual formation. LNS keeps the original problem and restricts admissible changes via destroy-and-repair, while D&C partitions the instance into independent subproblems from scratch. RS instead contracts retained fragments into a residual of the original COP, reducing the problem scale while preserving the original constraints for re-optimizing unfixed variables.

3 Preliminaries

Problem Definition. We consider COPs defined on a graph $G = (V, E)$, where V and E denote the node and edge sets, respectively. A feasible solution is represented by a binary decision vector $S \in \{0, 1\}^M$, where each entry corresponds to a problem-specific decision variable. Let $\Omega(G)$ denote the feasible solution space and $c(S, G)$ the corresponding objective cost. The goal is to find an optimal solution $S^* = \underset{S \in \Omega(G)}{\operatorname{argmin}}\, c(S, G)$. In this paper, we focus on three representative routing problems: TSP, CVRP, and CVRPTW. For these problems, we use the standard edge-based representation with $M = N^2$, where $N = |V|$ and $S_{i \cdot N + j} = 1$ indicates that edge (i, j) is selected. ***Formal definitions and a complete symbol list are given in Appendix B and Table 16.***

Generative GP Model. Recent generative models, such as diffusion, consistency and flow-based models (Sun & Yang, 2023; Li et al., 2023; 2024b; 2025), have shown strong global prediction capabilities for COPs. In this paper, we adopt rectified flow (RF) (Liu et al., 2022) as the foundation of our global predictor. RF learns continuous transport from a source distribution π_0 to a target distribution π_1 by modeling an ordinary differential equation (ODE), $\mathrm{d}S_\tau = \mathbf{v}(S_\tau, \tau)\mathrm{d}\tau$, which aligns naturally with COPs, where both distributions can be defined over feasible solutions. Given endpoint samples $S_0 \sim \pi_0$ and $S_1 \sim \pi_1$, RF forms a linear interpolation and learns a velocity field

$\mathbf{v}_\theta(S_\tau, \tau)$ to match transport direction. This source-target formulation facilitates incorporating the incumbent solution into global prediction, while the learned velocity field enables efficient inference by numerically integrating the ODE from $\tau = 0$ to $\tau = 1$ in only a few steps. We omit instance conditioning in this generic RF objective; conditional PRF later includes G. Along each interpolation path, transport velocity is constant and equals $S_1 - S_0$, and the model is trained by minimizing:

$$\mathcal{L}_{\mathrm{RF}} = \int_0^1 \mathbb{E}\left[\|(S_1 - S_0) - \mathbf{v}_\theta(S_\tau, \tau)\|_2^2\right] \mathrm{d}\tau;\ S_\tau = \tau S_1 + (1-\tau)S_0,\ \tau \in [0,1]. \tag{1}$$

RL-based LC Model. RL formulates solution construction as an MDP, where a policy π_θ sequentially selects feasible actions over T steps. At step t, given instance G and state s_t encoding construction progress, it samples $a_t \in \mathbf{a}(s_t)$ from $\pi_\theta(a_t \mid s_t, G)$. A trajectory $\xi = (s_0, a_0, s_1, \ldots, s_{T-1}, a_{T-1})$ induces a solution $S = \mathcal{S}(\xi, G)$ with terminal reward $R(G, S) = -c(S, G)$, and its log-probability decomposes as $\log \pi_\theta(\xi \mid G) = \sum_{t=0}^{T-1} \log \pi_\theta(a_t \mid s_t, G)$. The policy is optimized with REINFORCE to maximize the expected terminal reward, or equivalently to minimize:

$$\mathcal{L}_{\mathrm{RL}} = -\mathbb{E}_{\xi\sim\pi_\theta}\left[\big(R(G, \mathcal{S}(\xi, G)) - b\big)\ \log \pi_\theta(\xi \mid G)\right], \tag{2}$$

where b denotes a variance-reduction baseline. In this paper, we adopt SymNCO (Kim et al., 2022) as the RL-based local constructor for the *solve* stage of COReformer. Built on an attention-based policy, SymNCO improves RL training on Euclidean routing problems by exploiting geometric symmetries, where transformed instances preserve optimality and symmetric tours share the same objective value.

4 Methodology

4.1 Perturbed Rectified Flow for Global Prediction

Challenges of RF on COPs. When RF is applied to COPs, each instance is associated with a single target solution, reducing the target distribution to one isolated sample. Consequently, trajectories from different source solutions converge to the same endpoint, leaving little crossing at intermediate states and preventing RF from learning a shared velocity field across diverse source-target pairs. The transport therefore collapses onto a fixed high-quality structure, making the endpoint readily inferable from intermediate states, especially for binary solution representations. A further limitation arises in the iterative search setting of COReformer. During training, the source distribution is constructed from randomly sampled feasible solutions. In contrast, later iterations start from incumbent solutions generated by the search process. Since these incumbents are high-quality yet suboptimal, they follow a markedly different distribution from random solutions, creating a source-distribution mismatch between training and iterative inference.

Perturbed Rectified Flow (PRF). To address the challenges above, we propose PRF, which retains the rectified interpolation structure of RF but redefines the source and target as distributions over perturbed solution structures. For each instance, let S^* denote the optimal solution and let $\mathcal{P}(\cdot,\cdot,\cdot)$ denote a randomized perturbation operator that applies K local modification steps while preserving feasibility. PRF defines the source and target distributions as

Algorithm 1 PRF as Global Predictor $\hat{g}_\theta$.

Input: PRF model $\mathbf{v}_\theta(\cdot,\cdot,\cdot)$; instance G; source state S_0; time points $0 = \tau_0 < \tau_1 < \cdots < \tau_{N_\tau} = 1$
for $i = 0$ to $N_\tau - 1$ **do**
  $\mathbf{d}_{\tau_i} = \mathbf{v}_\theta(S_{\tau_i}, \tau_i, G)$
  $S_{\tau_{i+1}} \leftarrow S_{\tau_i} + \mathbf{d}_{\tau_i}(\tau_{i+1} - \tau_i)$
end for
Output: S_1 as the global heatmap $\mathcal{H}$

$$S_1 \sim \pi_1 = \mathcal{P}(S^*, G, K_1), \quad S_0 \sim \pi_0 = \mathrm{Mix}(\pi_0^{\mathrm{rnd}}, \mathcal{P}(S^*, G, K_2)), \quad K_2 > K_1, \tag{3}$$

where $\mathrm{Mix}(\cdot,\cdot)$ denotes equal-probability mixing, π_0^{rnd} denotes randomly sampled feasible solutions, and K_1 and K_2 control perturbations around S^*. The perturbed target distribution avoids collapse to an isolated sample and restores trajectory crossing at intermediate states, while the mixed source distribution provides partially structured states that better align with search incumbents, with random samples maintaining coverage in the first iteration. Although Eq. 1 is written in velocity form, we use a target-state parameterization: the network outputs $\hat{S}_1 = f_\theta(S_\tau, \tau, G)$ as a heatmap, and the velocity is induced as $\mathbf{v}_\theta(S_\tau, \tau, G) = \hat{S}_1 - S_0$. Thus matching $(S_1 - S_0)$ reduces to predicting S_1. At inference, the current incumbent S is perturbed to form S_0, and the induced PRF dynamics are integrated to obtain the global heatmap $\mathcal{H}$ for the reduce stage, as described in Algorithm 1.

4.2 REDUCTION AND LINKGRAPH FOR LOCAL CONSTRUCTION

Progressive Reduction. Given an instance G and a feasible solution $S \in \{0,1\}^M$, we introduce a retention indicator $\mathbf{y} \in \{0,1\}^M$, where $y_j = 1$ indicates that decision variable S_j is retained and fixed in the partial solution. Let $z(G, S, \mathbf{y})$ denote the size of the resulting residual task, with its specific definition depending on the underlying COP. Progressive reduction forms a nested sequence $Y = \{\mathbf{y}_0, \mathbf{y}_1, \mathbf{y}_2, \ldots\}$, where $\mathbf{y}_0 = \mathbf{0}$ and $|\mathbf{y}_{i+1}| = |\mathbf{y}_i| + 1$, with $|\mathbf{y}|$ denoting the number of retained variables. Let $z_i = z(G, S, \mathbf{y}_i)$ denote the residual size at step i, with z_0 corresponding to the size of the original problem. Along the retention sequence Y, the residual size is non-increasing, while the amount of reduction at each step depends on the problem-specific reduction rule. During inference, each local iteration samples a target residual size Z from $[Z_{\min}, Z_{\max}]$ to control solve-stage complexity and diversify the induced LinkGraphs. Decision variables are progressively retained until z_i reaches Z or the largest attainable value below Z; if the resulting residual task is smaller than Z, dummy variables are added and masked during residual solving.

LinkGraph Construction. Given the selected retention indicator $\mathbf{y}$, a problem-specific reduction operator $\mathcal{R}$ maps the instance G, incumbent solution S, and retained variables $\mathbf{y}$ to a LinkGraph $\mathcal{G} = \mathcal{R}(G, S, \mathbf{y}) = (\mathcal{V}, \mathcal{E})$. For the three COPs considered in this paper, $\mathcal{R}$ partitions the incumbent solution S into consecutive fragments $\mathcal{Q} = \{q_1, q_2, \ldots\}$, where each $q_i = (q_i^0, \ldots, q_i^{\ell_i})$ with $q_i^j \in V$ is either a singleton ($\ell_i = 0$) or a sub-chain ($\ell_i \geq 1$) whose edges are retained. A problem-specific mapping ϕ contracts each fragment q_i into a subset $\phi(q_i) \subseteq \mathcal{V}$ of one or two reduced nodes, with the reduced node set given by $\mathcal{V} = \bigcup_{q_i \in \mathcal{Q}} \phi(q_i)$. Since $|\phi(q_i)| \leq |q_i|$, the contraction does not increase the problem size, while the edge set $\mathcal{E}$ specifies the connectivity among the induced nodes according to the fragment structure in $\mathcal{Q}$. The mapping ϕ is instantiated as: **1) TSP.** A singleton maps to one node carrying the coordinate of q_i^0, while a sub-chain maps to its two endpoints, each carrying its coordinate and the travel distance accumulated along q_i. **2) CVRP.** Fragments are formed within depot-separated routes, with the depot always a singleton. A singleton retains its coordinates and demand, while a sub-chain maps to both endpoints with their coordinates, together with the aggregated demand and travel distance along q_i. **3) CVRPTW.** The route structure follows CVRP, but sub-chains are directed. A singleton maps as in CVRP, while a sub-chain maps only to its entry q_i^0, encoding entry and exit coordinates, aggregated demand and travel distance, and the service and time-window information accumulated along q_i.

LinkGraph Solving. Given $\mathcal{G}$, the reduced task is formulated as an MDP as in Sec. 3. At step t, the state s_t encodes the current position, visited nodes, and partners connected in $\mathcal{E}$, and a policy $\pi_\theta(a_t \mid s_t, \mathcal{G})$ selects the next node from the feasible action set while masking visited nodes, dummy padding, and infeasible choices under problem-specific constraints. Visiting a node with a partner in $\mathcal{E}$ forces that partner as the next action, traversing collapsed sub-chain interiors without explicit decisions. A trajectory ξ on $\mathcal{G}$ induces a solution $\mathcal{S}(\xi, \mathcal{G})$ which is then mapped back to a full feasible solution on G by unpacking sub-chains according to the fragment structure in $\mathcal{Q}$.

4.3 REDUCE-THEN-SOLVE FRAMEWORK FOR NCO AND COREFORMER

RS for NCO. When RS is applied to NCO, its reduce and solve stages can be realized by a global predictor g_θ and a local constructor l_θ, where, with a slight abuse of notation, θ denotes the respective parameters of each module. Given an instance G and an incumbent solution S, the global predictor first produces a retention heatmap $\mathcal{H}$ conditioned on (G, S). Guided by $\mathcal{H}$, progressive reduction stochastically selects retained variables to form the indicator $\mathbf{y}$, which is then used to construct the LinkGraph $\mathcal{G}$. The local constructor then generates a trajectory ξ on $\mathcal{G}$, inducing a solution $\mathcal{S}(\xi, \mathcal{G})$ that is expanded back to G and locally improved. The overall process can be summarized as follows:

$$\mathcal{H} = g_\theta(G, S); \quad \mathbf{y} \sim \mathcal{Y}(\mathcal{H}, S, Z); \quad \mathcal{G} = \mathcal{R}(G, S, \mathbf{y}); \quad \xi \sim l_\theta(\mathcal{G}); \tag{4}$$

$$S' = \mathrm{LocalSearch}\left(\mathrm{Expand}\left(\mathcal{S}(\xi, \mathcal{G})\right)\right); \quad S \leftarrow \operatorname{argmin} c(s, G), s \in \{S, S'\}, \tag{5}$$

where $\mathcal{Y}(\mathcal{H}, S, Z)$ denotes the stochastic progressive reduction process guided by $\mathcal{H}$ under the target residual size Z; $\mathrm{Expand}(\cdot)$ maps the solution on the LinkGraph back to a complete solution on G, and $\mathrm{LocalSearch}(\cdot)$ performs lightweight local improvement. Notably, $\mathcal{H}$ provides global prediction information conditioned on the current incumbent S, rather than specifying a single reduction. Therefore, within a global iteration, $\mathcal{H}$ can be reused to sample different retention indicators and LinkGraphs, reducing calls to g_θ and improving search efficiency.

COReformer Instantiation. We instantiate the NCO-level RS framework as COReformer, with PRF as $\hat{g}_\theta$ and SymNCO as $\hat{l}_\theta$. Both modules adopt GAT-based architectures, with details in Appendix C. Inference follows Algorithm 2, with further details in Appendix E. We sample a random feasible solution and decode the PRF heatmap with local search to obtain the initial incumbent $\mathbf{x}^*$. In each subsequent global iteration, $\mathbf{x}^*$ is randomly perturbed and fed to $\hat{g}_\theta$ to produce a heatmap $\mathcal{H}$, which is decoded into a global candidate. The same $\mathcal{H}$ is reused for I_L local iterations with different residual sizes Z, inducing diverse LinkGraphs solved by $\hat{l}_\theta$. After local improvement, the best among the incumbent and all candidates is retained as $\mathbf{x}^*$.

Algorithm 2 COReformer Inference.

Input: PRF (global predictor) $\hat{g}_\theta(\cdot,\cdot)$ with given time points; SymNCO (local constructor) $\hat{l}_\theta$; Instance G; perturbation range $[K_{\min}, K_{\max}]$; residual range $[Z_{\min}, Z_{\max}]$; perturbation operator $\mathcal{P}_K(\cdot,\cdot,\cdot)$; decoding function $\mathcal{D}(\cdot,\cdot)$; cost function $c(\cdot,\cdot)$; global steps I_G; local steps I_L.
Randomly initialize solution $\mathbf{x}$.
$\mathbf{x}^* \leftarrow \text{LocalSearch}\left(\mathcal{D}(\hat{g}_\theta(G,\mathbf{x}),G)\right)$
for $i = 2$ to I_G **do**
  $K \sim \mathcal{U}(K_{\min},\dots,K_{\max})$; $\mathcal{H} = \hat{g}_\theta\left(G,\mathcal{P}(\mathbf{x}^*,G,K)\right)$
  $\mathbf{x}_0 = \text{LocalSearch}\left(\mathcal{D}(\mathcal{H},G)\right)$
  for $j = 1$ to I_L **do**
    $Z \sim \mathcal{U}(Z_{\min},\dots,Z_{\max})$; $\mathbf{y} \sim \mathcal{Y}(\mathcal{H},\mathbf{x}^*,Z)$
    $\mathcal{G} = \mathcal{R}(G,\mathbf{x}^*,\mathbf{y})$; $\xi \sim \hat{l}_\theta(\mathcal{G})$
    $\mathbf{x}_j = \text{LocalSearch}\left(\text{Expand}\left(\mathcal{S}(\xi,\mathcal{G})\right)\right)$
  end for
  $\mathbf{x}^* \leftarrow \text{argmin}\, c(\mathbf{x},G), \mathbf{x} \in \{\mathbf{x}^*,\mathbf{x}_0,\cdots,\mathbf{x}_{I_L}\}$
end for
Output: Solution $\mathbf{x}^*$

Training and Data Augmentation. PRF and SymNCO are trained separately on the same labeled instances. For SymNCO, we sample a residual size Z, apply progressive reduction with random retention to the labeled solution S^*, and optimize the original RL objective on the resulting LinkGraph. To alleviate limited training data for PRF, we employ geometry-preserving augmentation. Specifically, TSP coordinates are centered at zero, while CVRP and CVRPTW translate the depot to the origin, followed by random rotation, random reflection, and min-max rescaling to the unit square. Since these transformations preserve node indices, the corresponding solution labels remain unchanged.

