NRP4CO: NEURAL RESIDUAL PROPAGATION FOR COMBINATORIAL OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Partial solutions guide prediction and search in neural combinatorial optimization, where solvers must decide which assignments to retain before completing a solution. Ranking-based completion uses relative score order, while committing assignments at a fixed threshold depends on score magnitudes. We propose NRP4CO, a framework for Neural Residual Propagation that couples learned commitments with deterministic constraint propagation. A Transformer conditioned on a partial solution produces separate edit and residual scores: edit scores guide primary selection and completion, while residual scores receive assignment supervision for thresholded commitment on unresolved variables. Problem-specific propagation rules derive implied assignments, and auxiliary propagation labels train the shared representation to encode these consequences. NRP4CO extends partial solutions and reconstructs incumbents during search. We prove that admissible commitments followed by sound propagation preserve the existence of a feasible completion. We evaluate NRP4CO on the traveling salesman problem (TSP), asymmetric traveling salesman problem (ATSP), capacitated vehicle routing problem (CVRP), maximum independent set (MIS), maximum clique (MCL), and minimum vertex cover (MVC). NRP4CO achieves the lowest optimality gap among the compared neural solvers on all 19 benchmark settings. Across TSP, ATSP, and maximum clique benchmarks, NRP4CO reduces optimality gaps by an average of 24.0%, 69.6%, and 79.0%, respectively, with corresponding average speedups of $11.06\times$, $4.46\times$, and $7.38\times$ over the neural baselines with the best reported gap on each dataset.

1 INTRODUCTION

Combinatorial optimization (CO) seeks a solution that satisfies discrete constraints and optimizes an objective, such as route length or the size of an independent set. Many CO problems are NP-hard (Karp, 2010). Neural combinatorial optimization (NCO) learns decisions from reference solutions or objective feedback (Bengio et al., 2021; Vinyals et al., 2015). Neural solvers construct solutions through successive choices (Kool et al., 2018; Kwon et al., 2020), decode global predictions (Joshi et al., 2019; Qiu et al., 2022), or improve existing candidates (Chen & Tian, 2019; Hottung et al., 2021). In each case, useful predictions must translate into feasible decisions at an affordable computational cost.

Partial solutions provide context for these predictions. COExpander (Ma et al., 2025a) alternates prediction with adaptive expansion, selecting variables and propagating assignments forced by the constraints. MaskCO (Chen et al., 2026) learns to recover reference solutions from retained selections and reuses this mapping for reconstruction during search. NEXCO (Wang et al., 2026) conditions masked diffusion on a partial solution, using confidence and feasibility projection to control iterative unmasking. These methods learn which values should guide the next stage of construction or reconstruction.

Completion and retained commitment use scores in different ways. A completion rule that depends only on score order needs relative priorities among the remaining choices. A commitment rule based on a threshold must also determine whether the evidence for an assignment is strong enough to fix it before completion. Two score vectors can induce the same ordering yet select different assignments

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.

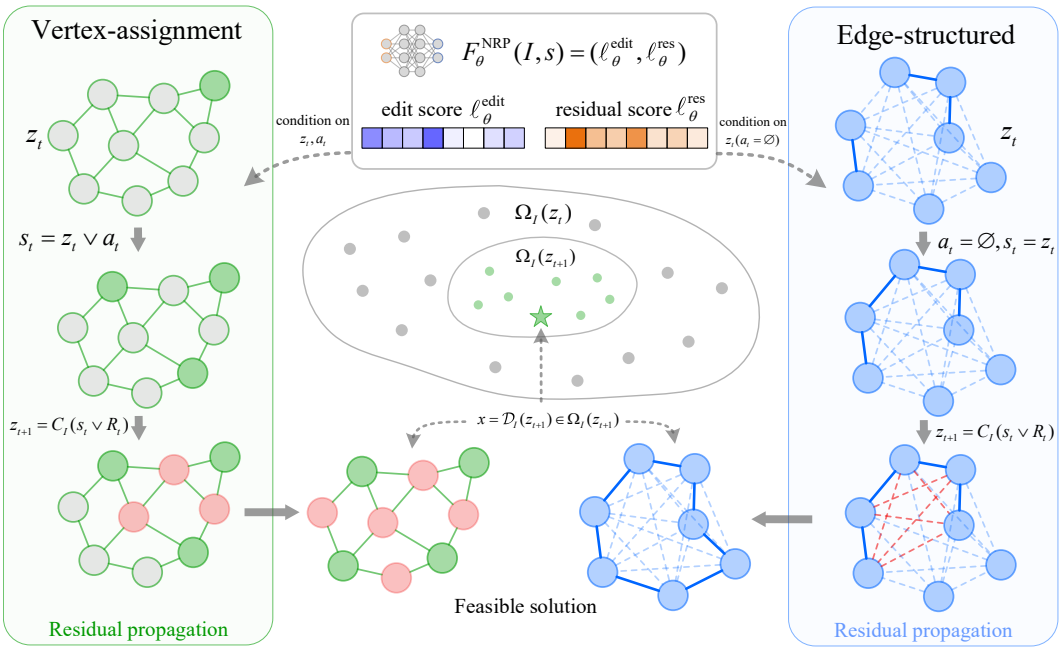


Figure 1: Overview of NRP4CO. For vertex-assignment problems, a primary edit conditions residual selection; for edge-structured problems, reconstruction supplies a closed partial state. In both cases, residual commitments are followed by deterministic constraint propagation and completion to a feasible solution.

at the same threshold. This distinction motivates assignment supervision for commitment scores alongside the relative priorities used for completion.

We propose **N**eural **R**esidual **P**ropagation (NRP), summarized in Figure 1. NRP4CO implements it with separate edit and residual outputs from a Transformer. Edit scores guide primary selection and completion, while residual scores select additional assignments on unresolved variables before completion. Assignment supervision trains residual scores for thresholded selection. Problem-specific propagation rules derive the consequences of accepted commitments, and auxiliary propagation labels train the shared representation to encode these consequences. Completion then preserves both the selected assignments and their propagated consequences.

The conditioning state follows the decision encoding. For vertex-assignment problems, a primary edit is supplied to the model before residual selection, allowing the next prediction to depend on that choice. For edge-structured problems, reconstruction supplies a closed partial assignment from an incumbent solution. Residual scores select commitments in this state. After propagation, a new query supplies edit scores for completion. NRP4CO develops realizations for TSP, ATSP, CVRP, MIS, MCL, and MVC, with a separate model for each problem class. Additional experiments on maximum cut (MCut) are presented in Appendix H. Our contributions are:

1. We introduce NRP4CO, which combines separate edit and residual predictions with deterministic constraint propagation. We show that score vectors with identical rankings can induce different thresholded commitments, motivating distinct supervision for completion and commitment.

2. We formalize NRP through partial assignments and sound closure and prove feasibility preservation under admissible commitments. For chains that retain their commitments and resolve at least one variable per active round, we bound the number of active rounds by the initial number of unresolved variables.

3. We evaluate NRP4CO on 19 benchmark settings across six CO problems, achieving the lowest gap among the compared neural solvers on all 19 settings. Against the neural baselines with the best reported gap on each dataset, average gap reductions on TSP, ATSP, and MCL are 24.0%, 69.6%, and 79.0%, with corresponding average speedups of $11.06\times$, $4.46\times$, and $7.38\times$.