5 Experiments

5.1 Experimental Setup

Datasets. 1) Synthetic Datasets. We follow the evaluation protocol of the ML4CO-Bench-101 (Ma et al., 2025b). Node coordinates for TSP, CVRP, and CVRPTW are sampled uniformly in the unit square. For TSP, we additionally use the Gaussian and Cluster distributions for the generalization study. For CVRP and CVRPTW, customer demands are drawn uniformly from $[1, 10]$, and vehicle capacities are set to 40, 50, 80, 100, 200, and 300 for instance sizes 50, 100, 200, 500, 1K, and 2K, respectively. Time-window constraints for CVRPTW follow RouteFinder (Berto et al., 2025), detailed in Appendix B. **2) Real-world Datasets.** For TSP, we use 49 instances from TSPLIB[1] with problem sizes ranging from 51 to 1002 nodes. For CVRP, we evaluate on CVRPLIB[2], using 70 instances from the X-set benchmark (Uchoa et al., 2017) ranging from 101 to 513 nodes.

Metrics. 1) Objective: The average objective of the solutions w.r.t. the corresponding instances. **2) Drop**: The relative performance drop w.r.t. the objective compared to the solutions obtained by the baseline exact solver or heuristic solver. **3) Time**: The average computational time per instance.

Training and Solving Settings. 1) Both PRF and SymNCO adopt GAT-based architectures; architectural details are provided in Appendix C, and hyperparameters and training settings in Appendix E. **2)** At solving stage, we consider two configurations: ***fewer*** and ***more***, prioritizing efficiency and solution quality, respectively; their search budgets and hyperparameters are provided in Appendix E.

Comparable Solvers. 1) The compared solvers and their settings are detailed in Appendix D. For some methods, we use two hyperparameter configurations to match our *fewer* and *more* solving budgets. **2)** Following the testing protocol of Ma et al. (2025b) for fair comparison, ***all main-table evaluations use a batch size of 1.*** In practice, COReformer ***supports batched GPU inference for parallel solving***, and we report the corresponding comparisons in Table 8. **3)** For TSP-100 to TSP-1K, we additionally report results with 8 parallel runs, following prior works (Li et al., 2024b; 2025).

[1] http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/
[2] http://vrp.atd-lab.inf.puc-rio.br/media/com_vrp/instances

Table 1: Results on TSP.

METHOD	TYPE	TSP-[UNIFORM-100]			TSP-[UNIFORM-500]			TSP-[UNIFORM-1K]		
		OBJ.↓	DROP↓	TIME↓	OBJ.↓	DROP↓	TIME↓	OBJ.↓	DROP↓	TIME↓
Concorde	Exact	7.76*	0.00%	0.24s	16.55*	0.00%	18.67s	23.12*	0.00%	84.41s
SymNCO + 2OPT	L2C-LC	7.84	1.09%	0.10s	–	–	–	–	–	–
GOAL + 2OPT	L2C-LC	7.86	1.37%	0.76s	17.05	3.03%	4.06s	23.81	2.97%	8.07s
GCN + 2OPT	L2C-GP	7.78	0.24%	0.01s	16.77	1.35%	0.06s	23.53	1.77%	0.23s
Fast-T2T + 2OPT	L2C-GP	7.76	0.08%	0.03s	16.69	0.87%	0.10s	23.42	1.32%	0.33s
COExpander + 2OPT	L2C-AE	7.76	0.06%	0.10s	16.63	0.49%	0.24s	23.34	0.95%	0.70s
LEHD (Greedy)	L2C-LC	7.80	0.55%	0.30s	16.80	1.53%	1.50s	23.84	3.12%	3.09s
SIL (Greedy)	L2C-LC	–	–	–	–	–	–	23.57	1.95%	5.20s
DIMES + MCTS	L2C-GP	7.90	1.82%	0.03s	17.78	7.44%	0.83s	25.03	8.28%	3.87s
GCN + MCTS	L2C-GP	7.76	0.07%	0.01s	16.66	0.68%	0.37s	23.38	1.13%	1.59s
GLOP (fewer)	D&C	7.77	0.16%	4.67s	17.08	3.20%	4.82s	24.01	3.85%	4.76s
GLOP (more)	D&C	7.76	0.05%	4.72s	16.91	2.20%	4.89s	23.84	3.11%	5.18s
UDC (fewer)	D&C	7.80	0.59%	0.10s	16.97	2.57%	0.16s	23.80	2.96%	0.24s
UDC (more)	D&C	7.78	0.32%	2.10s	16.82	1.67%	2.33s	23.53	1.77%	2.83s
NeuOpt (fewer)	L2S	7.81	0.67%	13.23s	–	–	–	–	–	–
NeuOpt (more)	L2S	7.78	0.34%	26.42s	–	–	–	–	–	–
LEHD (RRC) (fewer)	L2S	7.76	0.10%	3.30s	16.65	0.65%	15.91s	23.51	1.70%	31.71s
LEHD (RRC) (more)	L2S	7.76	0.03%	8.25s	16.62	0.42%	39.47s	23.45	1.42%	75.69s
SIL (PRC) (fewer)	L2S	–	–	–	–	–	–	23.40	1.20%	28.08s
SIL (PRC) (more)	L2S	–	–	–	–	–	–	23.33	0.92%	57.43s
COReformer (fewer)	L2S	7.76	0.02%	0.04s	16.56	0.10%	0.13s	23.16	0.20%	0.51s
COReformer (more)	L2S	**7.76**	**0.01%**	0.12s	**16.55**	**0.05%**	0.48s	**23.14**	**0.11%**	1.86s
Fast-T2T + 2OPT (×8)	L2C-GP	7.76	0.01%	0.12s	16.61	0.37%	0.59s	23.31	0.84%	2.47s
COExpander + 2OPT (×8)	L2C-AE	7.76	0.01%	0.33s	16.58	0.18%	1.27s	23.24	0.52%	5.10s
COReformer (fewer) (×8)	L2S	7.76	0.00%	0.04s	16.55	0.03%	0.20s	23.14	0.09%	1.20s
COReformer (more) (×8)	L2S	**7.76**	**0.00%**	0.14s	**16.55**	**0.01%**	0.76s	**23.13**	**0.05%**	4.52s

METHOD	TYPE	TSP-[UNIFORM-2K]			TSP-[UNIFORM-5K]			TSP-[UNIFORM-10K]		
		OBJ.↓	DROP↓	TIME↓	OBJ.↓	DROP↓	TIME↓	OBJ.↓	DROP↓	TIME↓
LKH (500)	Heuristic	32.44*	0.00%	7.14s	50.97*	0.00%	34.49s	71.78*	0.00%	332.76s
SIL (Greedy)	L2C-LC	33.16	2.24%	10.38s	52.58	3.16%	26.12s	74.69	4.05%	53.41s
GCN + 2OPT	L2C-GP	33.03	1.83%	0.95s	51.93	1.89%	7.33s	73.07	1.80%	31.65s
Fast-T2T + 2OPT	L2C-GP	33.00	1.75%	1.10s	51.80	1.65%	7.70s	73.03	1.74%	32.17s
COExpander + 2OPT	L2C-AE	32.91	1.45%	1.45s	51.67	1.38%	8.39s	72.86	1.50%	33.69s
GLOP (fewer)	D&C	33.85	4.35%	4.95s	53.61	5.17%	3.29s	75.61	5.34%	3.91s
GLOP (more)	D&C	33.70	3.90%	5.73s	53.37	4.71%	9.51s	75.29	4.89%	10.59s
UDC (fewer)	D&C	33.91	4.54%	0.52s	55.28	8.46%	2.30s	80.29	11.85%	12.16s
UDC (more)	D&C	33.23	2.45%	5.30s	53.76	5.48%	14.09s	77.70	8.25%	35.23s
SIL (PRC) (fewer)	L2S	33.01	1.75%	28.17s	52.33	2.67%	28.38s	73.94	3.01%	28.07s
SIL (PRC) (more)	L2S	32.88	1.35%	61.33s	52.16	2.34%	52.89s	73.64	2.59%	51.18s
COReformer (fewer)	L2S	32.60	0.51%	1.14s	51.46	0.96%	4.68s	72.68	1.25%	12.63s
COReformer (more)	L2S	**32.53**	**0.29%**	4.84s	**51.32**	**0.68%**	17.59s	**72.56**	**1.09%**	37.03s

Table 2: Results on CVRPTW.

METHOD	TYPE	CVRPTW-[UNIFORM-50]			CVRPTW-[UNIFORM-100]			CVRPTW-[UNIFORM-200]		
		OBJ.↓	DROP↓	TIME↓	OBJ.↓	DROP↓	TIME↓	OBJ.↓	DROP↓	TIME↓
PyVRP	Heuristic	16.04*	0.00%	10.01s	25.43*	0.00%	20.04s	41.60*	0.00%	60.02s
SymNCO	L2C-LC	17.04	6.29%	0.08s	27.60	8.52%	0.16s	46.00	10.55%	0.30s
MVMoE	L2C-LC	16.92	5.49%	0.16s	28.68	12.67%	0.30s	–	–	–
RF-TE	L2C-LC	16.36	2.03%	0.14s	26.24	3.15%	0.27s	–	–	–
CaDA (×8)	L2C-LC	16.28	1.50%	0.17s	26.07	2.51%	0.32s	–	–	–
MoSES (CaDA)	L2C-LC	16.26	1.38%	0.36s	26.03	2.34%	0.58s	–	–	–
COReformer (fewer)	L2S	16.13	0.55%	0.32s	25.77	1.32%	0.54s	42.70	2.59%	1.09s
COReformer (more)	L2S	**16.08**	**0.26%**	1.32s	**25.63**	**0.76%**	2.29s	**42.52**	**2.15%**	4.54s

5.2 EXPERIMENTAL RESULTS

Overview. The main results for the three COPs are presented in Table 1 (TSP), Table 2 (CVRPTW), and Table 3 (CVRP). More comprehensive results are provided in Appendix E, including Table 9 (TSP) and Table 10 (CVRP & CVRPTW). ***A unified comparison with previous L2C, L2S, and D&C SOTA solvers under fair settings is summarized in Tables 13, 14, and 15.***

Results on TSP. 1) Compared with previous L2C SOTA, COReformer (*more*) reduces the mean drop by **58.0%**, while *fewer* retains a **43.8%** reduction with a **2.5**× speedup. Note that on TSP-100 to TSP-1K we set I_L=0 and omit SymNCO, since PRF alone already learns well on TSP, whose constraints are relatively simple. **2)** Against previous L2S SOTA, COReformer (*more*) reduces the mean drop by **70.8%** and is **4.4**× faster. **3)** Relative to previous D&C SOTA, COReformer (*more*)

Table 3: Results on CVRP.

METHOD	TYPE	CVRP-[UNIFORM-50]			CVRP-[UNIFORM-100]		
		OBJ.↓	DROP↓	TIME↓	OBJ.↓	DROP↓	TIME↓
HGS	Heuristic	10.37*	0.00%	1.01s	15.56*	0.00%	20.03s
SymNCO + LS	L2C-LC	10.56	1.91%	0.09s	15.93	2.38%	0.17s
GOAL + LS	L2C-LC	10.63	2.52%	0.51s	15.96	2.55%	0.97s
LEHD (Greedy)	L2C-LC	10.89	5.01%	0.20s	16.22	4.20%	0.40s
Fast-T2T + LS	L2C-GP	10.87	4.84%	0.01s	16.29	4.70%	0.02s
COExpander + LS	L2C-AE	10.77	3.90%	0.04s	16.22	4.25%	0.06s
NeuOpt (fewer)	L2S	10.49	1.21%	16.26s	16.00	2.83%	27.31s
NeuOpt (more)	L2S	10.44	0.70%	32.49s	15.87	1.94%	54.56s
LEHD (RRC) (fewer)	L2S	10.52	1.44%	2.12s	15.81	1.56%	4.85s
LEHD (RRC) (more)	L2S	10.45	0.79%	5.17s	15.73	1.06%	10.98s
COReformer (fewer)	L2S	10.44	0.71%	0.20s	15.72	1.03%	0.31s
COReformer (more)	L2S	**10.42**	**0.47%**	0.80s	**15.67**	**0.68%**	1.19s
METHOD	TYPE	CVRP-[UNIFORM-200]			CVRP-[UNIFORM-500]		
		OBJ.↓	DROP↓	TIME↓	OBJ.↓	DROP↓	TIME↓
HGS	Heuristic	19.63*	0.00%	60.02s	37.15*	0.00%	360.38s
SymNCO + LS	L2C-LC	20.19	2.88%	0.34s	38.70	4.17%	0.88s
GOAL + LS	L2C-LC	20.40	3.94%	2.04s	38.12	2.62%	7.31s
LEHD (Greedy)	L2C-LC	20.46	4.20%	0.77s	38.99	4.94%	1.92s
Fast-T2T + LS	L2C-GP	20.66	5.29%	0.06s	39.20	5.53%	0.22s
COExpander + LS	L2C-AE	20.59	4.89%	0.15s	39.12	5.34%	0.61s
UDC (fewer)	D&C	–	–	–	39.02	5.04%	8.29s
UDC (more)	D&C	–	–	–	38.63	4.00%	39.90s
LEHD (RRC) (fewer)	L2S	20.03	2.04%	9.19s	38.38	3.32%	24.50s
LEHD (RRC) (more)	L2S	19.93	1.53%	19.90s	38.21	2.85%	49.37s
COReformer (fewer)	L2S	19.85	1.13%	0.77s	37.84	1.85%	2.33s
COReformer (more)	L2S	**19.77**	**0.69%**	3.48s	**37.63**	**1.27%**	10.24s
METHOD	TYPE	CVRP-[UNIFORM-1K]			CVRP-[UNIFORM-2K]		
		OBJ.↓	DROP↓	TIME↓	OBJ.↓	DROP↓	TIME↓
HGS	Heuristic	41.17*	0.00%	360.05s	57.18*	0.00%	360.30s
LEHD (Greedy)	L2C-LC	43.96	6.80%	3.90s	61.59	7.79%	9.93s
SIL (Greedy)	L2C-LC	44.08	6.98%	6.46s	59.62	4.20%	13.08s
GLOP (fewer)	D&C	47.06	14.31%	0.47s	63.60	11.12%	1.22s
GLOP (more)	D&C	46.91	13.93%	0.65s	63.19	10.39%	1.83s
UDC (fewer)	D&C	43.61	5.87%	8.91s	62.51	9.08%	11.48s
UDC (more)	D&C	43.04	4.53%	40.79s	60.86	6.30%	52.15s
LEHD (RRC) (fewer)	L2S	43.42	5.47%	37.82s	60.78	6.37%	96.96s
LEHD (RRC) (more)	L2S	43.16	4.83%	90.14s	60.48	5.84%	213.48s
SIL (PRC) (fewer)	L2S	43.65	5.93%	25.66s	59.56	4.10%	26.79s
SIL (PRC) (more)	L2S	43.46	5.48%	51.67s	59.21	3.48%	56.64s
COReformer (fewer)	L2S	43.08	4.63%	2.10s	59.75	4.55%	4.11s
COReformer (more)	L2S	**42.58**	**3.43%**	8.42s	**59.12**	**3.46%**	15.30s

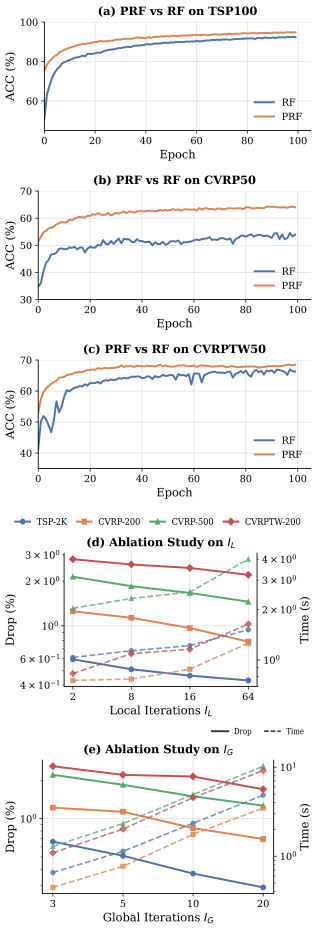

Figure 2: Ablation study.

reduces the mean drop by **85.6%**; even the *fewer* setting already surpasses both GLOP (*more*) and UDC (*more*) in solution quality.