2 Related Work

Learning from Partial Solutions and Adaptive Expansion. Local Construction solvers such as AM (Kool et al., 2018), POMO (Kwon et al., 2020), and Sym-NCO (Kim et al., 2022) choose successive feasible actions, while Global Prediction solvers such as GCN (Joshi et al., 2019), DIMES (Qiu et al., 2022), and Att-GCRN (Fu et al., 2021) score a full decision field before decoding. Methods that use partial solutions connect these endpoints but learn different decision roles. COExpander (Ma et al., 2025a) and NEXCO (Wang et al., 2026) ask which unresolved variables should be fixed next. MaskCO (Chen et al., 2026) asks how missing decisions should be recovered from retained context. NRP4CO asks which predicted assignments should be retained before a separate completion step. This question motivates independent edit and residual readouts, assignment supervision for residual scores, and explicit propagation after commitments are accepted.

Neural Reconstruction and Search. Neural rewriting methods such as Learn2Opt (Chen & Tian, 2019) and models that search in the latent space of a variational autoencoder (Hottung et al., 2021) improve an incumbent through learned moves. DIFUSCO (Sun & Yang, 2023) generates candidates through iterative denoising. T2T (Li et al., 2023) and Fast T2T (Li et al., 2024) add objective guidance or optimization consistency to this process. GenSCO (Li et al., 2025) trains a map that improves solutions and invokes generation repeatedly as a search operator. MaskCO (Chen et al., 2026) also reopens part of a solution before predicting its missing decisions. NRP4CO uses reconstruction within a neighborhood search (Shaw, 1998) to produce a partial assignment from the incumbent. Residual selection then commits additional assignments, propagation records their consequences, and completion preserves the resulting state.

Constraint Propagation and State Abstraction. Constraint programming derives implied assignments through sound propagation rules (Apt, 1999; Schulte & Stuckey, 2008). Saturating a rule family gives a closure operator (Davey & Priestley, 2002) that makes these consequences explicit without changing the feasible completion set. NEXCO (Wang et al., 2026) applies feasibility projection while adding candidates in confidence order. NRP4CO instead separates learned commitment from consequence derivation: residual scores choose an admissible edit set, and deterministic closure records everything forced by that set. Closed states also remove redundant representations of the same propagated information, paralleling state abstraction in BQ-NCO (Drakulic et al., 2023). The appendix proves that every closure class has a unique canonical representative.

3 Partial Assignments and Constraint Propagation

3.1 Partial Assignments and Admissible Edits

For an instance I, let U_I be its decision units and $\mathcal{X}_I \subseteq \{0,1\}^{U_I}$ its feasible solutions. Units are edges or arcs in edge-structured problems and vertices in vertex-assignment problems. We write the objective as $\min_{x \in \mathcal{X}_I} f_I(x)$, negating native maximization objectives.

A partial assignment $s \in \{-1,0,1\}^{U_I}$ records unresolved units as -1 and committed values as 0 or 1. Both committed values are retained under the information order:

$$s \preceq t \quad \iff \quad \forall u \in U_I,\ s(u) \neq -1 \implies t(u) = s(u). \tag{1}$$

We denote the unresolved units by $\mathcal{U}(s) = \{u \in U_I : s(u) = -1\}$.

The completion set is $\Omega_I(s) = \{x \in \mathcal{X}_I : s \preceq x\}$. A state is valid when this set is nonempty. We write $\mathcal{S}_I$ for all valid partial assignments. Adding commitments narrows the completion set: $s \preceq t$ implies $\Omega_I(t) \subseteq \Omega_I(s)$.

An edit set A assigns binary values to units in $D_A \subseteq \mathcal{U}(s)$. Applying it gives the compatible join $s \vee A$, which agrees with A on D_A and with s elsewhere.

The edit set is admissible when $\Omega_I(s \vee A) \neq \varnothing$. This condition concerns all assignments in the set together. From an empty MIS assignment, either endpoint of an edge can be selected individually, but selecting both is inadmissible. Each problem supplies an admissible commitment family $\mathcal{A}_I(s)$ whose members satisfy this condition. The selection rules in Section 4 update compatibility as commitments accumulate so that their combined edits remain in $\mathcal{A}_I(s)$.

3.2 CONSTRAINT PROPAGATION

For each problem class, we specify sound propagation rules: every assignment they add holds in every feasible completion of the input. Applying these rules until none changes the state computes the exact closure $C_I : \mathcal{S}_I \to \mathcal{S}_I$, where exact denotes saturation of the specified rules. Soundness gives $\Omega_I(C_I(s)) = \Omega_I(s)$, so propagation makes implied assignments explicit while preserving the completion set.

For MIS on the path $u - v - w$, selecting u forces $v = 0$, giving $C_I(1, -1, -1) = (1, 0, -1)$. The value of w remains a choice. NRP uses a closed state z, satisfying $C_I(z) = z$, between propagation rounds. The state representation therefore records both learned commitments and their propagated consequences. Appendix B.2 gives the closure properties, and Appendix A.2 illustrates the transitions for different problems.

4 NEURAL RESIDUAL PROPAGATION

NRP4CO uses residual scores to select commitments and edit scores to complete the resulting partial assignment. Propagation supplies the consequences of each selected set. These operations realize the NRP paradigm through learned commitment, deterministic propagation, and completion. Formally, each problem is represented by

$$\mathfrak{P}_I = (U_I, \mathcal{X}_I, C_I, \mathcal{A}_I, R_\tau, D_I), \tag{2}$$

where C_I is sound closure, $\mathcal{A}_I$ contains the permitted jointly admissible edits, R_τ selects residual commitments, and D_I completes a valid closed state using edit scores while retaining its assignments. For each problem, this representation specifies the decision encoding, C_I, $\mathcal{A}_I$, and D_I. The learned selector satisfies the admissibility condition $a_t \cup R_t \in \mathcal{A}_I(z_t)$. This formalization isolates the structure of each problem while retaining a common transition across all six classes.

4.1 RESIDUAL PROPAGATION

At a closed state z_t, a residual selector chooses assignments to retain before completion. For vertex-assignment problems, edit scores first select a primary edit a_t. The model then receives $s_t = z_t \vee a_t$ and predicts residual scores. This additional query updates the prediction using the primary choice. The input is taken before saturation: consequences of the primary edit remain unresolved in the input, and auxiliary assignment supervision trains the representation to predict them. Compatibility tests use the primary edit and its consequences when determining which residual assignments are eligible.

For edge-structured problems, reconstruction supplies the closed partial state and $a_t = \varnothing$. The residual output selects assignments to commit in this state. After selection and propagation, the model is queried again on the resulting state, and its edit output scores connections for completion. Thus both problem categories learn commitments that constrain completion, with the context supplied by a primary edit or by reconstruction.

Algorithm 1: Neural residual propagation transition

Input: Instance I, valid closed state z_t, scorer θ, residual threshold τ

1: Select a primary edit a_t using edit scores, or set $a_t = \varnothing$ for edge-structured problems.
2: $s_t \leftarrow z_t \vee a_t$
3: Query residual scores $\ell_\theta^{\text{res}}(I, s_t)$.
4: Select $R_t = R_\tau(I, z_t, a_t)$ using these scores, threshold τ, and the compatibility rules for I.
5: $z_{t+1} \leftarrow C_I(s_t \vee R_t)$
6: **return** z_{t+1}

The residual selector $R_\tau(I, z_t, a_t)$ ranks eligible unresolved assignments and accepts those passing a threshold τ. Its domain satisfies $D_{R_t} \subseteq \mathcal{U}(z_t \vee a_t)$. The permitted assignments follow the problem encoding. MIS and MCL select at most one additional vertex. MVC selects at most one exclusion. TSP selects a compatible pair of edges. ATSP selects one arc for inclusion or exclusion. CVRP accepts connections in score order and updates eligibility after each addition.

For a jointly admissible edit set $a_t \cup R_t$ in the sense of Section 3.1, the propagation transition is

$$z_{t+1} = C_I(z_t \vee a_t \vee R_\tau(I, z_t, a_t)) . \tag{3}$$

The selected assignments narrow the set of feasible completions, and closure records their consequences. Algorithm 1 gives the transition. The residual scores select the commitments. C_I computes the propagated state.

Compatibility is updated as assignments accumulate. MVC uses a closed copy of s_t to identify eligible exclusions while retaining the raw s_t for residual prediction. CVRP applies closure between residual additions and reuses the same scores. If no candidate passes the threshold, $R_t = \varnothing$. An empty combined edit set leaves the state unchanged. Appendix C specifies the candidate scores, thresholds, and compatibility rules.

4.2 Scoring and Training

We train on pairs $(I, x^\star)$ of an instance and a reference solution. Sampling retained reference decisions yields partial assignments with different amounts of context, following masked reconstruction for neural CO (Chen et al., 2026). The reference supplies assignment labels, while the propagation rules supply auxiliary labels. Training includes states before saturation, matching the residual query after a primary edit. Appendix D.2 specifies the sampling procedures and the additional perturbed contexts used for ATSP.

The need for a separate commitment signal follows from the information consumed by the two policies. Let M be a nonempty finite candidate set. Suppose a completion policy π_{comp} depends only on the weak ordering induced by a score vector $\ell \in \mathbb{R}^M$. A commitment policy that applies a fixed threshold is

$$\pi_{\mathrm{commit}}(\ell; \tau) = \{u \in M : \ell_u > \tau\}. \tag{4}$$

Proposition 1 (Score order does not determine threshold selection)**.** For every score vector ℓ and every strictly increasing function $h : \mathbb{R} \to \mathbb{R}$,

$$\pi_{\mathrm{comp}}(\ell) = \pi_{\mathrm{comp}}(h \circ \ell).$$

For any fixed τ, there exist strictly increasing affine maps h_- and h_+ such that

$$\pi_{\mathrm{commit}}(h_- \circ \ell; \tau) = \varnothing, \qquad \pi_{\mathrm{commit}}(h_+ \circ \ell; \tau) = M.$$

Hence score order is sufficient to preserve the completion decision, but it does not determine which candidates pass a fixed commitment threshold.

The proof is in Appendix B.1. Commitment therefore needs a scale anchored to assignment semantics in addition to relative priority.

A Transformer (Vaswani et al., 2017) conditioned on the instance and partial assignment produces the core decision outputs

$$F_\theta^{\mathrm{NRP}}(I, s) = (\ell_\theta^{\mathrm{edit}}, \ell_\theta^{\mathrm{res}}). \tag{5}$$

Its shared features feed independent edit and residual readouts. Edge-structured problems use pairwise outputs, while vertex-assignment problems use one output per vertex. Edit supervision follows the primary-selection and completion rule. Whenever residual selection uses a threshold, binary classification anchors its assignment scale, with ranking added when the selector also compares eligible candidates. The independent readouts allow these objectives to shape different score fields while retaining a shared representation. For vertex-assignment problems, they also consume different conditioning states, z_t and $z_t \vee a_t$. Table 8 gives the objective for every problem.

The auxiliary output $\ell_\theta^{\mathrm{close}}$ is representation supervision rather than a decision in the NRP transition. Its labels identify assignments forced by degree, connectivity, or adjacency constraints. Entries outside the implication labels receive a neutral target as specified in Appendix D.3. This signal trains the representation used by both decision outputs, while C_I performs exact propagation during solving. CVRP auxiliary labels include required connections as well as exclusions. Its inference closure uses exclusion rules.

The state prediction q_θ is optional search guidance and is not part of the NRP transition. The objectives are organized accordingly:

$$\begin{aligned} \mathcal{L}_{\mathrm{decision}} &= \mathbb{E}_{(I,x^\star),s}\left[\mathcal{L}_{\mathrm{edit}} + \lambda_{\mathrm{res}}\mathcal{L}_{\mathrm{res}}\right], \\ \mathcal{L}(\theta) &= \mathcal{L}_{\mathrm{decision}} + \lambda_{\mathrm{close}}\mathbb{E}[\mathcal{L}_{\mathrm{close}}] + \lambda_{\mathrm{state}}\mathbb{E}[\mathcal{L}_{\mathrm{state}}]. \end{aligned} \tag{6}$$

Setting $\lambda_{\mathrm{state}} = 0$ removes state guidance without changing residual propagation or completion. When enabled, the state loss fits the normalized reference structure missing from the context and supplies an optional acceptance score. Its target and use are given in Appendices D.2 and D.4. Models are trained separately for each problem class. CVRP shares parameters between its route representation and its representation of customer order, with edit and state supervision on the latter. Appendices D.1 and D.3 gives the architecture, masks, and complete objectives.

4.3 TEST-TIME SEARCH

Search maintains S parallel trajectories, each initialized by scoring and completing an empty assignment to obtain a feasible incumbent x_j. Subsequent cycles extend a retained partial state or reconstruct one from the incumbent. Reconstruction retains selected parts of the solution and reopens the remaining decisions. Residual selection commits assignments in this context, and completion constructs a candidate that retains them before local improvement. Perturbed score rankings diversify the trajectories.

TSP and ATSP reconstruction retains connections from a source tour at 1, leaves other decision units unresolved, and applies closure. Completion joins the resulting paths into a tour, followed by 2-opt. TSP uses edit scores to join compatible paths. ATSP solves a constrained linear assignment and merges the cycles. CVRP completes route components under capacity constraints and also proposes customer orders followed by a feasible split. For vertex-assignment problems, completion extends the selected independent set, clique, or cover exclusions. Appendix C gives these procedures.

With the default search setting, a candidate x' replaces the incumbent when

$$f_I(x') < f_I(x_j) - \varepsilon, \tag{7}$$

where $\varepsilon \geq 0$ is a numerical tolerance. Each proposal is compared with the current incumbent, including earlier accepted updates in the same cycle. After at most C cycles, the solver returns the solution with the smallest objective among the final incumbents. Appendix D.4 gives the proposal order and the optional acceptance rule that uses the predicted state value.

After initialization, TSP and ATSP use two model query stages per cycle. CVRP uses three. Vertex-assignment problems query the raw state after the primary edit and the closed state after residual selection, with an additional query when a state is reopened. Queries are batched across trajectories. Propagation, completion, and local improvement contribute additional computation. Appendix D.4 gives the complete proposal order.

4.4 FEASIBILITY AND PROPAGATION PROGRESS

The formalization separates learned selection from the conditions required for feasibility: the selector chooses within $\mathcal{A}_I$, and closure may add only sound consequences.