Results on CVRP. 1) Compared with previous L2C SOTA, COReformer (*more*) reduces the mean drop by **51.9%**, while *fewer* retains a **33.1%** reduction. **2)** Against previous L2S SOTA, COReformer (*more*) reduces the mean drop by **30.8%** and is **6.6**× faster. **3)** On CVRP-500 to CVRP-2K, COReformer (*more*) reduces the mean drop of UDC by **45.0%** with a **3.9**× speedup.

Results on CVRPTW. Compared with previous L2C SOTA, COReformer (*more*) reduces the mean drop by **77.9%** on CVRPTW-50 to CVRPTW-200, while *fewer* retains a **68.8%** reduction.

5.3 ABLATION STUDY AND GENERALIZATION STUDY

Ablation Study on PRF. We isolate the effect of PRF by comparing it with vanilla RF (Liu et al., 2022) under identical training settings. At each of the 100 training epochs, we evaluate the corresponding checkpoint on the held-out test set. Here, heatmap accuracy denotes the average score assigned to ground-truth solution entries after clamping the predicted heatmap to $(0, 1)$, rather than standard binary edge-classification accuracy or downstream tour cost. As shown in Fig. 2, PRF consistently achieves higher accuracy throughout training, demonstrating the benefit of non-degenerate perturbed source and target distributions for learning effective retention heatmaps.

Ablation Study on I_G and I_L. Table 7 in Appendix E studies the two search budgets on TSP-2K, CVRP-200, CVRP-500, and CVRPTW-200, with Fig. 2 visualizing the trade-offs between drop and time. The upper block fixes the default *fewer* global budget and varies $I_L \in \{2, 8, 16, 64\}$, while the lower block fixes $I_L = 8$ and varies $I_G \in \{3, 5, 10, 20\}$. Increasing either budget generally

Table 4: Results on MCut.

METHOD	TYPE	MCUT-[BA-2K-3K]		
		OBJ.↑	DROP↓	TIME↓
Gurobi (3600s) (Gurobi Optimization, 2023)	Exact	7217.900*	0.000%	3601.342s
Meta-EGN (Wang & Li, 2023)	L2C-GP	6712.120	6.994%	0.535s
GCN (Joshi et al., 2019)	L2C-GP	6979.260	3.291%	0.060s
VAG-CO (Sanokowski et al., 2023)	L2C-GP	7124.880	1.287%	0.059s
Fast-T2T (Li et al., 2024b)	L2C-GP	7308.260	-1.258%	0.700s
COExpander (Ma et al., 2025a)	L2C-AE	7369.980	-2.111%	0.720s
DiffUCO (Sanokowski et al., 2024)	L2C-GP	7384.020	-2.306%	2.480s
RLD (Feng & Yang, 2025)	Heuristic	7386.320	-2.332%	1.121s
RLD (more) (Feng & Yang, 2025)	Heuristic	7392.580	-2.421%	5.574s
RS-RLD ($I_{RS} = 10$)	Heuristic	7390.640	-2.393%	1.887s
RS-RLD ($I_{RS} = 20$)	Heuristic	7392.040	-2.413%	2.666s
RS-RLD ($I_{RS} = 50$)	Heuristic	**7393.340**	**-2.440%**	**4.854s**

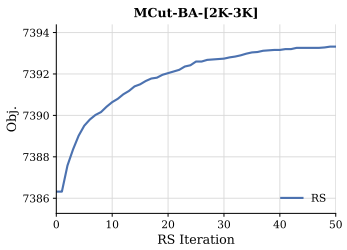

Figure 3: Mean objective of the RS iterative search on MCut.

improves the objective, but increasing I_G incurs substantially higher runtime, as each global iteration requires a new PRF prediction and residual solving, whereas additional local iterations reuse the current heatmap. Importantly, we do not cherry-pick (I_G, I_L) for individual datasets. Instead, we use $I_L = 0$ on TSP-100 to TSP-1K and a unified $I_L = 8$ elsewhere, and only adjust I_G between the *fewer* and *more* settings to keep the overall solving time comparable to competing NCO methods.

Generalization Study. We evaluate Uniform-trained models on Gaussian and Cluster TSP. As shown in Table 9, compared to M^{2}GenCO (Pan et al., 2026a), COReformer reduces the drop by **86.5%** and **85.1%** on average. On TSPLIB and CVRPLIB, Tables 11 and 12 further show that COReformer reduces the average drop by **74.2%** over COExpander and **80.6%** over SymNCO, respectively.

6 EXTENDED DISCUSSION AND CONCLUSION

Generalized RS. The main experiments instantiate the NCO form of RS as COReformer, combining a learning-based global predictor (PRF) and a local constructor (SymNCO) for routing problems. The generality of RS stems from its reduction mechanism, rather than any particular choice of these modules. At each iteration, RS retains part of the incumbent and reduces the original COP to a smaller residual task within the same problem family. The resulting task is typically a variant of the original problem: frozen assignments induce additional constraints and modify the effective costs, rather than simply producing a smaller instance of the same form. Euclidean TSP provides a concrete example. Retaining tour fragments contracts them into nodes in the LinkGraph, yielding a residual TSP over the unfixed nodes and fragment endpoints, with costs defined by the induced distance matrix rather than the original node coordinates. The same reduction principle extends to other COPs, while the specific reduction and solving procedures can be tailored to the problem at hand.

Learning-Free RS Solver for MCut. To demonstrate RS beyond learned routing, we apply it to Maximum Cut (MCut, defined in Appendix B) using Regularized Langevin Dynamics (RLD) (Feng & Yang, 2025) as a classical heuristic, without any learned predictor or constructor. Given an incumbent partition, each iteration fixes a subset of vertex labels and reduces the instance to a residual MCut over the remaining vertices, with edge-induced assignment biases from the fixed set. The residual solution is then expanded to the original graph and accepted if it improves the cut. We denote this learning-free variant by RS-RLD and evaluate it with $I_{RS} \in \{10, 20, 50\}$. The RLD hyperparameters are provided in Appendix E. Following ML4CO-Bench-101 (Ma et al., 2025b), we use its MCut-[BA-2K-3K] dataset, which consists of BA graphs (Barabási & Albert, 1999) with 2K–3K vertices, and compare against the benchmark baselines. Table 4 and Fig. 3 show that the cut progressively improves as the number of RS iterations increases, both across different search budgets and throughout the search trajectory, demonstrating the effectiveness of the RS framework even without learned components.

Conclusion and Future Work. We proposed Reduce-then-Solve (RS), a general framework that improves incumbents through reduction-based residual re-optimization. We instantiate its NCO form as COReformer by combining PRF, LinkGraph-based reduction, and SymNCO, achieving strong performance on TSP, CVRP, and CVRPTW. A learning-free instantiation on MCut further demonstrates that RS is not restricted to neural solvers and can flexibly incorporate classical heuristics. More generally, RS provides a flexible reduction-based paradigm in which different prediction and solving methods can be incorporated according to the structure of the target COP. Future work will explore broader applications of RS to diverse COPs, more efficient residual solvers to accelerate inference, and stronger global predictors for more effective reduction guidance.

REFERENCES

David Applegate and William Cook. Concorde tsp solver, 2006.

Yunpeng Ba, Xi Lin, Changliang Zhou, Ruihao Zheng, Zhenkun Wang, Xinyan Liang, Zhichao Lu, Jianyong Sun, Yuhua Qian, and Qingfu Zhang. Survey on neural routing solvers. *arXiv preprint arXiv:2602.21761*, 2026.

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286 (5439):509–512, 1999.

Federico Berto, Chuanbo Hua, Junyoung Park, Minsu Kim, Hyeonah Kim, Jiwoo Son, Haeyeon Kim, Joungho Kim, and Jinkyoo Park. RL4CO: a unified reinforcement learning for combinatorial optimization library. In *Neural Information Processing Systems 2023, Workshop: New Frontiers in Graph Learning*, 2023.

Federico Berto, Chuanbo Hua, Nayeli Gast Zepeda, André Hottung, Niels Wouda, Leon Lan, Junyoung Park, Kevin Tierney, and Jinkyoo Park. RouteFinder: Towards Foundation Models for Vehicle Routing Problems. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856.

Paulo R d O Costa, Jason Rhuggenaath, Yingqian Zhang, and Alp Akcay. Learning 2-opt heuristics for the traveling salesman problem via deep reinforcement learning. In *Asian conference on machine learning*, pp. 465–480. PMLR, 2020.

Darko Drakulic, Sofia Michel, Florian Mai, Arnaud Sors, and Jean-Marc Andreoli. Bq-nco: Bisimulation quotienting for efficient neural combinatorial optimization. In *Advances in Neural Information Processing Systems*, 2023.

Darko Drakulic, Sofia Michel, and Jean-Marc Andreoli. GOAL: A generalist combinatorial optimization agent learner. In *The Thirteenth International Conference on Learning Representations*, 2025.

Shengyu Feng and Yiming Yang. Regularized langevin dynamics for combinatorial optimization. *arXiv preprint arXiv:2502.00277*, 2025.

Zhang-Hua Fu, Kai-Bin Qiu, and Hongyuan Zha. Generalize a small pre-trained model to arbitrarily large tsp instances. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 7474–7482, 2021.

Fred Glover. Tabu searchâĂŤpart i. *ORSA Journal on computing*, 1(3):190–206, 1989.

LLC Gurobi Optimization. Gurobi optimizer reference manual (2020), 2023.

Keld Helsgaun. An extension of the lin-kernighan-helsgaun tsp solver for constrained traveling salesman and vehicle routing problems. *Roskilde: Roskilde University*, 12, 2017.

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 *The Eleventh International Conference on Learning Representations*, 2023.

Chaitanya K Joshi, Thomas Laurent, and Xavier Bresson. An efficient graph convolutional network technique for the travelling salesman problem. *arXiv preprint arXiv:1906.01227*, 2019.

Minsu Kim, Junyoung Park, and Jinkyoo Park. Sym-nco: Leveraging symmetricity for neural combinatorial optimization. *Advances in Neural Information Processing Systems*, 35:1936–1949, 2022.

Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! In *International Conference on Learning Representations*, 2019.

Yeong-Dae Kwon, Jinho Choo, Byoungjip Kim, Iljoo Yoon, Youngjune Gwon, and Seungjai Min. Pomo: Policy optimization with multiple optima for reinforcement learning. *Advances in Neural Information Processing Systems*, 33:21188–21198, 2020.

Han Li, Fei Liu, Zhi Zheng, Yu Zhang, and Zhenkun Wang. Cada: Cross-problem routing solver with constraint-aware dual-attention. *arXiv preprint arXiv:2412.00346*, 2024a.

Yang Li, Jinpei Guo, Runzhong Wang, and Junchi Yan. From distribution learning in training to gradient search in testing for combinatorial optimization. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Yang Li, Jinpei Guo, Runzhong Wang, Hongyuan Zha, and Junchi Yan. Fast t2t: Optimization consistency speeds up diffusion-based training-to-testing solving for combinatorial optimization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024b.

Yang Li, Lvda Chen, Haonan Wang, Runzhong Wang, and Junchi Yan. Generation as search operator for test-time scaling of diffusion-based combinatorial optimization. In *Advances in Neural Information Processing Systems*, 2025.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

Fu Luo, Xi Lin, Fei Liu, Qingfu Zhang, and Zhenkun Wang. Neural combinatorial optimization with heavy decoder: Toward large scale generalization. *Advances in Neural Information Processing Systems*, 36:8845–8864, 2023.

Fu Luo, Xi Lin, Yaoxin Wu, Zhenkun Wang, Tong Xialiang, Mingxuan Yuan, and Qingfu Zhang. Boosting neural combinatorial optimization for large-scale vehicle routing problems. In *International Conference on Learning Representations*, volume 2025, pp. 14754–14780, 2025.

Jiale Ma, Wenzheng Pan, Yang Li, and Junchi Yan. Coexpander: Adaptive solution expansion for combinatorial optimization. In *Forty-second International Conference on Machine Learning*, 2025a.

Jiale Ma, Wenzheng Pan, Yang Li, and Junchi Yan. Ml4co-bench-101: Benchmark machine learning for classic combinatorial problems on graphs. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2025b.

Yining Ma, Zhiguang Cao, and Yeow Meng Chee. Learning to search feasible and infeasible regions of routing problems with flexible neural k-opt. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Yimeng Min, Yiwei Bai, and Carla P. Gomes. Unsupervised learning for solving the travelling salesman problem. In *Advances in Neural Information Processing Systems*, 2023.

Yuichi Nagata and Shigenobu Kobayashi. A powerful genetic algorithm using edge assembly crossover for the traveling salesman problem. *INFORMS Journal on Computing*, 25(2):346–363, 2013.

Wenzheng Pan, Hao Xiong, Jiale Ma, Wentao Zhao, Yang Li, and Junchi Yan. UniCO: On unified combinatorial optimization via problem reduction to matrix-encoded general TSP. In *The Thirteenth International Conference on Learning Representations*, 2025.

Wenzheng Pan, Jiale Ma, Nuoyan Chen, Yang Li, and Junchi Yan. Problem distributions as tasks: Repurposing meta learning for generative combinatorial optimization towards multi-task pretraining and adaptation. In *Forty-third International Conference on Machine Learning*, 2026a.

Yuxin Pan, Zhiguang Cao, Chengyang Gu, Liu Liu, Peilin Zhao, Yize Chen, and Fangzhen Lin. Multi-task vehicle routing solver via mixture of specialized experts under state-decomposable mdp. *Advances in Neural Information Processing Systems*, 38:32577–32620, 2026b.

Christian Prins. A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & operations research*, 31(12):1985–2002, 2004.

Ruizhong Qiu, Zhiqing Sun, and Yiming Yang. DIMES: A differentiable meta solver for combinatorial optimization problems. In *Advances in Neural Information Processing Systems 35*, 2022.

Sebastian Sanokowski, Wilhelm Berghammer, Sepp Hochreiter, and Sebastian Lehner. Variational annealing on graphs for combinatorial optimization. *Advances in Neural Information Processing Systems*, 36:63907–63930, 2023.

Sebastian Sanokowski, Sepp Hochreiter, and Sebastian Lehner. A diffusion model framework for unsupervised neural combinatorial optimization. In *Forty-first International Conference on Machine Learning*, 2024.

Paul Shaw. Using constraint programming and local search methods to solve vehicle routing problems. In *International conference on principles and practice of constraint programming*, pp. 417–431. Springer, 1998.

Zhiqing Sun and Yiming Yang. DIFUSCO: Graph-based diffusion solvers for combinatorial optimization. In *Thirty-seventh Conference on Neural Information Processing Systems*, 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.

Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau, Nadia Lahrichi, and Walter Rei. A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. *Operations Research*, 60 (3):611–624, 2012.

Haoyu Peter Wang and Pan Li. Unsupervised learning for combinatorial optimization needs meta learning. In *The Eleventh International Conference on Learning Representations*, 2023.

Runzhong Wang, Ziao Guo, Wenzheng Pan, Jiale Ma, Yikai Zhang, Nan Yang, Qi Liu, Longxuan Wei, Hanxue Zhang, Chang Liu, Zetian Jiang, Xiaokang Yang, and Junchi Yan. Pygmtools: A python graph matching toolkit. *Journal of Machine Learning Research*, 25(33):1–7, 2024.

Yu Wang, Yang Li, Jiale Ma, Junchi Yan, and Yi Chang. Nexco: Native solution expansion for diffusion-based combinatorial optimization. In *The Fourteenth International Conference on Learning Representations*, 2026.

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. URL https://doi.org/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*, 2024.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in neural information processing systems*, 32, 2019.

Zhi Zheng, Changliang Zhou, Tong Xialiang, Mingxuan Yuan, and Zhenkun Wang. Udc: A unified neural divide-and-conquer framework for large-scale combinatorial optimization problems. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Jianan Zhou, Zhiguang Cao, Yaoxin Wu, Wen Song, Yining Ma, Jie Zhang, and Chi Xu. Mvmoe: Multi-task vehicle routing solver with mixture-of-experts. *arXiv preprint arXiv:2405.01029*, 2024.

APPENDIX

A STATEMENTS AND LIMITATIONS

AI Use Statement. Generative AI tools were used only to enhance the readability and linguistic presentation of author-written text. They were not used to develop the methodology, formulate mathematical claims, design experiments, or generate and process data. All suggested edits were manually reviewed by the authors, who assume full responsibility for the final submission.

Ethics Statement. This work involves no human participants, personally identifiable information, or sensitive private data. All datasets, software, and experimental procedures comply with their respective licensing agreements and research policies. To the best of the authors' knowledge, this work poses no additional ethical concerns.

Limitations. 1) The performance of COReformer depends on both PRF and SymNCO. When either module is relatively weak, the overall search quality degrades accordingly, which is reflected in the stronger results on TSP compared with CVRP and CVRPTW. We view this as a limitation of the current instantiation rather than of RS itself: as NCO continues to advance, stronger global predictors and local constructors can be substituted into the framework without changing the reduce-then-solve (RS) procedure. **2)** Inference of the local constructor remains relatively slow, which we attribute mainly to the sequential decoding bottleneck of the RL4CO (Berto et al., 2023) implementation. Targeted acceleration of LC inference is a promising direction for future work. **3)** The current instantiation still relies on supervised solutions to train PRF and therefore cannot be trained in a fully unsupervised manner, which may restrict settings where high-quality labels are expensive to obtain. Nevertheless, the synthetic labeled instances are generated once and can be reused by the broader NCO community, and data augmentation can further reduce the amount of labeled data required.

B PROBLEM DEFINITIONS

Following Sec. 3, we consider the three COPs defined on a graph $G = (V, E)$ with $N = |V|$. Formal definitions of the three routing problems studied in this paper are given below.

Traveling Salesman Problem (TSP). Given G and a cost matrix $C \in \mathbb{R}^{N\times N}$, where C_{ij} denotes the cost of edge (i, j), the objective is to find a Hamiltonian tour $\pi = (i_1, \cdots, i_N)$ that minimizes the total cost: $\sum_{k=1}^{N-1} C_{i_k i_{k+1}} + C_{i_N i_1}$.

Capacitated Vehicle Routing Problem (CVRP). Given G, a depot node $v_0 \in V$, a cost matrix $C \in \mathbb{R}^{N\times N}$, a demand vector $d \in \mathbb{R}^N_+$, and a vehicle capacity $Q > 0$, the goal is to plan a set of routes Π, each route $\pi \in \Pi$ starting and ending at the depot v_0, such that each customer node is visited exactly once and the total demand on each route does not exceed Q, i.e., $\sum_{i\in\pi} d_i \le Q$. The objective is to minimize the total cost of all routes: $\min_{\Pi} \sum_{\pi\in\Pi} \sum_{(i,j)\in\pi} C_{ij}$.

Capacitated Vehicle Routing Problem with Time Windows (CVRPTW). CVRPTW extends CVRP by further associating each node $v_i \in V$ with a service time $p_i \ge 0$ and a time window $[w_i^{\text{beg}}, w_i^{\text{end}}]$. In addition to the capacity and routing constraints of CVRP, each route must satisfy the corresponding temporal constraints: if a vehicle arrives at node v_i at time α_i, it starts service at $\beta_i = \max(\alpha_i, w_i^{\text{beg}})$ and must satisfy $\beta_i \le w_i^{\text{end}}$; after completing service, it departs at time $\beta_i + p_i$. The objective remains to minimize the total travel cost over all routes: $\min_{\Pi} \sum_{\pi\in\Pi} \sum_{(i,j)\in\pi} C_{ij}$.

Time-Window Generation. Time-window constraints for CVRPTW follow RouteFinder (Berto et al., 2025) and other previous works (Berto et al., 2023; Pan et al., 2026b) and With unit vehicle speed, the depot is assigned a window $[0, T_{\max}]$ with $T_{\max} = 4.6$ and zero service time. For each customer i with depot-to-customer distance d_{0i}, the service time p_i and window length Δ_i are sampled uniformly from $[0.15, 0.18]$ and $[0.18, 0.20]$, respectively. The window start w_i^{beg} is sampled uniformly from $[d_{0i},\ T_{\max} - p_i - \Delta_i - d_{0i}]$, and the window end is set to $w_i^{\text{end}} = w_i^{\text{beg}} + \Delta_i$.

Maximum Cut (MCut). Given G and an adjacency matrix $C \in \mathbb{R}^{N\times N}$, a *cut* $(U, \overline{U})$ partitions the node set V into two disjoint subsets. The objective is to maximize $\sum_{i\in U,\, j\in\overline{U}} C_{ij}$.

C Graph Neural Architecture

Both PRF and SymNCO (Kim et al., 2022) use GAT-based architectures. For SymNCO, we follow the Attention Model implementation in RL4CO (Berto et al., 2023), adapting only the input embeddings to encode LinkGraph features. PRF employs an encoder-decoder GAT to predict a retention heatmap $\mathcal{H}$ conditioned on the instance G and interpolant S_τ. The encoder processes instance features, while the decoder additionally incorporates solution and flow-time information.

Node Embedding. Node features are projected to $\mathbf{h}_v^0 \in \mathbb{R}^h$. For TSP, 2-D coordinates are mapped by a linear layer. For CVRP, coordinates, demand, and a depot indicator are embedded separately, concatenated, and fused by a linear layer. For CVRPTW, service times and time-window bounds are further included in the fused embedding.

GAT Layer. Both the encoder and decoder consist of pre-norm GAT layers, each applying multi-head self-attention (MHSA) followed by a SwiGLU feed-forward block, with residual connections:

$$\mathbf{h}_v^{l+\frac{1}{2}} = \mathbf{h}_v^l + \mathrm{MHSA}\big(\mathrm{LN}\big(\mathbf{h}_v^l\big); B\big), \quad \mathbf{h}_v^{l+1} = \mathbf{h}_v^{l+\frac{1}{2}} + \mathrm{SwiGLU}\Big(\mathrm{LN}\big(\mathbf{h}_v^{l+\frac{1}{2}}\big)\Big), \tag{6}$$

where LN denotes LayerNorm without affine bias. Within each MHSA, queries, keys, and values are obtained via a fused projection $W^{qkv} \in \mathbb{R}^{3h\times h}$ (without bias), and attention logits are stabilized by a tanh softcap before an optional additive bias B:

$$A_{ij} = \delta \cdot \tanh\left(\frac{\mathbf{q}_i^\top \mathbf{k}_j}{\delta\sqrt{d_k}}\right) + B_{ij}, \qquad \mathbf{o}_i = \sum_j \mathrm{Softmax}_j(A_{ij})\,\mathbf{v}_j, \tag{7}$$

where δ is the softcap constant and $d_k = h/N_{\text{heads}}$ is the per-head dimension. The SwiGLU block uses a gated hidden width of $3h$.

Encoder and Decoder. 1) The encoder contains L_{enc} GAT layers with $B = \mathbf{0}$, modeling pairwise interactions over the complete graph without explicit adjacency bias. A linear projection then maps the encoder output to the decoder width. **2)** The decoder consists of L_{dec} GAT layers that condition on the interpolant $S_\tau \in \mathbb{R}^{N\times N}$ and the flow time τ. Specifically, τ is scaled by 1000 and encoded using a sinusoidal embedding with frequency dimension 256, followed by a two-layer SiLU MLP to obtain $\mathbf{h}_\tau \in \mathbb{R}^h$, which is added to all node embeddings. Each decoder GAT layer then uses the interpolant itself as the additive attention bias, $B = S_\tau$, so that the current transport state directly modulates which node pairs receive high attention.

Output Layer. A two-layer SiLU MLP followed by RMSNorm (Zhang & Sennrich, 2019) produces node features $\mathbf{z}_v \in \mathbb{R}^h$. Pairwise logits are computed as $\mathbf{z}_i^\top \mathbf{z}_j/\sqrt{h}$ and bounded by a tanh softcap with constant δ_{out}. After masking self-loops, a row-wise softmax produces the heatmap $\mathcal{H}$, which parameterizes the predicted target state $\hat{S}_1 = f_\theta(S_\tau, \tau, G)$ rather than the signed velocity directly. The velocity used by PRF is induced as $\hat{S}_1 - S_0$, as described in Algorithm 1.

D Comparable Solvers and their Settings

Baseline Solvers. We include exact and classical heuristic solvers as references, specifically Concorde (Applegate & Cook, 2006), GA-EAX (Nagata & Kobayashi, 2013), LKH (Helsgaun, 2017), HGS (Vidal et al., 2012), and PyVRP (Wouda et al., 2024).

L2C Solvers. Methods selected from ML4CO-Bench-101 include COExpander (Ma et al., 2025a), GCN (Joshi et al., 2019), DIFUSCO (Sun & Yang, 2023), Fast-T2T (Li et al., 2024b), DIMES (Qiu et al., 2022), UTSP (Min et al., 2023), SymNCO (Berto et al., 2023), and GOAL (Drakulic et al., 2025). We additionally include the greedy variants of two L2C-LC solvers, LEHD and SIL (Luo et al., 2023; 2025). For CVRPTW, we further compare with MVMoE (Zhou et al., 2024), RF-TE (Berto et al., 2025), CaDA (Li et al., 2024a), and MoSES(CaDA) (Pan et al., 2026b).

L2S and D&C Solvers. For L2S, we consider NeuOpt (Ma et al., 2023), GenSCO (Li et al., 2025), and the reconstruction-enhanced variants LEHD (RRC) and SIL (PRC) (Luo et al., 2023; 2025). For D&C, we consider GLOP (Ye et al., 2024) and UDC (Zheng et al., 2024).

Generalization. We compare with M^{2}GenCO (Pan et al., 2026a) on out-of-distribution instances.

Note on GenSCO. Consistent with a recent survey (Ba et al., 2026), we find that GenSCO generalizes poorly to larger and out-of-distribution TSP instances (Table 9), which we attribute primarily to its model configuration rather than the flow-based paradigm itself. As it uses a Flax/JAX implementation instead of the standard PyTorch pipeline, we omit it from the main tables and instead use it as an RF-style global predictor for a controlled small-scale comparison with PRF (Fig. 2).

Solving Settings. We detail the inference hyperparameters and solving budgets of the compared solvers below. To ensure a consistent comparison with our *fewer* and *more* settings, we use two hyperparameter configurations for some methods when applicable.

1) Fast-T2T and COExpander. Here I_s denotes the number of denoising steps at inference, and D_s is the maximum number of determination steps used by COExpander. We set $D_s = 3$ for COExpander, and $I_s = 5$ on TSP and $I_s = 1$ on CVRP for both methods.

2) GLOP. For TSP, we follow the *fewer* and *more* settings specified in the original paper, and use the TSP-1K and TSP-10K settings for TSP-2K and TSP-5K, respectively. For CVRP, we adopt the original settings as *fewer* and double the revision iterations to obtain *more*. All remaining decoding and partitioning settings are kept unchanged between the two configurations.

3) UDC. We use a subproblem size of 100 and sample $\alpha = 50$ initial solutions throughout, where α denotes the number of candidate solutions rather than data augmentation. For TSP, *fewer* and *more* use $x = 2$ and $x = 50$ refinement iterations, respectively, with an effective POMO size of 2; for CVRP, they use $x = 50$ and $x = 250$, with an effective POMO size of 1 during evaluation. All other inference settings remain unchanged.

4) NeuOpt. We follow the original setting with $\mathrm{D2A} = 1$ and use $T = 1\mathrm{K}$ as *more* and $T = 500$ as *fewer*. All other settings follow the original paper.

5) LEHD and SIL. We follow the official inference procedures. For their reconstruction-enhanced variants, LEHD (RRC) uses 20 and 50 random reconstruction cycles for the *fewer* and *more* settings, while SIL (PRC) uses 10 and 20 parallel reconstruction iterations, respectively.

E Supplementary Experiments and Details

Hardware. All models are trained and tested using NVIDIA H800 (80G) GPUs and an Intel(R) Xeon(R) Platinum 8558 96-Core Processor CPU. For all test evaluations, a single GPU is utilized, and the batch size is set to 1 to ensure a fair comparison of the solving time across different models.

Architectural Hyperparameters. 1) For PRF, we use embedding dimension $h = 256$, $N_{\text{heads}} = 8$ attention heads, $L_{\text{enc}} = 16$ encoder layers, and $L_{\text{dec}} = 6$ decoder layers. The attention softcap is $\delta = 30$ and the output logit softcap is $\delta_{\text{out}} = 50$. **2)** For SymNCO, we follow RL4CO with embedding dimension 128, 8 attention heads, 3 encoder layers, feed-forward hidden size 512, and batch normalization. Both models are trained with a weight decay of 10^{-4}.

Training and Solving Settings. 1) Table 5 summarizes the residual-size range Z, training schedules, and local iterations I_L for SymNCO. For larger instances, models are warm-started from the corresponding smaller-scale checkpoints, as indicated in the Pretrain column. **2)** Table 6 summarizes the perturbation ranges, training schedules, and inference budgets for PRF, where N_τ denotes the number of ODE steps and I_G^f / I_G^m the global iterations under the *fewer* and *more* configurations, respectively. The tabulated k_1 and k_2 are perturbation ratios, distinct from the step counts K_1 and K_2 in the main text; the latter are obtained by $K_1 = k_1 N$ and $K_2 = k_2 N$.

Complete Results. We provide complete results on synthetic TSP (Table 9), CVRP and CVRPTW (Table 10), TSPLIB (Table 11), and CVRPLIB (Table 12). Note that TSPLIB and CVRPLIB are evaluated under the *more* setting.

Learning-free RS on MCut. Both the initial full-graph solve and the subsequent residual solves adopt Regularized Langevin Dynamics (RLD) (Feng & Yang, 2025). Using the original RLD hyperparameters $\{\tau_0, d, K, T\}$, we set $\{2.0, 20, 1000, 1000\}$ for initialization and $\{2.0, 20, 100, 200\}$ for each residual solve. The standalone RLD baseline uses the same initialization setting, while RLD (more) increases the number of steps to $T = 5000$. At every RS iteration, the residual size is sampled uniformly from 60%–80% of the original number of vertices.

Implementation Details of Algorithm 2. For clarity, discrete feasible solutions in Algorithm 2 are denoted by $\mathbf{x}$, corresponding to S in Eq. 5, to avoid visual confusion with the continuous source state S_0 in Algorithm 1. We implement the stochastic progressive reduction $\mathcal{Y}$ by converting the heatmap $\mathcal{H}$ into keep scores on eligible incumbent edges: tour edges for TSP and non-depot route edges for CVRP and CVRPTW. These edges are independently retained according to their keep scores; if the residual size is still larger than the sampled target Z, we repeat the thinning process on the remaining edges with gradually reduced probabilities until the target size or the closest attainable smaller size is reached. The retained consecutive edges form fragments that are contracted into the LinkGraph, with problem-specific node features and feasibility masks handled by the corresponding local constructor. In Algorithm 2, $\mathcal{D}$ decodes the PRF heatmap into a feasible solution before $\mathrm{LocalSearch}$; the same local search is applied after expanding a LinkGraph solution. We follow the problem-specific decoding and local-search settings of ML4CO-Bench-101 (Ma et al., 2025b). **1) TSP.** $\mathcal{D}$ uses greedy heatmap-based decoding followed by Two-Opt, and $\mathcal{P}_K$ applies K random 2-opt moves. **2) CVRP.** $\mathcal{D}$ uses capacity-aware greedy decoding, $\mathrm{LocalSearch}$ follows the benchmark CVRP local search (Prins, 2004), and $\mathcal{P}_K$ applies K random feasible relocate or intra-route 2-opt moves with equal probability. **3) CVRPTW.** Decoding and local search additionally enforce time-window feasibility, while $\mathcal{P}_K$ applies K random time-window-aware relocate or intra-route 2-opt moves, accepting only moves that preserve both capacity and time-window feasibility.