Proposition 2 (NRP transitions preserve feasibility)**.** Let $z_0 \in \mathcal{S}_I^{\mathrm{cl}}$, let C_I be a sound closure operator, and let every selected set $A_t = a_t \cup R_t$ belong to $\mathcal{A}_I(z_t)$. Then

$$z_{t+1} = C_I(z_t \vee A_t) \quad \Longrightarrow \quad z_{t+1} \in \mathcal{S}_I^{\mathrm{cl}} \text{ and } \Omega_I(z_{t+1}) \neq \varnothing.$$

If every active round commits at least one unresolved unit and no round reopens a commitment, the number T of active rounds satisfies

$$T \leq |\mathcal{U}(z_0)|.$$

The proof is in Appendix B.3. For vertex-assignment problems, membership in $\mathcal{A}_I$ follows from the conditions for independent sets, cliques, or exclusions from a vertex cover. For edge-structured problems, it follows from the path and capacity conditions maintained during reconstruction and residual selection. The ATSP argument covers one signed residual assignment followed by constrained assignment and cycle merging. Appendix B.4 proves admissibility for each problem.

We account for assignments made within a round by separating the selected set $A_t = a_t \cup R_t$ from its propagated consequences:

$$\begin{aligned} g_t &= |\mathcal{U}(z_t)| - |\mathcal{U}(z_{t+1})| = |D_{A_t}| + c_t, \\ c_t &= |\mathcal{U}(z_t \vee A_t)| - |\mathcal{U}(z_{t+1})|. \end{aligned} \tag{8}$$

Here c_t counts assignments derived by closure, and g_t counts the total assignments resolved in the round. To isolate the additional progress of a residual, Appendix B.5 also compares the resulting state with closure of the primary edit alone. The same appendix accounts for the length of a finite propagation chain and gives a conditional bound on its number of rounds. Model queries and the completion work are counted separately in Appendix D.4.

Table 1: Results on TSP and ATSP. * denotes reported results from prior work.

Method	TSP-50			TSP-100			TSP-500			TSP-1000		
	Obj.↓	Gap↓	Time↓	Obj.↓	Gap↓	Time↓	Obj.↓	Gap↓	Time↓	Obj.↓	Gap↓	Time↓
Classical Reference Solvers												
Concorde* (Applegate et al., 2006)	5.69	–	0.06s	7.76	–	0.24s	16.55	–	18.67s	23.12	–	84.41s
LKH (500)* (Helsgaun, 2017)	5.69	0.00%	0.06s	7.76	0.00%	0.18s	16.55	0.00%	1.85s	23.12	0.01%	4.64s
Learning-based Solvers with Direct Decoding												
GCN* (Joshi et al., 2019)	5.69	0.07%	0.01s	7.78	0.24%	0.01s	16.77	1.35%	0.06s	23.53	1.77%	0.23s
DIMES* (Qiu et al., 2022)	5.89	3.58%	0.01s	8.11	4.54%	0.01s	17.66	6.71%	0.31s	24.91	7.74%	0.66s
LEHD PRC 100* (Luo et al., 2023)	–	–	–	7.76	0.01%	0.64s	16.61	0.34%	3.75s	23.44	1.22%	20.16s
RL4CO (SymNCO)* (Berto et al., 2025)	5.73	0.68%	0.17s	7.89	1.75%	0.32s	–	–	–	–	–	–
BQ-NCO + Beam-16* (Drakulic et al., 2023)	5.79	1.72%	0.24s	7.87	1.48%	0.90s	16.77	1.33%	4.02s	23.51	1.71%	10.34s
Learning-based Solvers with Test-Time Scaling												
DIFUSCO (T_s=50) (Sun & Yang, 2023)	5.69	0.13%	0.36s	7.78	0.28%	0.47s	16.82	1.64%	1.14s	23.57	1.94%	4.01s
T2T (T_s=50,T_g=30) (Li et al., 2023)	5.69	0.02%	1.09s	7.76	0.07%	1.43s	16.68	0.82%	3.10s	23.44	1.40%	9.38s
Fast T2T (T_s=5,T_g=5) (Li et al., 2024)	5.69	0.01%	0.29s	7.76	0.03%	0.33s	16.61	0.39%	1.50s	23.25	0.58%	6.18s
StruDiCO (T_s=3/5,T_g=3/5)* (Wang et al., 2025)	5.69	0.01%	0.06s	7.76	0.02%	0.08s	16.60	0.33%	0.67s	23.24	0.54%	2.61s
COExpander (D_s=3,T_s=5) (Ma et al., 2025a)	5.69	0.02%	0.11s	7.76	0.04%	0.20s	16.63	0.52%	0.62s	23.34	0.95%	2.30s
NEXCO (D_s=3) (Wang et al., 2026)	–	–	–	7.76	0.03%	0.22s	16.62	0.42%	1.31s	23.28	0.71%	3.12s
NEXCO (D_s=7) (Wang et al., 2026)	–	–	–	7.76	0.02%	0.49s	16.58	0.22%	3.26s	23.26	0.60%	6.62s
NRP4CO (S=1,C=16)	5.69	0.002%	0.02s	7.76	0.011%	0.02s	16.59	0.237%	0.29s	23.19	0.315%	0.89s
NRP4CO (S=1,C=64)	5.69	0.001%	0.04s	7.76	0.009%	0.08s	16.57	0.175%	1.00s	23.17	0.203%	2.71s
NRP4CO (S=16,C=64)	5.69	-0.000%	0.43s	7.76	0.000%	1.30s	16.57	0.168%	9.90s	23.16	0.183%	38.17s

Method	ATSP-50			ATSP-100			ATSP-200			ATSP-500		
	Obj.↓	Gap↓	Time↓	Obj.↓	Gap↓	Time↓	Obj.↓	Gap↓	Time↓	Obj.↓	Gap↓	Time↓
LKH (1000)* (Helsgaun, 2017)	1.55	0.00%	0.10s	1.57	0.00%	0.24s	1.56	0.00%	0.72s	1.57	0.00%	4.38s
MatNet ($\times$16)* (Kwon et al., 2021)	1.56	0.30%	0.04s	1.59	1.58%	0.07s	3.73	138.40%	0.16s	–	–	–
GOAL + Beam-16* (Drakulic et al., 2025)	1.63	4.74%	0.35s	1.62	3.53%	0.91s	1.61	2.86%	4.69s	1.70	8.13%	32.24s
COExpander (S=4,D_s=3,T_s=5) (Ma et al., 2025a)	1.56	0.17%	0.20s	1.58	0.95%	0.71s	1.59	1.50%	2.64s	1.60	1.57%	16.48s
NRP4CO (S=64,C=1)	1.56	0.091%	0.05s	1.57	0.126%	0.15s	1.57	0.293%	0.64s	1.58	0.553%	3.34s
NRP4CO (S=256,C=1)	1.56	0.035%	0.18s	1.57	0.063%	0.52s	1.57	0.201%	2.29s	1.58	0.520%	12.10s
NRP4CO (S=512,C=1)	1.55	0.021%	0.35s	1.57	0.044%	1.02s	1.57	0.158%	4.14s	1.58	0.501%	23.25s

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Problems and datasets. We evaluate NRP4CO on three edge-structured problems and three vertex-assignment problems. The edge-structured benchmarks are uniform Euclidean TSP-50/100/500/1000, ATSP-50/100/200/500 represented by full directed cost matrices, and uniform CVRP-50/100/200/500. The vertex-assignment benchmarks are RB-[200-300], RB-[800-1200], and ER-[700-800] for MIS and RB-SMALL/LARGE for MCL and MVC, following the benchmark families in ML4CO-Bench-101 (Ma et al., 2025b). The TSP, ATSP, and CVRP test sets contain 1,280/1,280/128/128, 2,500/2,500/100/100, and 10,000/10,000/100/100 instances across their four scales. Each vertex-assignment test set contains 500 instances, except ER-[700-800], which contains 128. For MIS, RB-[200-300] uses 90,000 training instances, while ER-[700-800] uses 163,840. ER instances have 700–800 vertices and independent edge probability 0.15, following NEXCO (Wang et al., 2026). We also evaluate the TSP and CVRP models on TSPLIB (Reinelt, 1991) and CVRPLIB (Uchoa et al., 2017); Appendices F.4 and F.5 gives the settings and results. The test sets, reference objectives, and run configurations are summarized in Appendices E.1 and E.2.