Table 5: SymNCO Settings.

Problem	Dataset	Size	Z	Training				Testing	
				Pretrain	Epoch × Steps	BS	LR	Model	I_L
TSP	Uniform-1K	128	$(50, 200)$	–	100×5000	128	1e-4	–	–
TSP	Uniform-2K	64	$(50, 200)$	Uniform-1K	100×5000	128	1e-4	Uniform-2K	8
TSP	Uniform-5K	32	$(50, 200)$	Uniform-2K	50×2000	128	1e-4	Uniform-5K	8
TSP	Uniform-10K	16	$(50, 200)$	Uniform-5K	50×2000	128	1e-4	Uniform-10K	8
CVRP	Uniform-50	10000	$(40, 50)$	–	100×5000	128	1e-4	Uniform-50	8
CVRP	Uniform-100	10000	$(50, 100)$	Uniform-50	100×5000	128	1e-4	Uniform-100	8
CVRP	Uniform-200	100	$(50, 200)$	Uniform-100	50×5000	128	1e-4	Uniform-200	8
CVRP	Uniform-500	100	$(100, 200)$	Uniform-200	50×2000	128	1e-4	Uniform-500	8
CVRP	Uniform-1K	100	$(100, 200)$	Uniform-500	50×2000	128	1e-4	Uniform-1K	8
CVRP	Uniform-2K	100	$(100, 200)$	Uniform-1K	50×2000	128	1e-4	Uniform-2K	8
CVRPTW	Uniform-50	1000	$(40, 50)$	–	100×5000	128	1e-4	Uniform-50	8
CVRPTW	Uniform-100	1000	$(50, 100)$	Uniform-50	100×5000	128	1e-4	Uniform-100	8
CVRPTW	Uniform-200	1000	$(50, 200)$	Uniform-100	50×5000	128	1e-4	Uniform-200	8

Table 6: PRF Settings.

Problem	Dataset	Size	$\mathcal{P}$		Training				Testing			
			k_1	k_2	Pretrain	Epoch × Steps	BS	LR	Model	N_τ	I_G^f	I_G^m
TSP	Uniform-100	1280	$(0.2, 0.5)$	$(0.6, 0.8)$	–	100×5000	512	1e-4	Uniform-100	1	3	10
TSP	Uniform-500	128	$(0.2, 0.5)$	$(0.6, 0.8)$	Uniform-100	100×5000	128	1e-4	Uniform-500	4	5	20
TSP	Uniform-1K	128	$(0.2, 0.5)$	$(0.6, 0.8)$	Uniform-500	100×5000	64	1e-4	Uniform-1K	4	10	40
TSP	Uniform-2K	64	$(0.2, 0.5)$	$(0.6, 0.8)$	Uniform-1K	50×5000	16	1e-4	Uniform-2K	4	5	20
TSP	Uniform-5K	32	$(0.2, 0.5)$	$(0.6, 0.8)$	Uniform-2K	50×2000	8	1e-4	Uniform-5K	4	5	20
TSP	Uniform-10K	16	$(0.2, 0.5)$	$(0.6, 0.8)$	Uniform-5K	10×2000	2	1e-5	Uniform-10K	4	3	10
TSP	Gaussian-100	1280	$(0.2, 0.5)$	$(0.6, 0.8)$	–	–	–	–	Uniform-100	1	3	10
TSP	Gaussian-200	128	$(0.2, 0.5)$	$(0.6, 0.8)$	–	–	–	–	Uniform-500	1	5	20
TSP	Gaussian-500	128	$(0.2, 0.5)$	$(0.6, 0.8)$	–	–	–	–	Uniform-500	1	5	20
TSP	Cluster-100	1280	$(0.2, 0.5)$	$(0.6, 0.8)$	–	–	–	–	Uniform-100	1	3	10
TSP	Cluster-200	128	$(0.2, 0.5)$	$(0.6, 0.8)$	–	–	–	–	Uniform-500	1	5	20
TSP	Cluster-500	128	$(0.2, 0.5)$	$(0.6, 0.8)$	–	–	–	–	Uniform-500	1	5	20
CVRP	Uniform-50	10000	$(0.2, 0.5)$	$(0.6, 0.8)$	–	100×5000	512	1e-4	Uniform-50	4	3	10
CVRP	Uniform-100	10000	$(0.2, 0.5)$	$(0.6, 0.8)$	Uniform-50	100×5000	512	1e-4	Uniform-100	4	3	10
CVRP	Uniform-200	100	$(0.2, 0.5)$	$(0.6, 0.8)$	Uniform-100	50×5000	256	1e-4	Uniform-200	4	5	20
CVRP	Uniform-500	100	$(0.2, 0.5)$	$(0.6, 0.8)$	Uniform-200	50×5000	128	1e-4	Uniform-500	4	5	20
CVRP	Uniform-1K	100	$(0.2, 0.5)$	$(0.6, 0.8)$	Uniform-500	50×2000	64	1e-4	Uniform-1K	4	3	10
CVRP	Uniform-2K	100	$(0.2, 0.5)$	$(0.6, 0.8)$	Uniform-1K	50×2000	16	1e-4	Uniform-2K	4	3	10
CVRPTW	Uniform-50	1000	$(0.5, 1.0)$	$(1.2, 1.5)$	–	100×5000	512	1e-4	Uniform-50	4	3	10
CVRPTW	Uniform-100	1000	$(0.5, 1.0)$	$(1.2, 1.5)$	Uniform-50	100×5000	512	1e-4	Uniform-100	4	3	10
CVRPTW	Uniform-200	100	$(0.5, 1.0)$	$(1.2, 1.5)$	Uniform-100	50×5000	256	1e-4	Uniform-200	4	3	10

Table 7: Ablation Study on I_G and I_L.

Settings		TSP-[Uniform-2K]			CVRP-[Uniform-200]			CVRP-[Uniform-500]			CVRPTW-[Uniform-200]		
I_G	I_L	Obj.↓	Drop↓	Time↓	Obj.↓	Drop↓	Time↓	Obj.↓	Drop↓	Time↓	Obj.↓	Drop↓	Time↓
fewer	2	32.629	0.596%	1.037s	19.875	1.253%	0.756s	37.948	2.140%	2.035s	42.798	2.803%	0.831s
fewer	8	32.602	0.511%	1.136s	19.852	1.133%	0.770s	37.839	1.848%	2.332s	42.702	2.586%	1.092s
fewer	16	32.586	0.463%	1.220s	19.821	0.967%	0.882s	37.775	1.664%	2.548s	42.619	2.449%	1.161s
fewer	64	32.576	0.430%	1.518s	19.783	0.783%	1.260s	37.694	1.453%	3.997s	42.514	2.203%	1.641s
3	8	32.651	0.660%	0.656s	19.868	1.221%	0.447s	37.977	2.215%	1.290s	42.702	2.586%	1.092s
5	8	32.602	0.511%	1.136s	19.852	1.133%	0.770s	37.839	1.848%	2.332s	42.544	2.213%	2.019s
10	8	32.556	0.370%	2.346s	19.795	0.843%	1.761s	37.712	1.499%	4.933s	42.516	2.147%	4.540s
20	8	32.530	0.288%	4.844s	19.766	0.692%	3.477s	37.627	1.267%	10.239s	42.333	1.704%	9.165s

Table 8: Batched GPU inference. Following Ma et al. (2025b), the main tables use batch size 1 for fair comparison, while COReformer supports batched GPU inference. BS, TOTAL, and TIME denote the batch size, total test-set solving time, and per-batch solving time, respectively.

METHOD	TSP-[UNIFORM-100]				TSP-[UNIFORM-500]				TSP-[UNIFORM-1K]			
	OBJ.↓	DROP↓	TOTAL↓	TIME↓	OBJ.↓	DROP↓	TOTAL↓	TIME↓	OBJ.↓	DROP↓	TOTAL↓	TIME↓
Concorde	7.756*	0.000%	5.077m	0.238s	16.546*	0.000%	39.834m	18.672s	23.118*	0.000%	3.001h	84.413s
SymNCO + 2OPT (BS = 1)	7.840	1.087%	2.133m	0.100s	–	–	–	–	–	–	–	–
SymNCO + 2OPT (BS = 8)	7.843	1.124%	30.799s	0.192s	–	–	–	–	–	–	–	–
SymNCO + 2OPT (BS = 16)	7.842	1.110%	22.035s	0.276s	–	–	–	–	–	–	–	–
SymNCO + 2OPT (BS = 32)	7.840	1.076%	18.041s	0.451s	–	–	–	–	–	–	–	–
SymNCO + 2OPT (BS = 128)	7.840	1.087%	15.156s	1.516s	–	–	–	–	–	–	–	–
GOAL + 2OPT (BS = 1)	7.862	1.369%	16.299m	0.764s	17.047	3.027%	1.442h	4.057s	23.806	2.974%	17.199m	8.065s
GOAL + 2OPT (BS = 16)	7.861	1.356%	70.803s	0.885s	17.022	2.878%	37.312s	4.664s	23.800	2.948%	2.338m	17.538s
GOAL + 2OPT (BS = 128)	7.861	1.360%	45.329s	4.533s	17.033	2.943%	22.108s	22.108s	23.800	2.949%	2.007m	2.007m
GLOP (fewer) (BS = 1)	7.769	0.163%	1.662h	4.674s	17.075	3.198%	10.278m	4.818s	24.009	3.853%	10.153m	4.759s
GLOP (fewer) (BS = 16)	7.768	0.159%	6.562m	4.922s	17.075	3.198%	41.837s	5.230s	24.009	3.853%	45.308s	5.664s
GLOP (fewer) (BS = 128)	7.769	0.164%	3.051m	18.303s	17.075	3.198%	15.182s	15.182s	24.009	3.853%	27.580s	27.580s
GLOP (more) (BS = 1)	7.760	0.052%	1.679h	4.723s	16.909	2.197%	10.436m	4.892s	23.838	3.113%	11.054m	5.182s
GLOP (more) (BS = 16)	7.760	0.048%	14.079m	10.559s	16.909	2.194%	96.216s	12.027s	23.838	3.113%	2.732m	20.492s
GLOP (more) (BS = 128)	7.760	0.047%	10.755m	64.529s	16.909	2.194%	74.705s	74.705s	23.838	3.113%	87.808s	87.808s
NeuOpt (fewer) (BS = 1)	7.808	0.670%	4.702h	13.225s	–	–	–	–	–	–	–	–
NeuOpt (fewer) (BS = 128)	7.808	0.670%	2.427m	14.559s	–	–	–	–	–	–	–	–
NeuOpt (more) (BS = 1)	7.782	0.335%	9.395h	26.423s	–	–	–	–	–	–	–	–
NeuOpt (more) (BS = 128)	7.782	0.335%	4.848m	29.089s	–	–	–	–	–	–	–	–
LEHD (fewer) (BS = 1)	7.764	0.104%	1.174h	3.302s	16.654	0.653%	33.948m	15.913s	23.511	1.698%	1.127h	31.705s
LEHD (fewer) (BS = 128)	7.763	0.091%	36.797s	3.680s	16.654	0.653%	80.949s	80.949s	23.511	1.698%	8.267m	8.267m
LEHD (more) (BS = 1)	7.758	0.026%	2.932h	8.247s	16.615	0.419%	1.403h	39.472s	23.446	1.419%	2.691h	75.686s
LEHD (more) (BS = 128)	7.758	0.023%	85.034s	8.503s	16.615	0.419%	3.079m	3.079m	23.446	1.419%	20.620m	20.620m
COReformer (fewer) (BS = 1)	7.758	0.022%	44.973s	0.035s	16.563	0.102%	16.992s	0.133s	23.163	0.195%	65.669s	0.770s
COReformer (fewer) (BS = 8)	**7.758**	**0.021%**	7.834s	0.049s	16.564	0.110%	4.128s	0.258s	**23.160**	**0.182%**	21.223s	1.326s
COReformer (fewer) (BS = 16)	7.758	0.022%	4.360s	0.054s	**16.562**	**0.096%**	3.633s	0.454s	23.162	0.189%	19.691s	2.461s
COReformer (fewer) (BS = 32)	7.758	0.024%	2.773s	0.069s	16.562	0.100%	3.424s	0.856s	23.162	0.192%	18.549s	4.637s
COReformer (fewer) (BS = 128)	7.758	0.024%	1.961s	0.196s	16.562	0.096%	3.349s	3.349s	23.162	0.189%	17.700s	17.700s
COReformer (more) (BS = 1)	7.757	0.011%	2.503m	0.117s	16.554	0.050%	61.797s	0.483s	23.145	0.114%	3.961m	1.859s
COReformer (more) (BS = 8)	7.757	0.008%	23.135s	0.145s	16.554	0.051%	12.811s	0.801s	23.144	0.114%	76.983s	4.793s
COReformer (more) (BS = 16)	**7.756**	**0.008%**	12.350s	0.154s	**16.553**	**0.046%**	10.890s	1.361s	23.144	0.110%	70.667s	8.833s
COReformer (more) (BS = 32)	7.757	0.010%	6.716s	0.168s	16.554	0.049%	9.992s	2.498s	**23.142**	**0.105%**	67.167s	16.792s
COReformer (more) (BS = 128)	7.756	0.008%	3.945s	0.395s	16.554	0.047%	9.519s	9.519s	23.144	0.111%	61.848s	61.848s