Reference solutions and metrics. Reference objectives are produced by Concorde (Applegate et al., 2006) for TSP, LKH (Helsgaun, 2017) with 1,000 trials for ATSP, HGS (Vidal et al., 2012) for CVRP, KaMIS (Lamm et al., 2016) for MIS, and Gurobi (Gurobi Optimization, 2023) with a time limit for MCL and MVC. We report the mean objective (Obj.), relative optimality gap (Gap), and average inference time per instance (Time). Using the minimized objective convention $\tilde{f}_I$ from Section 3, the gap is $100(\tilde{f}_I(x) - \tilde{f}_I(x^{\mathrm{ref}}))/|\tilde{f}_I(x^{\mathrm{ref}})|$. A negative gap means that a method improves upon the reference objective obtained within the solver time limit. Runtime includes neural queries, residual selection, deterministic closure, completion, and local improvement.

Model and search settings. We train a separate NRP4CO model for each problem class using the architecture and supervision in Appendices D.1 to D.3. At test time, S is the number of parallel trajectories and C is the cycle limit. The tables include several (S, C) pairs to show the balance

Table 2: Results on CVRP across problem sizes. * denotes reported results from prior work.

Method	CVRP-50			CVRP-100			CVRP-200			CVRP-500		
	Obj.↓	Gap↓	Time↓	Obj.↓	Gap↓	Time↓	Obj.↓	Gap↓	Time↓	Obj.↓	Gap↓	Time↓
HGS* (Vidal et al., 2012)	10.37	0.00%	1.01s	15.56	0.00%	20.03s	19.63	0.00%	60.02s	37.15	0.00%	360.38s
RL4CO (SymNCO)* (Berto et al., 2025)	10.56	1.91%	0.09s	15.93	2.38%	0.17s	20.19	2.88%	0.34s	38.70	4.17%	0.88s
COExpander (D_s=3) (Ma et al., 2025a)	10.77	3.90%	0.04s	16.22	4.25%	0.06s	20.59	4.89%	0.15s	39.12	5.34%	0.76s
NEXCO (D_s=3) (Wang et al., 2026)	10.47	0.95%	0.08s	15.79	1.46%	0.18s	20.14	2.57%	0.52s	–	–	–
NEXCO (D_s=5) (Wang et al., 2026)	10.45	0.76%	0.18s	15.76	1.24%	0.18s	20.06	2.16%	0.97s	–	–	–
NRP4CO (S=1,C=16)	10.43	0.635%	0.04s	15.74	1.118%	0.11s	19.93	1.512%	0.68s	38.26	2.949%	4.25s
NRP4CO (S=1,C=64)	10.41	0.421%	0.15s	15.68	0.724%	0.42s	19.83	1.026%	2.14s	38.01	2.280%	13.83s
NRP4CO (S=16,C=64)	10.37	0.069%	1.88s	15.60	0.227%	5.67s	19.74	0.562%	23.85s	37.89	1.952%	91.62s

between solution quality and runtime. Training and evaluation use an Intel Xeon Gold 6348 CPU at 2.60GHz and one NVIDIA A100 40GB GPU.

5.2 Solution Quality across Six CO Problems

Tables 1 to 3 and 5 report solution quality and runtime on 19 combinations of datasets and problem sizes. NRP4CO achieves the best gap among the listed learning-based methods on all 19 combinations. For the compared baseline and NRP4CO configurations with gaps g_b, g_n and runtimes t_b, t_n, we define $\text{Gap Reduction} = 100(g_b - g_n)/|g_b|$ and $\text{Speedup} = t_b/t_n$. Positive Gap Reduction denotes a smaller gap, and Speedup above $1\times$ denotes faster inference.

TSP and ATSP. On TSP, NRP4CO with $S = 1, C = 16$ achieves a 27.2% Gap Reduction and $5.42\times$ speedup over the prior neural baseline with the best gap at each scale, using mean gaps and runtimes across the four sizes in Table 1. On TSP-1000, NRP4CO improves the NEXCO (Wang et al., 2026) result from 0.60% in 6.62s to 0.315% in 0.89s. For ATSP, NRP4CO with $S = 64, C = 1$ achieves a 74.6% mean Gap Reduction and $4.79\times$ mean speedup over COExpander (Ma et al., 2025a) across all four scales.

CVRP. The balance between solution quality and runtime changes with CVRP size. Against NEXCO (Wang et al., 2026) on CVRP-50/100/200 and RL4CO (Berto et al., 2025) on CVRP-500, NRP4CO with $S = 1, C = 16$ reduces the mean gap by 25.5%. Speedup decreases from $4.54\times$ on CVRP-50 to $0.21\times$ on CVRP-500. On CVRP-500, NRP4CO lowers the gap from 4.17% to 2.949%, and additional search reaches 1.952%. NRP4CO improves both gap and runtime on CVRP-50/100/200. On CVRP-500, lower gaps require more inference time.

Table 3: Results on MIS. * denotes reported results from prior work.

Method	RB-[200-300]			RB-[800-1200]			ER-[700-800]		
	Obj.↑	Gap↓	Time↓	Obj.↑	Gap↓	Time↓	Obj.↑	Gap↓	Time↓
KaMIS* (Lamm et al., 2016)	20.09	–	45.81s	43.00	–	56.97s	44.97	–	48.49s
GFlowNets* (Zhang et al., 2023)	19.18	4.57%	0.06s	37.48	13.14%	0.52s	41.14	8.53%	0.35s
Fast T2T (T_g=5,T_s=5)* (Li et al., 2024)	19.50	2.89%	0.41s	–	–	–	40.69	9.51%	1.03s
StruDiCO (T_s=5,T_g=5)* (Wang et al., 2025)	19.75	1.71%	0.18s	–	–	–	42.13	6.32%	0.55s
COExpander (S=4,D_s=20) (Ma et al., 2025a)	19.71	1.87%	0.35s	41.44	3.58%	3.82s	42.56	5.34%	2.05s
NEXCO (D_s=7) (Wang et al., 2026)	19.76	1.66%	0.14s	41.25	4.07%	1.00s	42.98	4.20%	0.56s
NRP4CO (S=32,C=4)	19.85	1.14%	0.18s	40.87	4.85%	1.80s	42.92	4.55%	1.25s
NRP4CO (S=64,C=4)	19.90	0.90%	0.30s	41.06	4.40%	3.17s	43.28	3.75%	2.22s
NRP4CO (S=64,C=16)	19.97	0.57%	0.87s	41.70	2.94%	9.93s	43.96	2.24%	6.23s

MIS. Across the three MIS datasets in Table 3, the best gaps reported by NRP4CO yield a 39.1% mean Gap Reduction relative to the best prior neural gap on each dataset. On ER-[700-800], the gap falls from 4.20% for NEXCO (Wang et al., 2026) to 2.24%, a 46.7% Gap Reduction. The mean runtime gives an $8.88\times$ speedup over KaMIS (Lamm et al., 2016). This setting takes longer than the neural baselines and remains much faster than the reference solver.

MCL and MVC. On MCL, the faster of the two NRP4CO settings in Table 5 reduces the mean gap of COExpander (Ma et al., 2025a) by 70.8% and gives a $5.80\times$ speedup across RB-SMALL/LARGE. Increasing the search budget lowers the two gaps to 0.00% and 0.66%. On MVC, the NRP4CO setting with the best gaps reduces the mean gap of COExpander (Ma et al., 2025a) by 79.2% and gives a $27.33\times$ mean speedup over Gurobi (Gurobi Optimization, 2023).

Table 4: Summary comparison of NRP4CO with each dataset's lowest-gap neural baseline. Details in Table 11.

Problem	Gap Reduction↑	Speedup↑
TSP	24.00%	11.06×
ATSP	69.56%	4.46×
CVRP	21.57%	1.95×
MIS	11.34%	0.64×
MCL	78.97%	7.38×
MVC	45.48%	0.71×

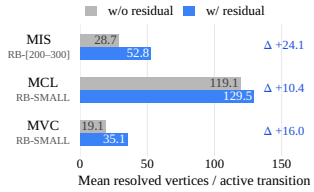


Figure 2: Propagation progress w/ & w/o residual selection.

Table 5: Results on MCL and MVC. * denotes reported results from (Ma et al., 2025a).

Maximum Clique (MCL)	RB-SMALL			RB-LARGE		
	Obj.↑	Gap↓	Time↓	Obj.↑	Gap↓	Time↓
Gurobi* (Gurobi Optimization, 2023)	19.08	0.00%	0.90s	40.18	0.00%	276.66s
Meta-EGN* (Wang & Li, 2023)	17.51	8.30%	0.27s	33.79	15.49%	0.54s
DiffUCO: CE (S=4,F=1)* (Sanokowski et al., 2024)	16.21	12.53%	1.41s	–	–	–
COExpander (S=1,D_s=20,T_s=1) (Ma et al., 2025a)	18.77	1.89%	0.11s	36.76	8.80%	0.41s
COExpander (S=4,D_s=5,T_s=20) (Ma et al., 2025a)	19.01	0.45%	1.74s	39.03	3.08%	18.67s
NRP4CO (S=32,C=1)	19.08	0.04%	0.19s	39.74	0.99%	3.33s
NRP4CO (S=64,C=8)	19.08	0.00%	0.83s	39.86	0.66%	25.54s
Minimum Vertex Cover (MVC)	**RB-SMALL**			**RB-LARGE**		
	Obj.↓	Gap↓	Time↓	Obj.↓	Gap↓	Time↓
Gurobi* (Gurobi Optimization, 2023)	205.76	0.00%	3.34s	968.23	0.00%	290.23s
Meta-EGN* (Wang & Li, 2023)	208.97	1.56%	0.30s	1010.69	4.40%	1.03s
COExpander (S=1,D_s=20,T_s=1) (Ma et al., 2025a)	206.75	0.48%	0.14s	969.92	0.18%	1.04s
COExpander (S=4,D_s=20,T_s=1) (Ma et al., 2025a)	206.51	0.37%	0.15s	969.81	0.16%	1.35s
NRP4CO (S=64,C=1)	205.88	0.06%	0.17s	969.67	0.15%	2.41s
NRP4CO (S=64,C=8)	205.81	0.02%	0.56s	969.10	0.09%	10.18s

5.3 Ablation Studies

Table 6: Ablation studies of residual selection and deterministic constraint propagation.

Method	TSP-1000		CVRP-500		MCL RB-LARGE	
	Gap↓	Time↓	Gap↓	Time↓	Gap↓	Time↓
NRP4CO	0.315%	0.888s	2.949%	4.250s	0.993%	3.333s
w/o residual selection	0.377%	0.852s	3.046%	4.160s	1.074%	1.075s
w/o constraint propagation	0.370%	1.912s	3.106%	4.178s	1.057%	1.072s

We ablate residual selection and deterministic constraint propagation on TSP-1000, CVRP-500, and MCL RB-LARGE. Removing residual selection sets $R_t = \varnothing$ in every transition. The propagation ablation replaces C_I with the identity mapping while retaining compatibility checks and feasible completion. TSP-1000 and CVRP-500 use $S = 1, C = 16$, and MCL RB-LARGE uses $S = 32, C = 1$. Both components reduce the final optimality gap across the three benchmarks in Table 6. Without residual selection, the gap rises from 0.315% to 0.377% on TSP-1000, from 2.949% to 3.046% on CVRP-500, and from 0.993% to 1.074% on MCL RB-LARGE. Identity closure raises the corresponding gaps to 0.370%, 3.106%, and 1.057%.

To examine how residual selection affects propagation in vertex-assignment problems, we hold the primary edit fixed in each active candidate transition and evaluate closure with and without the residual assignment. Figure 2 shows that residual selection increases the mean number of resolved units per active transition from 28.7 to 52.8 on MIS, from 119.1 to 129.5 on MCL, and from 19.1 to 35.1 on MVC. This additional progress includes the residual assignment and the further consequences it induces beyond closure of the primary edit, as defined in Appendix B.5.

6 Conclusion

We have introduced NRP4CO, a Neural Residual Propagation framework for edge-structured problems and vertex-assignment problems. The model predicts separate edit and residual scores for completion and thresholded commitment, with assignment supervision for the residual output. Deterministic constraint propagation derives the consequences of accepted commitments, and completion retains the resulting assignments when constructing a feasible solution. This mechanism supports both extension of partial solutions and reconstruction during search. Our analysis establishes that admissible commitments followed by sound propagation preserve the existence of a feasible completion. Experiments on six CO problems show that NRP4CO achieves the lowest optimality gap among the compared neural solvers across all 19 benchmark settings. On ATSP-500, NRP4CO reduces the gap from 1.57% to 0.553% and inference time from 16.48s to 3.34s relative to COExpander. Ablation studies show that residual selection and deterministic constraint propagation each improve solution quality on TSP-1000, CVRP-500, and MCL RB-LARGE. The propagation analysis shows that residual assignments increase the mean number of resolved variables per active transition on MIS, MCL, and MVC.

AI USE STATEMENT

In this work, we used generative AI tools for no tasks with required disclosure. We have not used generative AI tools for any other tasks with required disclosure, and the rest of the required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for drafting parts of this research paper and editing the paper to improve readability. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work follows the ICLR Code of Ethics and studies methods for neural combinatorial optimization. The experiments use synthetic instances and public benchmark datasets for standard routing and graph problems. They involve no human subjects, personal or sensitive data, or applications with immediate safety implications. The studied methods and benchmarks raise no specific privacy, security, discrimination, or fairness concerns. The authors are responsible for the accuracy and integrity of the reported methods and results.