METHOD	CVRP-[UNIFORM-50]				CVRP-[UNIFORM-100]				CVRP-[UNIFORM-200]			
	OBJ.↓	DROP↓	TOTAL↓	TIME↓	OBJ.↓	DROP↓	TOTAL↓	TIME↓	OBJ.↓	DROP↓	TOTAL↓	TIME↓
HGS	10.366*	0.000%	2.792h	1.005s	15.563*	0.000%	55.631h	20.027s	19.630*	0.000%	1.667h	60.024s
SymNCO + LS (BS = 1)	10.565	1.910%	15.333m	0.092s	15.933	2.379%	28.833m	0.173s	20.193	2.880%	34.100s	0.341s
SymNCO + LS (BS = 10)	10.567	1.927%	1.787m	0.107s	15.933	2.379%	3.816m	0.229s	20.191	2.875%	5.979s	0.598s
SymNCO + LS (BS = 20)	10.567	1.934%	68.815s	0.138s	15.930	2.359%	2.524m	0.303s	20.182	2.828%	4.430s	0.886s
SymNCO + LS (BS = 50)	10.566	1.923%	46.012s	0.230s	15.929	2.356%	1.771m	0.531s	20.228	3.064%	3.537s	1.768s
SymNCO + LS (BS = 100)	10.565	1.912%	37.007s	0.370s	15.931	2.361%	1.525m	0.915s	20.168	2.752%	3.243s	3.243s
GOAL + LS (BS = 1)	10.628	2.519%	1.408h	0.507s	15.959	2.548%	2.692h	0.969s	20.396	3.936%	3.395m	2.307s
GOAL + LS (BS = 20)	10.626	2.500%	4.317m	0.518s	15.959	2.553%	10.427m	1.251s	20.376	3.817%	32.166s	6.433s
GOAL + LS (BS = 100)	10.628	2.519%	1.584m	0.950s	15.959	2.548%	7.363m	4.418s	20.400	3.952%	29.046s	29.046s
NeuOpt (fewer) (BS = 1)	10.491	1.206%	45.172h	16.262s	16.003	2.827%	75.848h	27.305s	–	–	–	–
NeuOpt (fewer) (BS = 100)	10.491	1.206%	28.469m	17.082s	16.003	2.827%	1.083h	38.996s	–	–	–	–
NeuOpt (more) (BS = 1)	10.438	0.695%	90.256h	32.492s	15.865	1.940%	151.544h	54.556s	–	–	–	–
NeuOpt (more) (BS = 100)	10.438	0.695%	56.882m	34.129s	15.865	1.940%	2.164h	77.914s	–	–	–	–
LEHD (fewer) (BS = 1)	10.517	1.441%	5.895h	2.122s	15.807	1.555%	13.475h	4.851s	20.029	2.039%	15.321m	9.193s
LEHD (fewer) (BS = 100)	10.516	1.433%	6.111m	3.667s	15.804	1.538%	11.445m	6.867s	20.029	2.039%	13.763s	13.763s
LEHD (more) (BS = 1)	10.450	0.792%	14.360h	5.170s	15.730	1.064%	30.498h	10.979s	19.928	1.527%	33.163m	19.898s
LEHD (more) (BS = 100)	10.450	0.797%	15.395m	9.237s	15.729	1.057%	26.853m	16.112s	19.928	1.527%	29.297s	29.297s
COReformer (fewer) (BS = 1)	10.441	0.710%	32.899m	0.197s	15.724	1.031%	51.089m	0.307s	19.852	1.133%	76.969s	0.770s
COReformer (fewer) (BS = 10)	10.441	0.712%	4.701m	0.282s	**15.722**	**1.024%**	7.005m	0.420s	19.844	1.097%	14.583s	1.458s
COReformer (fewer) (BS = 20)	10.440	0.706%	2.889m	0.347s	15.724	1.031%	4.238m	0.508s	**19.835**	**1.053%**	9.858s	1.972s
COReformer (fewer) (BS = 50)	10.441	0.710%	1.730m	0.519s	15.723	1.028%	2.549m	0.765s	19.842	1.087%	6.692s	3.346s
COReformer (fewer) (BS = 100)	**10.440**	**0.704%**	1.326m	0.795s	15.723	1.026%	2.042m	1.225s	19.836	1.054%	5.990s	5.990s
COReformer (more) (BS = 1)	10.416	0.472%	2.234h	0.804s	15.669	0.678%	3.315h	1.193s	19.766	0.692%	5.796m	3.477s
COReformer (more) (BS = 10)	**10.415**	**0.462%**	18.847m	1.131s	15.668	0.676%	28.804m	1.728s	19.755	0.641%	59.236s	5.924s
COReformer (more) (BS = 20)	10.415	0.466%	11.402m	1.368s	**15.667**	**0.667%**	17.591m	2.111s	**19.742**	**0.577%**	38.165s	7.633s
COReformer (more) (BS = 50)	10.416	0.472%	6.795m	2.039s	15.669	0.683%	10.502m	3.151s	19.760	0.668%	25.199s	12.600s
COReformer (more) (BS = 100)	10.415	0.462%	5.228m	3.137s	15.669	0.680%	8.386m	5.031s	19.764	0.691%	20.887s	20.887s

METHOD	CVRPTW-[UNIFORM-50]				CVRPTW-[UNIFORM-100]				CVRPTW-[UNIFORM-200]			
	OBJ.↓	DROP↓	TOTAL↓	TIME↓	OBJ.↓	DROP↓	TOTAL↓	TIME↓	OBJ.↓	DROP↓	TOTAL↓	TIME↓
PyVRP	16.038*	0.000%	2.781h	10.012s	25.431*	0.000%	5.566h	20.036s	41.597*	0.000%	1.667h	60.018s
SymNCO (BS = 1)	17.042	6.294%	80.645s	0.081s	27.599	8.520%	2.568m	0.154s	45.996	10.555%	29.813s	0.298s
SymNCO (BS = 10)	17.042	6.294%	9.316s	0.093s	27.599	8.520%	17.442s	0.174s	45.996	10.555%	3.404s	0.340s
MVMoE (BS = 1)	16.922	5.489%	2.620m	0.157s	28.679	12.672%	4.916m	0.295s	–	–	–	–
MVMoE (BS = 10)	16.922	5.489%	17.434s	0.174s	28.679	12.672%	33.124s	0.331s	–	–	–	–
RF-TE (BS = 1)	16.364	2.029%	2.381m	0.143s	26.235	3.151%	4.547m	0.273s	–	–	–	–
RF-TE (BS = 10)	16.364	2.029%	15.090s	0.151s	26.235	3.151%	28.090s	0.281s	–	–	–	–
CaDA (×8) (BS = 1)	16.278	1.496%	2.884m	0.173s	26.070	2.514%	5.364m	0.322s	–	–	–	–
CaDA (×8) (BS = 10)	16.278	1.496%	18.087s	0.181s	26.070	2.514%	33.604s	0.336s	–	–	–	–
MoSES (CaDA) (BS = 1)	16.261	1.384%	5.921m	0.355s	26.029	2.344%	9.633m	0.578s	–	–	–	–
MoSES (CaDA) (BS = 10)	16.261	1.380%	36.581s	0.366s	26.030	2.348%	59.435s	0.594s	–	–	–	–
COReformer (fewer) (BS = 1)	16.128	0.553%	5.245m	0.315s	**25.773**	**1.322%**	9.039m	0.542s	**42.702**	**2.586%**	1.820m	1.092s
COReformer (fewer) (BS = 10)	**16.123**	**0.520%**	48.047s	0.480s	25.780	1.354%	75.152s	0.752s	42.800	2.816%	17.768s	1.777s
COReformer (more) (BS = 1)	**16.080**	**0.255%**	21.988m	1.319s	**25.628**	**0.761%**	38.194m	2.292s	42.516	2.147%	7.616m	4.540s
COReformer (more) (BS = 10)	16.081	0.262%	3.048m	1.829s	25.634	0.785%	5.858m	3.515s	**42.492**	**2.087%**	85.385s	8.538s

Table 9: Complete results on TSP.

METHOD	TSP-[UNIFORM-100]			TSP-[UNIFORM-500]			TSP-[UNIFORM-1K]		
	OBJ.↓	DROP↓	TIME↓	OBJ.↓	DROP↓	TIME↓	OBJ.↓	DROP↓	TIME↓
Baseline Solvers									
Concorde (Applegate & Cook, 2006)	7.756*	0.000%	0.238s	16.546*	0.000%	18.672s	23.118*	0.000%	84.413s
LKH (500) (Helsgaun, 2017)	7.756	0.001%	0.176s	16.546	0.002%	1.848s	23.119	0.005%	4.641s
GA-EAX (Nagata & Kobayashi, 2013)	7.756	0.000%	1.862s	16.546	0.001%	1.857s	23.118	0.000%	17.544s
L2C Solvers									
SymNCO + 2OPT (Berto et al., 2023)	7.840	1.087%	0.100s	–	–	–	–	–	–
GOAL + 2OPT (Drakulic et al., 2025)	7.862	1.369%	0.764s	17.047	3.027%	4.057s	23.806	2.974%	8.065s
LEHD (Greedy) (Luo et al., 2023)	7.799	0.552%	0.304s	16.799	1.530%	1.497s	23.839	3.117%	3.090s
SIL (Greedy) (Luo et al., 2025)	–	–	–	–	–	–	23.569	1.953%	5.202s
GCN + 2OPT (Joshi et al., 2019)	7.775	0.243%	0.009s	16.769	1.348%	0.063s	23.527	1.769%	0.227s
GCN + MCTS (Fu et al., 2021)	7.762	0.073%	0.013s	16.658	0.676%	0.367s	23.380	1.131%	1.594s
DIMES + 2OPT (Qiu et al., 2022)	8.108	4.543%	0.012s	17.655	6.707%	0.314s	24.906	7.735%	0.662s
DIMES + MCTS (Qiu et al., 2022)	7.897	1.819%	0.025s	17.776	7.435%	0.833s	25.032	8.279%	3.870s
UTSP + 2OPT (Min et al., 2023)	8.161	5.215%	0.017s	17.710	7.033%	0.090s	24.765	7.122%	0.290s
UTSP + MCTS (Min et al., 2023)	8.052	3.817%	0.065s	17.913	8.261%	1.574s	25.069	8.440%	2.231s
DIFUSCO + 2OPT (Sun & Yang, 2023)	7.776	0.265%	2.691s	16.810	1.593%	2.781s	23.540	1.826%	3.422s
Fast-T2T + 2OPT (Li et al., 2024b)	7.762	0.078%	0.031s	16.690	0.869%	0.102s	23.423	1.319%	0.328s
Fast-T2T + 2OPT (×8) (Li et al., 2024b)	7.757	0.009%	0.118s	16.607	0.370%	0.591s	23.313	0.844%	2.471s
COExpander + 2OPT (Ma et al., 2025a)	7.760	0.058%	0.102s	16.626	0.487%	0.242s	23.337	0.946%	0.703s
COExpander + 2OPT (×8) (Ma et al., 2025a)	7.756	0.008%	0.329s	16.575	0.179%	1.265s	23.239	0.523%	5.104s
D&C Solvers and L2S Solvers									
GLOP (fewer) (Ye et al., 2024)	7.769	0.163%	4.674s	17.075	3.198%	4.818s	24.009	3.853%	4.759s
GLOP (more) (Ye et al., 2024)	7.760	0.052%	4.723s	16.909	2.197%	4.892s	23.838	3.113%	5.182s
UDC (fewer) (Zheng et al., 2024)	7.802	0.591%	0.099s	16.970	2.566%	0.156s	23.802	2.959%	0.239s
UDC (more) (Zheng et al., 2024)	7.781	0.323%	2.095s	16.822	1.670%	2.327s	23.528	1.774%	2.833s
NeuOpt (fewer) (Ma et al., 2023)	7.808	0.670%	13.225s	–	–	–	–	–	–
NeuOpt (more) (Ma et al., 2023)	7.782	0.335%	26.423s	–	–	–	–	–	–
LEHD (RRC) (fewer) (Luo et al., 2023)	7.764	0.104%	3.302s	16.654	0.653%	15.913s	23.511	1.698%	31.705s
LEHD (RRC) (more) (Luo et al., 2023)	7.758	0.026%	8.247s	16.615	0.419%	39.472s	23.446	1.419%	75.686s
SIL (PRC) (fewer) (Luo et al., 2025)	–	–	–	–	–	–	23.396	1.204%	28.080s
SIL (PRC) (more) (Luo et al., 2025)	–	–	–	–	–	–	23.331	0.920%	57.428s
COReformer (fewer)	7.758	0.022±0.089%	0.035s	16.563	0.102±0.110%	0.133s	23.163	0.195±0.114%	0.513s
COReformer (fewer) (×8)	7.756	0.003±0.020%	0.043s	16.550	0.028±0.037%	0.201s	23.139	0.089±0.063%	1.201s
COReformer (more)	7.757	0.011±0.050%	0.117s	16.554	0.050±0.062%	0.483s	23.145	0.114±0.080%	1.859s
COReformer (more) (×8)	7.756	0.001±0.013%	0.139s	16.548	0.014±0.028%	0.758s	23.129	0.049±0.043%	4.523s

METHOD	TSP-[UNIFORM-2K]			TSP-[UNIFORM-5K]			TSP-[UNIFORM-10K]		
	OBJ.↓	DROP↓	TIME↓	OBJ.↓	DROP↓	TIME↓	OBJ.↓	DROP↓	TIME↓
LKH (500) (Helsgaun, 2017)	32.436*	0.000%	7.143s	50.968*	0.000%	34.494s	71.782*	0.000%	332.758s
SIL (Greedy) (Luo et al., 2025)	33.163	2.242%	10.379s	52.579	3.161%	26.119s	74.688	4.048%	53.409s
GCN + 2OPT (Joshi et al., 2019)	33.028	1.829%	0.953s	51.929	1.889%	7.329s	73.071	1.795%	31.646s
Fast-T2T + 2OPT (Li et al., 2024b)	33.001	1.745%	1.101s	51.805	1.646%	7.698s	73.027	1.735%	32.169s
COExpander + 2OPT (Ma et al., 2025a)	32.907	1.454%	1.448s	51.669	1.378%	8.393s	72.861	1.504%	33.686s
GLOP (fewer) (Ye et al., 2024)	33.846	4.346%	4.953s	53.605	5.174%	3.286s	75.614	5.338%	3.909s
GLOP (more) (Ye et al., 2024)	33.700	3.897%	5.729s	53.367	4.708%	9.508s	75.288	4.885%	10.594s
UDC (fewer) (Zheng et al., 2024)	33.908	4.539%	0.522s	55.277	8.455%	2.297s	80.289	11.851%	12.158s
UDC (more) (Zheng et al., 2024)	33.232	2.454%	5.300s	53.758	5.476%	14.086s	77.702	8.247%	35.231s
SIL (PRC) (fewer) (Luo et al., 2025)	33.005	1.754%	28.173s	52.327	2.666%	28.377s	73.939	3.005%	28.067s
SIL (PRC) (more) (Luo et al., 2025)	32.875	1.353%	61.329s	52.161	2.342%	52.887s	73.640	2.588%	51.181s
COReformer (fewer)	32.602	0.511±0.171%	1.136s	51.455	0.956±0.124%	4.684s	72.679	1.250±0.088%	12.627s
COReformer (more)	32.530	0.288±0.139%	4.844s	51.315	0.681±0.132%	17.593s	72.562	1.087±0.101%	37.031s

METHOD	TSP-[GAUSSIAN-100]			TSP-[GAUSSIAN-200]			TSP-[GAUSSIAN-500]		
	OBJ.↓	DROP↓	TIME↓	OBJ.↓	DROP↓	TIME↓	OBJ.↓	DROP↓	TIME↓
Concorde (Applegate & Cook, 2006)	34.031*	0.000%	0.438s	48.127*	0.000%	1.753s	77.521*	0.000%	19.952s
LKH (500) (Helsgaun, 2017)	34.031	0.000%	0.174s	48.127	0.000%	0.373s	77.521	0.000%	1.218s
GenSCO (C=20) (×8) (Li et al., 2025)	34.031	0.001%	–	143.431	198.028%	–	584.478	653.965%	–
M^2GenCO + MCTS (Pan et al., 2026a)	34.258	0.681%	0.018s	48.690	1.177%	0.088s	78.899	1.783%	0.716s
COReformer (fewer)	34.055	0.071±0.174%	0.039s	48.227	0.206±0.287%	0.071s	78.170	0.833±0.646%	0.134s
COReformer (fewer) (×8)	34.034	0.009±0.035%	0.044s	48.149	0.045±0.088%	0.082s	77.812	0.375±0.248%	0.235s
COReformer (more)	34.039	0.024±0.071%	0.125s	48.166	0.081±0.127%	0.262s	77.822	0.387±0.277%	0.500s
COReformer (more) (×8)	34.032	0.003±0.016%	0.153s	48.138	0.022±0.067%	0.301s	77.667	0.187±0.169%	0.926s

METHOD	TSP-[CLUSTER-100]			TSP-[CLUSTER-200]			TSP-[CLUSTER-500]		
	OBJ.↓	DROP↓	TIME↓	OBJ.↓	DROP↓	TIME↓	OBJ.↓	DROP↓	TIME↓
Concorde (Applegate & Cook, 2006)	5.526*	0.000%	0.290s	6.912*	0.000%	0.965s	10.723*	0.000%	5.073s
LKH (500) (Helsgaun, 2017)	5.530	0.005%	0.114s	6.913	0.011%	0.542s	10.725	0.017%	2.063s
GenSCO (C=20) (×8) (Li et al., 2025)	5.528	0.025%	–	7.165	3.648%	–	10.795	0.676%	–
M^2GenCO + MCTS (Pan et al., 2026a)	5.549	0.426%	0.019s	7.050	1.939%	0.163s	11.063	3.103%	0.977s
COReformer (fewer)	5.530	0.072±0.220%	0.038s	6.937	0.358±0.395%	0.078s	10.841	1.101±0.691%	0.121s
COReformer (fewer) (×8)	5.528	0.023±0.133%	0.046s	6.920	0.111±0.175%	0.083s	10.789	0.612±0.413%	0.168s
COReformer (more)	5.528	0.032±0.134%	0.127s	6.923	0.158±0.287%	0.262s	10.790	0.625±0.465%	0.429s
COReformer (more) (×8)	5.527	0.011±0.078%	0.145s	6.917	0.063±0.139%	0.332s	10.764	0.379±0.288%	0.610s

Table 10: Complete results on CVRP and CVRPTW.

Method	CVRP-[Uniform-50]			CVRP-[Uniform-100]			CVRP-[Uniform-200]		
	Obj.↓	Drop↓	Time↓	Obj.↓	Drop↓	Time↓	Obj.↓	Drop↓	Time↓
Baseline Solvers									
HGS (Vidal et al., 2012)	10.366*	0.000%	1.005s	15.563*	0.000%	20.027s	19.630*	0.000%	60.024s
L2C Solvers									
SymNCO + LS (Berto et al., 2023)	10.565	1.910%	0.092s	15.933	2.379%	0.173s	20.193	2.880%	0.341s
GOAL + LS (Drakulic et al., 2025)	10.628	2.519%	0.507s	15.959	2.548%	0.969s	20.396	3.936%	2.037s
LEHD (Greedy) (Luo et al., 2023)	10.886	5.007%	0.195s	16.217	4.197%	0.398s	20.455	4.199%	0.765s
Fast-T2T + LS (Li et al., 2024b)	10.871	4.836%	0.013s	16.294	4.698%	0.018s	20.662	5.290%	0.063s
COExpander + LS (Ma et al., 2025a)	10.773	3.903%	0.037s	16.224	4.253%	0.055s	20.587	4.893%	0.153s
L2S Solvers									
NeuOpt (fewer) (Ma et al., 2023)	10.491	1.206%	16.262s	16.003	2.827%	27.305s	–	–	–
NeuOpt (more) (Ma et al., 2023)	10.438	0.695%	32.492s	15.865	1.940%	54.556s	–	–	–
LEHD (RRC) (fewer) (Luo et al., 2023)	10.517	1.441%	2.122s	15.807	1.555%	4.851s	20.029	2.039%	9.193s
LEHD (RRC) (more) (Luo et al., 2023)	10.450	0.792%	5.170s	15.730	1.064%	10.979s	19.928	1.527%	19.898s
COReformer (fewer)	10.441	0.710±0.782%	0.197s	15.724	1.031±0.690%	0.307s	19.852	1.133±0.546%	0.770s
COReformer (more)	10.416	0.472±0.583%	0.804s	15.669	0.678±0.512%	1.193s	19.766	0.692±0.444%	3.477s

Method	CVRP-[Uniform-500]			CVRP-[Uniform-1K]			CVRP-[Uniform-2K]		
	Obj.↓	Drop↓	Time↓	Obj.↓	Drop↓	Time↓	Obj.↓	Drop↓	Time↓
Baseline Solvers									
HGS (Vidal et al., 2012)	37.154*	0.000%	360.376s	41.171*	0.000%	360.051s	57.181*	0.000%	360.302s
L2C Solvers									
SymNCO + LS (Berto et al., 2023)	38.700	4.173%	0.883s	–	–	–	–	–	–
GOAL + LS (Drakulic et al., 2025)	38.118	2.620%	7.312s	–	–	–	–	–	–
LEHD (Greedy) (Luo et al., 2023)	38.990	4.944%	1.922s	43.964	6.798%	3.896s	61.585	7.791%	9.926s
SIL (Greedy) (Luo et al., 2025)	–	–	–	44.077	6.980%	6.460s	59.620	4.198%	13.076s
Fast-T2T + LS (Li et al., 2024b)	39.195	5.530%	0.215s	–	–	–	–	–	–
COExpander + LS (Ma et al., 2025a)	39.121	5.337%	0.605s	–	–	–	–	–	–
D&C Solvers and L2S Solvers									
GLOP (fewer) (Ye et al., 2024)	–	–	–	47.059	14.307%	0.467s	63.602	11.115%	1.224s
GLOP (more) (Ye et al., 2024)	–	–	–	46.907	13.928%	0.646s	63.190	10.390%	1.826s
UDC (fewer) (Zheng et al., 2024)	39.018	5.043%	8.288s	43.610	5.868%	8.912s	62.508	9.081%	11.478s
UDC (more) (Zheng et al., 2024)	38.629	3.995%	39.899s	43.043	4.528%	40.789s	60.861	6.299%	52.152s
LEHD (RRC) (fewer) (Luo et al., 2023)	38.383	3.315%	24.503s	43.416	5.466%	37.824s	60.778	6.367%	96.962s
LEHD (RRC) (more) (Luo et al., 2023)	38.208	2.852%	49.371s	43.155	4.833%	90.138s	60.483	5.835%	213.484s
SIL (PRC) (fewer) (Luo et al., 2025)	–	–	–	43.645	5.929%	25.658s	59.560	4.102%	26.787s
SIL (PRC) (more) (Luo et al., 2025)	–	–	–	43.457	5.479%	51.666s	59.211	3.484%	56.639s
COReformer (fewer)	37.839	1.848±0.506%	2.332s	43.075	4.629±0.668%	2.095s	59.749	4.552±0.946%	4.112s
COReformer (more)	37.627	1.267±0.443%	10.239s	42.577	3.426±0.606%	8.416s	59.125	3.465±0.919%	15.304s

Method	CVRPTW-[Uniform-50]			CVRPTW-[Uniform-100]			CVRPTW-[Uniform-200]		
	Obj.↓	Drop↓	Time↓	Obj.↓	Drop↓	Time↓	Obj.↓	Drop↓	Time↓
Baseline Solvers									
PyVRP (Wouda et al., 2024)	16.038*	0.000%	10.012s	25.431*	0.000%	20.036s	41.597*	0.000%	60.018s
L2C Solvers									
SymNCO (Berto et al., 2023)	17.042	6.290%	0.082s	27.599	8.520%	0.156s	45.996	10.550%	0.302s
MVMoE (Zhou et al., 2024)	16.922	5.489%	0.157s	28.679	12.672%	0.295s	–	–	–
RF-TE (Berto et al., 2025)	16.364	2.029%	0.143s	26.235	3.151%	0.273s	–	–	–
CaDA (×8) (Li et al., 2024a)	16.278	1.496%	0.173s	26.070	2.514%	0.322s	–	–	–
MoSES (CaDA) (Pan et al., 2026b)	16.261	1.384%	0.355s	26.029	2.344%	0.578s	–	–	–
L2S Solvers									
COReformer (fewer)	16.128	0.553±0.780%	0.315s	25.773	1.322±0.934%	0.542s	42.702	2.586±1.337%	1.092s
COReformer (more)	16.080	0.255±0.533%	1.319s	25.628	0.761±0.669%	2.292s	42.516	2.147±1.009%	4.540s

Table 11: Generalization Study on TSPLIB.

INSTANCE	MODEL	EXACT	GCN		Fast-T2T		COExpander		COReformer	
			OBJ.↓	DROP↓	OBJ.↓	DROP↓	OBJ.↓	DROP↓	OBJ.↓	DROP↓
eil51	TSP-100	6.701	6.703	0.026%	6.711	0.143%	6.701	0.000%	6.701	0.000%
berlin52	TSP-100	4.348	4.348	0.000%	4.349	0.004%	4.349	0.004%	4.348	0.000%
st70	TSP-100	6.839	6.840	0.000%	6.840	0.000%	6.840	0.000%	6.839	0.000%
eil76	TSP-100	7.561	7.668	1.414%	7.561	0.000%	7.561	0.000%	7.561	0.000%
pr76	TSP-100	5.518	5.606	1.582%	5.518	0.000%	5.518	0.000%	5.518	0.000%
rat99	TSP-100	5.671	5.671	0.000%	5.675	0.079%	5.675	0.079%	5.671	0.000%
kroA100	TSP-100	5.408	5.408	0.000%	5.408	0.000%	5.408	0.000%	5.408	0.000%
kroB100	TSP-100	5.626	5.668	0.742%	5.668	0.742%	5.668	0.742%	5.626	0.000%
kroC100	TSP-100	5.276	5.276	0.000%	5.452	3.330%	5.452	3.330%	5.276	0.000%
kroD100	TSP-100	5.431	5.431	0.000%	5.431	0.000%	5.431	0.000%	5.431	0.000%
kroE100	TSP-100	5.557	5.641	1.495%	5.567	0.178%	5.570	0.232%	5.557	0.000%
rd100	TSP-100	8.065	8.065	0.000%	8.065	0.000%	8.065	0.000%	8.065	0.000%
eil101	TSP-100	8.536	8.536	0.000%	8.536	0.000%	8.536	0.000%	8.536	0.000%
lin105	TSP-100	4.756	4.756	0.000%	4.793	0.779%	4.756	0.000%	4.756	0.000%
pr107	TSP-100	3.778	3.778	0.000%	3.865	2.292%	3.816	1.005%	3.778	0.000%
pr124	TSP-100	5.009	5.013	0.075%	5.009	0.000%	5.009	0.000%	5.009	0.000%
bier127	TSP-100	6.106	6.165	0.953%	6.176	1.140%	6.215	1.771%	6.106	0.000%
ch130	TSP-100	8.751	8.771	0.235%	8.751	0.000%	8.751	0.000%	8.751	0.000%
pr136	TSP-100	7.767	7.875	1.391%	7.767	0.010%	7.767	0.010%	7.767	0.010%
pr144	TSP-100	4.538	4.591	1.175%	4.557	0.417%	4.555	0.386%	4.538	0.000%
ch150	TSP-100	9.342	9.429	0.939%	9.342	0.000%	9.342	0.000%	9.342	0.000%
kroA150	TSP-100	6.693	6.723	0.444%	6.775	1.230%	6.733	0.595%	6.693	0.000%
kroB150	TSP-100	6.635	6.667	0.482%	6.647	0.188%	6.647	0.189%	6.639	0.067%
pr152	TSP-500	5.234	5.391	2.997%	5.291	1.097%	5.244	0.191%	5.249	0.283%
u159	TSP-500	6.574	6.574	0.000%	6.574	0.000%	6.574	0.000%	6.574	0.000%
rat195	TSP-500	7.993	8.049	0.705%	8.026	0.419%	8.049	0.706%	8.002	0.114%
d198	TSP-500	3.924	3.998	1.884%	3.950	0.649%	3.944	0.488%	3.967	1.091%
kroA200	TSP-500	7.437	7.475	0.513%	7.491	0.723%	7.441	0.056%	7.437	0.000%
kroB200	TSP-500	7.467	7.481	0.197%	7.573	1.421%	7.569	1.373%	7.467	0.000%
ts225	TSP-500	10.554	10.740	1.759%	10.754	1.893%	10.675	1.150%	10.554	0.000%
tsp225	TSP-500	7.895	7.989	1.191%	7.911	0.200%	7.960	0.822%	7.895	0.000%
pr226	TSP-500	5.279	5.288	0.169%	5.308	0.551%	5.315	0.683%	5.282	0.052%
gil262	TSP-500	12.050	12.166	0.963%	12.075	0.215%	12.070	0.166%	12.050	0.000%
pr264	TSP-500	6.200	6.245	0.731%	6.200	0.000%	6.200	0.000%	6.200	0.000%
a280	TSP-500	9.238	9.412	1.883%	9.247	0.096%	9.238	0.000%	9.238	0.000%
pr299	TSP-500	6.638	6.743	1.569%	6.729	1.367%	6.725	1.306%	6.640	0.022%
lin318	TSP-500	10.170	10.305	1.327%	10.209	0.386%	10.244	0.727%	10.170	0.000%
rd400	TSP-500	15.344	15.492	0.964%	15.407	0.412%	15.359	0.100%	15.347	0.021%
fl417	TSP-500	6.286	6.355	1.112%	6.413	2.033%	6.354	1.095%	6.376	1.439%
pr439	TSP-500	8.991	9.255	2.936%	9.065	0.822%	9.123	1.475%	8.995	0.044%
pcb442	TSP-500	13.364	13.543	1.337%	13.450	0.645%	13.441	0.576%	13.366	0.016%
d493	TSP-500	9.350	9.509	1.696%	9.510	1.706%	9.469	1.265%	9.473	1.308%
u574	TSP-500	12.023	12.097	0.613%	12.149	1.048%	12.092	0.577%	12.047	0.200%
rat575	TSP-500	13.619	13.694	0.551%	13.791	1.260%	13.671	0.382%	13.648	0.211%
p654	TSP-1000	7.196	7.516	4.454%	7.240	0.624%	7.240	0.621%	7.285	1.247%
d657	TSP-1000	12.212	12.323	0.911%	12.334	1.003%	12.288	0.620%	12.214	0.014%
u724	TSP-1000	14.437	14.598	1.119%	14.593	1.087%	14.582	1.009%	14.471	0.235%
rat783	TSP-1000	15.247	15.373	0.828%	15.342	0.625%	15.418	1.127%	15.270	0.156%
pr1002	TSP-1000	16.397	16.645	1.517%	16.561	1.001%	16.490	0.571%	16.400	0.020%
mean	–	**8.062**	**8.140**	**0.916%**	**8.115**	**0.649%**	**8.105**	**0.519%**	**8.072**	**0.134%**

Table 12: Generalization Study on CVRPLIB. BKS: best known solutions.