REPRODUCIBILITY STATEMENT

Sections 3 to 5 specify the partial-assignment formalization, residual-propagation transition, learning objective, test-time search, and experimental protocol. Appendices B to F provide the formal assumptions and proofs, problem-specific realizations, model architecture, supervision objectives, search procedures, dataset and reference-solution provenance, training configurations, ablations, and additional analyses. Section 5.1 and Appendix E.1 describe the synthetic and public benchmark instances and reference solvers used for evaluation. Section 5.1 and Appendices D.4 and E.2 report the hardware and run configurations. Code will be made publicly available upon acceptance.

REFERENCES

David Applegate, Robert Bixby, Vasek Chvatal, and William Cook. Concorde TSP solver. `https://www.math.uwaterloo.ca/tsp/concorde/index.html`, 2006.

Krzysztof R. Apt. The essence of constraint propagation. *Theoretical Computer Science*, 221(1–2): 179–210, 1999.

Yoshua Bengio, Andrea Lodi, and Antoine Prouvost. Machine learning for combinatorial optimization: A methodological tour d'horizon. *European Journal of Operational Research*, 290(2): 405–421, 2021. doi: 10.1016/j.ejor.2020.07.063.

Federico Berto, Chuanbo Hua, Junyoung Park, Laurin Luttmann, Yining Ma, Fanchen Bu, Jiarui Wang, Haoran Ye, Minsu Kim, Sanghyeok Choi, et al. Rl4co: an extensive reinforcement learning for combinatorial optimization benchmark. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 5278–5289, 2025.

Lvda Chen, Yang Li, and Junchi Yan. Maskco: Masked generation drives effective representation learning and exploiting for combinatorial optimization. In *International Conference on Learning Representations*, volume 2026, pp. 133725–133753, 2026.

Xinyun Chen and Yuandong Tian. Learning to perform local rewriting for combinatorial optimization. *Advances in neural information processing systems*, 32, 2019.

Brian A Davey and Hilary A Priestley. *Introduction to lattices and order*. Cambridge university press, 2002.

Darko Drakulic, Sofia Michel, Florian Mai, Arnaud Sors, and Jean-Marc Andreoli. Bq-nco: Bisimulation quotienting for efficient neural combinatorial optimization. *Advances in Neural Information Processing Systems*, 36:77416–77429, 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.

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.

LLC Gurobi Optimization. Gurobi optimizer reference manual, 2023. URL `https://www.gurobi.com`.

Keld Helsgaun. An extension of the lin-kernighan-helsgaun tsp solver for constrained traveling salesman and vehicle routing problems. *Roskilde: Roskilde University*, 12:966–980, 2017.

André Hottung, Bhanu Bhandari, and Kevin Tierney. Learning a latent search space for routing problems using variational autoencoders. In *International Conference on Learning Representations*, 2021.

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.

Richard M. Karp. Reducibility among combinatorial problems. In *50 Years of Integer Programming 1958–2008: From the Early Years to the State-of-the-Art*, pp. 219–241. Springer, 2010. doi: 10.1007/978-3-540-68279-0_8.

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! *arXiv preprint arXiv:1803.08475*, 2018.

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.

Yeong-Dae Kwon, Jinho Choo, Iljoo Yoon, Minah Park, Duwon Park, and Youngjune Gwon. Matrix encoding networks for neural combinatorial optimization. *Advances in Neural Information Processing Systems*, 34:5138–5149, 2021.

Sebastian Lamm, Peter Sanders, Christian Schulz, Darren Strash, and Renato F Werneck. Finding near-optimal independent sets at scale. In *2016 Proceedings of the eighteenth workshop on algorithm engineering and experiments (ALENEX)*, pp. 138–150. SIAM, 2016.

Yang Li, Jinpei Guo, Runzhong Wang, and Junchi Yan. T2t: From distribution learning in training to gradient search in testing for combinatorial optimization. In *Advances in 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 *Advances in Neural Information Processing Systems*, 2024.

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.

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.

Jiale Ma, Wenzheng Pan, Yang Li, and Junchi Yan. Coexpander: Adaptive solution expansion for combinatorial optimization. In *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.

Ruizhong Qiu, Zhiqing Sun, and Yiming Yang. Dimes: A differentiable meta solver for combinatorial optimization problems. *Advances in Neural Information Processing Systems*, 35:25531–25546, 2022.

Gerhard Reinelt. Tsplib—a traveling salesman problem library. *ORSA journal on computing*, 3(4): 376–384, 1991.

Sebastian Sanokowski, Sepp Hochreiter, and Sebastian Lehner. A diffusion model framework for unsupervised neural combinatorial optimization. In *International Conference on Machine Learning*, pp. 43346–43367. PMLR, 2024.

Christian Schulte and Peter J Stuckey. Efficient constraint propagation engines. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 31(1):1–43, 2008.

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. *Advances in neural information processing systems*, 36:3706–3731, 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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 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.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Haoyu Wang and Pan Li. Unsupervised learning for combinatorial optimization needs meta-learning. *arXiv preprint arXiv:2301.03116*, 2023.

Yu Wang, Yang Li, Junchi Yan, and Yi Chang. StruDiCO: Structured denoising diffusion with gradient-free inference-stage boosting for memory and time efficient combinatorial optimization. In *Advances in Neural Information Processing Systems*, 2025.

Yu Wang, Yang Li, Jiale Ma, Junchi Yan, and Yi Chang. Native adaptive solution expansion for diffusion-based combinatorial optimization. In *International Conference on Learning Representations*, volume 2026, pp. 36466–36486, 2026.

Dinghuai Zhang, Hanjun Dai, Nikolay Malkin, Aaron C Courville, Yoshua Bengio, and Ling Pan. Let the flows tell: Solving graph combinatorial problems with gflownets. *Advances in neural information processing systems*, 36:11952–11969, 2023.

A Notation and Worked Examples

A.1 Notation Summary

We use the native minimization objective for edge-structured problems and MVC, and negate the set cardinality for MIS and MCL. Each decision unit counts once, including an undirected edge stored in two symmetric matrix entries.

Table 7: Notation for closed partial assignments, learning, and search.