INSTANCE	MODEL	BKS	SymNCO		COReformer	
			OBJ.↓	DROP↓	OBJ.↓	DROP↓
X-n101-k25	CVRP-200	27.905	31.702	13.605%	28.504	2.144%
X-n106-k14	CVRP-200	26.362	26.802	1.668%	26.671	1.169%
X-n110-k13	CVRP-200	14.979	15.219	1.599%	15.119	0.929%
X-n115-k10	CVRP-200	12.933	14.112	9.115%	12.943	0.083%
X-n120-k6	CVRP-200	13.437	13.672	1.749%	13.477	0.296%
X-n125-k30	CVRP-200	56.164	62.810	11.833%	57.415	2.227%
X-n129-k18	CVRP-200	29.005	29.356	1.211%	29.282	0.957%
X-n134-k13	CVRP-200	11.351	11.810	4.045%	11.526	1.539%
X-n139-k10	CVRP-200	13.678	13.804	0.920%	13.729	0.373%
X-n143-k7	CVRP-200	15.776	16.602	5.239%	16.026	1.586%
X-n148-k46	CVRP-200	43.671	45.190	3.478%	44.118	1.024%
X-n153-k22	CVRP-200	21.572	36.105	67.371%	21.812	1.111%
X-n157-k13	CVRP-200	17.572	17.870	1.696%	17.622	0.288%
X-n162-k11	CVRP-200	14.412	14.707	2.044%	14.470	0.402%
X-n167-k10	CVRP-200	20.766	21.287	2.510%	21.140	1.801%
X-n172-k51	CVRP-200	46.974	79.562	69.372%	47.398	0.901%
X-n176-k26	CVRP-200	47.816	56.172	17.475%	49.176	2.844%
X-n181-k23	CVRP-200	26.809	27.092	1.057%	26.906	0.364%
X-n186-k15	CVRP-200	24.251	24.721	1.936%	24.448	0.811%
X-n190-k8	CVRP-200	17.037	17.418	2.238%	17.285	1.454%
X-n195-k51	CVRP-200	44.274	60.037	35.604%	44.911	1.441%
X-n200-k36	CVRP-200	60.084	61.860	2.957%	61.516	2.384%
X-n204-k19	CVRP-200	19.588	20.034	2.279%	19.812	1.142%
X-n209-k16	CVRP-200	30.691	31.107	1.355%	31.094	1.312%
X-n214-k11	CVRP-200	10.879	11.278	3.665%	11.057	1.631%
X-n219-k73	CVRP-200	117.601	126.506	7.572%	117.707	0.090%
X-n223-k34	CVRP-200	40.478	42.331	4.576%	41.075	1.473%
X-n228-k23	CVRP-200	25.747	27.065	5.119%	26.178	1.675%
X-n233-k16	CVRP-200	19.297	19.731	2.249%	19.688	2.026%
X-n237-k14	CVRP-200	27.078	27.397	1.178%	27.504	1.572%
X-n242-k48	CVRP-200	82.757	84.427	2.018%	84.256	1.811%
X-n247-k50	CVRP-200	37.966	63.388	66.958%	38.602	1.674%
X-n251-k28	CVRP-200	38.885	39.798	2.348%	39.348	1.191%
X-n256-k16	CVRP-500	18.942	19.432	2.582%	19.126	0.972%
X-n261-k13	CVRP-500	26.591	27.591	3.759%	27.606	3.815%
X-n266-k58	CVRP-500	75.489	78.007	3.336%	76.793	1.728%
X-n270-k35	CVRP-500	35.432	37.645	6.244%	35.893	1.298%
X-n275-k28	CVRP-500	22.924	23.829	3.948%	23.264	1.485%
X-n280-k17	CVRP-500	33.503	36.544	9.074%	33.991	1.455%
X-n284-k15	CVRP-500	20.421	21.472	5.148%	21.050	3.080%
X-n289-k60	CVRP-500	95.351	105.407	10.546%	97.139	1.875%
X-n294-k50	CVRP-500	47.315	49.012	3.588%	48.263	2.004%
X-n298-k31	CVRP-500	34.247	35.698	4.236%	35.561	3.837%
X-n303-k21	CVRP-500	21.961	22.889	4.225%	22.459	2.266%
X-n308-k13	CVRP-500	25.980	26.891	3.505%	26.692	2.738%
X-n313-k71	CVRP-500	94.437	107.451	13.780%	96.618	2.310%
X-n317-k53	CVRP-500	78.456	93.404	19.053%	78.644	0.240%
X-n322-k28	CVRP-500	29.846	31.861	6.753%	30.538	2.319%
X-n327-k20	CVRP-500	27.681	28.733	3.799%	28.122	1.595%
X-n331-k15	CVRP-500	31.155	32.349	3.834%	31.839	2.197%
X-n336-k84	CVRP-500	139.283	157.244	12.896%	142.874	2.578%
X-n344-k43	CVRP-500	42.194	43.542	3.194%	43.137	2.235%
X-n351-k40	CVRP-500	25.948	27.259	5.051%	26.691	2.861%
X-n359-k29	CVRP-500	51.512	53.347	3.562%	52.960	2.812%
X-n367-k17	CVRP-500	24.741	25.711	3.920%	25.499	3.065%
X-n376-k94	CVRP-500	147.881	148.848	0.654%	148.363	0.326%
X-n384-k52	CVRP-500	66.086	68.387	3.482%	67.405	1.997%
X-n393-k38	CVRP-500	38.386	39.702	3.427%	38.980	1.547%
X-n401-k29	CVRP-500	66.204	68.101	2.865%	67.211	1.521%
X-n411-k19	CVRP-500	19.741	21.476	8.791%	20.347	3.072%
X-n420-k130	CVRP-500	108.137	152.736	41.243%	110.322	2.020%
X-n429-k61	CVRP-500	65.777	70.841	7.698%	67.226	2.202%
X-n439-k37	CVRP-500	36.484	38.941	6.735%	36.919	1.191%
X-n449-k29	CVRP-500	55.282	59.498	7.627%	57.029	3.160%
X-n459-k26	CVRP-500	25.279	26.814	6.070%	26.117	3.313%
X-n469-k138	CVRP-500	222.364	272.896	22.725%	227.721	2.409%
X-n480-k70	CVRP-500	89.842	92.732	3.217%	92.179	2.601%
X-n491-k59	CVRP-500	66.591	71.887	7.952%	69.011	3.633%
X-n502-k39	CVRP-500	69.262	74.836	8.049%	69.597	0.483%
X-n513-k21	CVRP-500	24.273	25.259	4.061%	24.819	2.247%
mean		**45.183**	**50.132**	**8.981%**	**45.997**	**1.746%**

Table 13: COReformer vs. Previous L2C SOTA Solvers.

Problem	Dataset	Size	Previous L2C SOTA Solvers			COReformer (fewer)		COReformer (more)	
			Method	Drop↓	Time↓	Drop↓	Time↓	Drop↓	Time↓
TSP	Uniform-100	1280	COExpander (×8) (Ma et al., 2025a)	0.008%	0.329s	0.003%	0.043s	0.001%	0.139s
TSP	Uniform-500	128	COExpander (×8) (Ma et al., 2025a)	0.179%	1.265s	0.028%	0.201s	0.014%	0.758s
TSP	Uniform-1K	128	COExpander (×8) (Ma et al., 2025a)	0.523%	5.104s	0.089%	1.201s	0.049%	4.523s
TSP	Uniform-2K	64	COExpander (Ma et al., 2025a)	1.454%	1.448s	0.511%	1.136s	0.288%	4.844s
TSP	Uniform-5K	32	COExpander (Ma et al., 2025a)	1.378%	8.393s	0.956%	4.684s	0.681%	17.593s
TSP	Uniform-10K	16	COExpander (Ma et al., 2025a)	1.504%	33.686s	1.250%	12.627s	1.087%	37.031s
CVRP	Uniform-50	10000	SymNCO (Berto et al., 2023)	1.910%	0.092s	0.710%	0.197s	0.472%	0.804s
CVRP	Uniform-100	10000	SymNCO (Berto et al., 2023)	2.379%	0.173s	1.031%	0.307s	0.678%	1.193s
CVRP	Uniform-200	100	SymNCO (Berto et al., 2023)	2.880%	0.341s	1.133%	0.770s	0.692%	3.477s
CVRP	Uniform-500	100	GOAL (Drakulic et al., 2025)	2.620%	7.312s	1.848%	2.332s	1.267%	10.239s
CVRP	Uniform-1K	100	LEHD (Greedy) (Luo et al., 2023)	6.798%	3.896s	4.629%	2.095s	3.426%	8.416s
CVRP	Uniform-2K	100	SIL (Greedy) (Luo et al., 2025)	4.198%	13.076s	4.552%	4.112s	3.465%	15.304s
CVRPTW	Uniform-50	1000	MoSES (CaDA) (Pan et al., 2026b)	1.384%	0.355s	0.553%	0.315s	0.255%	1.319s
CVRPTW	Uniform-100	1000	MoSES (CaDA) (Pan et al., 2026b)	2.344%	0.578s	1.322%	0.542s	0.761%	2.292s
CVRPTW	Uniform-200	100	SymNCO (Berto et al., 2023)	10.550%	0.302s	2.586%	1.092s	2.147%	4.540s
TSP	***Summary***	***mean***	Previous L2C SOTA Solvers	**0.841%**	**8.371s**	**0.473%**	**3.315s**	**0.353%**	**10.815s**
CVRP	***Summary***	***mean***	Previous L2C SOTA Solvers	**3.464%**	**4.148s**	**2.317%**	**1.636s**	**1.667%**	**6.572s**
CVRPTW	***Summary***	***mean***	Previous L2C SOTA Solvers	**4.759%**	**0.412s**	**1.487%**	**0.650s**	**1.054%**	**2.717s**

Table 14: COReformer vs. Previous L2S SOTA Solvers.

Problem	Dataset	Size	Method	Method (fewer)		Method (more)		COReformer (fewer)		COReformer (more)	
				Drop↓	Time↓	Drop↓	Time↓	Drop↓	Time↓	Drop↓	Time↓
TSP	Uniform-100	1280	LEHD (RRC) (Luo et al., 2023)	0.104%	3.302s	0.026%	8.247s	0.022%	0.035s	0.011%	0.117s
TSP	Uniform-500	128	LEHD (RRC) (Luo et al., 2023)	0.653%	15.913s	0.419%	39.472s	0.102%	0.133s	0.050%	0.483s
TSP	Uniform-1K	128	SIL (PRC) (Luo et al., 2025)	1.204%	28.080s	0.920%	57.428s	0.195%	0.513s	0.114%	1.859s
TSP	Uniform-2K	64	SIL (PRC) (Luo et al., 2025)	1.754%	28.173s	1.353%	61.329s	0.511%	1.136s	0.288%	4.844s
TSP	Uniform-5K	32	SIL (PRC) (Luo et al., 2025)	2.666%	28.377s	2.342%	52.887s	0.956%	4.684s	0.681%	17.593s
TSP	Uniform-10K	16	SIL (PRC) (Luo et al., 2025)	3.005%	28.067s	2.588%	51.181s	1.250%	12.627s	1.087%	37.031s
CVRP	Uniform-50	10000	NeuOpt (Ma et al., 2023)	1.206%	16.262s	0.695%	32.492s	0.710%	0.197s	0.472%	0.804s
CVRP	Uniform-100	10000	LEHD (RRC) (Luo et al., 2023)	1.555%	4.851s	1.064%	10.979s	1.031%	0.307s	0.678%	1.193s
CVRP	Uniform-200	100	LEHD (RRC) (Luo et al., 2023)	2.039%	9.193s	1.527%	19.898s	1.133%	0.770s	0.692%	3.477s
CVRP	Uniform-500	100	LEHD (RRC) (Luo et al., 2023)	3.315%	24.503s	2.852%	49.371s	1.848%	2.332s	1.267%	10.239s
CVRP	Uniform-1K	100	LEHD (RRC) (Luo et al., 2023)	5.466%	37.824s	4.833%	90.138s	4.629%	2.095s	3.426%	8.416s
CVRP	Uniform-2K	100	SIL (PRC) (Luo et al., 2025)	4.102%	26.787s	3.484%	56.639s	4.552%	4.112s	3.465%	15.304s
TSP	***Summary***	***mean***	Previous L2S SOTA Solvers	**1.564%**	**21.985s**	**1.275%**	**45.091s**	**0.506%**	**3.188s**	**0.372%**	**10.321s**
CVRP	***Summary***	***mean***	Previous L2S SOTA Solvers	**2.947%**	**19.903s**	**2.409%**	**43.253s**	**2.317%**	**1.636s**	**1.667%**	**6.572s**

Table 15: COReformer vs. Previous D&C SOTA Solvers.

Problem	Dataset	Size	Method	Method (fewer)		Method (more)		COReformer (fewer)		COReformer (more)	
				Drop↓	Time↓	Drop↓	Time↓	Drop↓	Time↓	Drop↓	Time↓
TSP	Uniform-100	1280	GLOP (Ye et al., 2024)	0.163%	4.674s	0.052%	4.723s	0.022%	0.035s	0.011%	0.117s
TSP	Uniform-500	128	UDC (Zheng et al., 2024)	2.566%	0.156s	1.670%	2.327s	0.102%	0.133s	0.050%	0.483s
TSP	Uniform-1K	128	UDC (Zheng et al., 2024)	2.959%	0.239s	1.774%	2.833s	0.195%	0.513s	0.114%	1.859s
TSP	Uniform-2K	64	UDC (Zheng et al., 2024)	4.539%	0.522s	2.454%	5.300s	0.511%	1.136s	0.288%	4.844s
TSP	Uniform-5K	32	GLOP (Ye et al., 2024)	5.174%	3.286s	4.708%	9.508s	0.956%	4.684s	0.681%	17.593s
TSP	Uniform-10K	16	GLOP (Ye et al., 2024)	5.338%	3.909s	4.885%	10.594s	1.250%	12.627s	1.087%	37.031s
CVRP	Uniform-500	100	UDC (Zheng et al., 2024)	5.043%	8.288s	3.995%	39.899s	1.848%	2.332s	1.267%	10.239s
CVRP	Uniform-1K	100	UDC (Zheng et al., 2024)	5.868%	8.912s	4.528%	40.789s	4.629%	2.095s	3.426%	8.416s
CVRP	Uniform-2K	100	UDC (Zheng et al., 2024)	9.081%	11.478s	6.299%	52.152s	4.552%	4.112s	3.465%	15.304s
TSP	***Summary***	***mean***	Previous D&C SOTA Solvers	**3.457%**	**2.131s**	**2.591%**	**5.881s**	**0.506%**	**3.188s**	**0.372%**	**10.321s**
CVRP	***Summary***	***mean***	Previous D&C SOTA Solvers	**6.664%**	**9.559s**	**4.941%**	**44.280s**	**3.676%**	**2.846s**	**2.719%**	**11.320s**

Table 16: Notation used throughout the paper.

SYMBOL	DESCRIPTION
	Problem and solutions
$G = (V, E)$	Problem instance as a graph with node set V and edge set E
$N = \|V\|$, M	Number of nodes; number of decision variables ($M = N^2$ for routing)
C	Cost matrix (routing) or adjacency matrix (MCut)
$S \in \{0,1\}^M$	Binary decision vector (feasible solution) in the main text and RS equations
$\mathbf{x}, \mathbf{x}^*$	Discrete feasible solution and incumbent in Algorithm 2 (same role as S)
S^*	Optimal (labeled) solution used for SL / RL training
$\Omega(G)$, $c(S, G)$	Feasible set and objective cost of solution S on instance G
π, Π	A single route; a set of routes (CVRP and CVRPTW)
v_0, d, Q	Depot node, customer demands, and vehicle capacity
p_i, $[w_i^{\text{beg}}, w_i^{\text{end}}]$	Service time and time window of customer i (CVRPTW)
α_i, β_i	Arrival time and service start time at node i (CVRPTW)
$T_{\max}$, Δ_i	Depot time horizon and sampled window length for TW generation
$(U, \overline{U})$	Bipartition defining a cut on MCut
	Perturbed Rectified Flow (PRF)
π_0, π_1	Source and target distributions over (perturbed) solutions
π_0^{rnd}, Mix	Random-feasible source component; equal-probability mixing operator
S_0, S_1, S_τ	Continuous transport endpoints and interpolant at flow time $\tau \in [0, 1]$
$\mathbf{v}_\theta$, f_θ	Velocity field; target-state network outputting heatmap $\hat{S}_1$
$\mathcal{P}(\cdot,\cdot,\cdot)$ / $\mathcal{P}_K$	Feasibility-preserving perturbation operator applying K local moves
K_1, K_2	Training perturbation budgets for the target / source distributions ($K_2 > K_1$)
k_1, k_2	Perturbation ratios in Table 6; related by $K_1 = k_1 N$ and $K_2 = k_2 N$
$K, [K_{\min}, K_{\max}]$	Inference perturbation budget, sampled from $[K_{\min}, K_{\max}]$ in Algorithm 2
$\mathcal{H}$	Retention heatmap produced by PRF for progressive reduction
N_τ	Number of ODE / Euler integration steps at PRF inference
	Reduction and LinkGraph
$\mathbf{y} \in \{0,1\}^M$	Retention indicator (y_j=1: keep / fix decision variable S_j)
Y, $z(G, S, \mathbf{y})$	Nested progressive-retention sequence; residual size under retention $\mathbf{y}$
$Z, [Z_{\min}, Z_{\max}]$	Target residual size (sampled per local iteration) and its range
$\mathcal{Y}(\mathcal{H}, S, Z)$	Stochastic progressive reduction guided by $\mathcal{H}$ under target size Z
$\mathcal{R}$, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$	Reduction operator; induced LinkGraph with reduced nodes / edges
$\mathcal{Q}$, q_i, ϕ	Incumbent fragments, the i-th fragment, and fragment-to-node mapping
	Solve stage and RS search
g_θ / $\hat{g}_\theta$, l_θ / $\hat{l}_\theta$	Global predictor (PRF) and local constructor (SymNCO)
π_θ, s_t, a_t, b	RL policy, MDP state / action at step t, and variance-reduction baseline
ξ, $\mathcal{S}(\xi, \cdot)$	Construction trajectory; induced solution on instance G or LinkGraph $\mathcal{G}$
$\mathcal{D}$, Expand, LocalSearch	Heatmap decoding, LinkGraph unpacking, and lightweight local improvement
I_G, I_L	Numbers of global (PRF) and local (SymNCO) search iterations
I_{RS}	Number of RS iterations for learning-free RS-RLD on MCut