Symbol	Meaning
I, U_I	Problem instance and its finite set of decision units.
$\mathcal{X}_I$, f_I	Feasible complete solutions and the minimization objective.
s, t	Partial assignments in $\{-1, 0, 1\}^{U_I}$, where -1 is unresolved and $0, 1$ are committed values.
$s \preceq t$	Every commitment of s is retained in t.
$\mathcal{U}(s)$	Unresolved units $\{u : s(u) = -1\}$.
$\Omega_I(s)$	Feasible completions $\{x \in \mathcal{X}_I : s \preceq x\}$.
$\mathcal{S}_I$	Valid partial assignments, for which $\Omega_I(s) \neq \varnothing$.
A, D_A	An edit set and the unresolved units to which it assigns values.
$s \vee A$	The compatible assignment obtained by applying A to s.
$\mathcal{A}_I(s)$	Family of jointly admissible commitment sets for problem I at state s.
C_I	Exact saturation of the propagation rules specified for problem I.
$\mathcal{S}_I^{\mathrm{cl}}$	Closed states, satisfying $C_I(z) = z$.
$s \sim_{C_I} t$	Closure equivalence: $C_I(s) = C_I(t)$.
$[s]_{C_I}$	The equivalence class represented by the closed image $C_I(s)$.
D_I	Completion operator that returns a feasible solution retaining the input commitments.
$\mathfrak{P}_I$	Formal representation $(U_I, \mathcal{X}_I, C_I, \mathcal{A}_I, R_\tau, D_I)$ of NRP.
z_t, s_t	Closed state at round t and intermediate state $s_t = z_t \vee a_t$.
a_t, R_t	Primary edit and additional residual commitments.
R_τ	Residual selector using learned scores, threshold τ, and compatibility rules.
A_t, g_t	Combined edits $a_t \cup R_t$ and progress $\lvert\mathcal{U}(z_t)\rvert - \lvert\mathcal{U}(z_{t+1})\rvert$.
c_t	Number of assignments derived by closure in round t.
Δ_t^{res}	Additional assignments resolved relative to closure of the primary edit alone.
m, T, $\bar{g}$	Initial unresolved count, propagation chain length, and a lower bound on conditional expected progress.
F_θ^{NRP}	Core neural scorer producing edit and residual decision scores.
$\ell_\theta^{\mathrm{edit}}$, $\ell_\theta^{\mathrm{res}}$	Scores for primary selection and completion, and for residual selection.
$\ell_\theta^{\mathrm{close}}$	Auxiliary assignment prediction supervised by propagation consequences or the target specified for the problem.
q_θ, $\hat{q}_\theta$	Optional raw state prediction and its sigmoid transform for search guidance.
$x^\star$, y^q	Reference solution and the state target defined relative to it.
x_j, v_j	Feasible incumbent on trajectory j and its associated state score.
$\Phi_\beta(x, v)$	Acceptance score $f_I(x) + \beta v$, with fixed $\beta \geq 0$.
S, C	Number of search trajectories and maximum search cycles per trajectory.
ε	Nonnegative tolerance in the strict acceptance test.

The operator C_I computes closure, C is the search cycle limit, and T counts rounds in one propagation chain.

A.2 Closure and Residual Transitions: Worked Examples

Independent set: a learned choice and a forced exclusion. Consider the path $u - v - w$, with assignments written in that vertex order. From $z_0 = (-1, -1, -1)$, a primary commitment $a_0 = \{u \mapsto 1\}$ gives the intermediate state $(1, -1, -1)$. Independence forces $v = 0$, so $C_I(1, -1, -1) = (1, 0, -1)$. The value of w remains a choice. A residual commitment $R_0 = \{w \mapsto 1\}$ is compatible with the primary edit, and

$$z_1 = C_I(z_0 \vee a_0 \vee R_0) = (1, 0, 1).$$

Two assignments come from learning and one from propagation, giving $g_0 = 3$. The states $(1, -1, -1)$ and $(1, 0, -1)$ have the same closed image and hence represent the same closure class. In contrast, the batch $\{u \mapsto 1, v \mapsto 1\}$ is inadmissible because it cannot be extended to an independent set.

Clique: a residual commitment changes what follows. On the same path, selecting v leaves both u and w compatible with the current clique. The state $(-1, 1, -1)$ is already closed. Selecting u as an additional residual commitment then excludes w, because $\{u, w\}$ is not an edge:

$$C_I\big((-1, 1, -1) \vee \{u \mapsto 1\}\big) = (1, 1, 0).$$

Here the residual supplies a choice that the initial primary commitment did not imply. Closure derives the exclusion that follows from that additional choice.

Vertex cover: prediction and eligibility use different states. For the path $u - v - w$, committing $u = 0$ forces $v = 1$. Residual prediction uses the raw state $(0, -1, -1)$, and eligibility uses the closed copy $(0, 1, -1)$. A residual exclusion $w = 0$ gives $(0, 1, 0)$, whose cover is $\{v\}$.

TSP: saturation of degree and connectivity consequences. Consider four cities and order the six undirected decision units as $(12, 13, 14, 23, 24, 34)$. The partial path $1 - 2 - 3$ is $s = (1, -1, -1, 1, -1, -1)$. City 2 already has degree two, forcing $24 = 0$. Edge 13 would close a subtour containing three cities, forcing $13 = 0$. City 1 then requires $14 = 1$, and city 3 requires $34 = 1$. Consequently, $C_I(s) = (1, 0, 1, 1, 0, 1)$, which represents the tour $1-2-3-4-1$. The two prescribed path edges induce four further assignments, including both inclusions and exclusions.

B Theoretical Results

B.1 Proof That Score Order Does Not Determine Threshold Selection

Proof of Proposition 1. A strictly increasing h preserves every comparison and tie among score entries. It therefore preserves the weak order of ℓ, and any policy that depends only on this order satisfies $\pi_{\mathrm{comp}}(\ell) = \pi_{\mathrm{comp}}(h \circ \ell)$. For the thresholded policy, choose constants

$$c_- < \tau - \max_{u \in M} \ell_u, \qquad c_+ > \tau - \min_{u \in M} \ell_u,$$

and define $h_-(x) = x + c_-$ and $h_+(x) = x + c_+$. Both maps are strictly increasing and preserve the completion ordering. The first places every entry below τ, while the second places every entry above τ. Their commitment sets are therefore $\varnothing$ and M, respectively. The same construction applies when selection uses a lower cutoff or both a lower and an upper cutoff.

B.2 Closure Assumptions and Canonical Representatives

Fix an instance I and the information order from Equation (1). A propagation rule is a map $\rho : \mathcal{S}_I \to \mathcal{S}_I$. The rule is extensive when $s \preceq \rho(s)$, monotone when $s \preceq t$ implies $\rho(s) \preceq \rho(t)$, and sound when every assignment that it adds holds in every $x \in \Omega_I(s)$. Rules are applied only to valid states. Soundness and extensivity give

$$\Omega_I(\rho(s)) = \Omega_I(s). \tag{9}$$

An extension cannot introduce a feasible completion, and soundness preserves every existing completion.

Saturation and closure. Let $\mathcal{R}_I = \{\rho_1, \ldots, \rho_K\}$ be a finite family of extensive, monotone, sound rules. Apply a rule whenever it changes the state, stopping when every rule leaves it unchanged. Each effective application resolves at least one unit, so at most $|\mathcal{U}(s)|$ such applications are possible. If t is any common fixed point extending s, monotonicity gives $s_k \preceq t$ at every intermediate step, since $\rho_{i_k}(s_k) \preceq \rho_{i_k}(t) = t$. The terminal state is therefore the least common fixed point above s, independent of rule order. Denote it by $C_I(s)$.

This construction gives the standard closure properties (Apt, 1999; Davey & Priestley, 2002):

$$\begin{aligned} s &\preceq C_I(s), && \text{(extensivity)} \\ s \preceq t &\implies C_I(s) \preceq C_I(t), && \text{(monotonicity)} \\ C_I(C_I(s)) &= C_I(s). && \text{(idempotence)} \end{aligned} \tag{10}$$

Extensivity follows from the rule sequence, monotonicity from leastness, and idempotence from saturation. Equation (9) also gives $\Omega_I(C_I(s)) = \Omega_I(s)$.

Closed representation. Let $\mathcal{S}_I^{\mathrm{cl}} = \{z : C_I(z) = z\}$. Define $s \sim_{C_I} t$ when $C_I(s) = C_I(t)$. The closure quotient $\mathcal{S}_I/\sim_{C_I}$ groups assignments with the same propagated information.

Proposition 3 (Canonical closed representation)**.** Let C_I be an extensive, monotone, and idempotent operator on $\mathcal{S}_I$. Every equivalence class $[s]_{C_I}$ contains exactly one closed state, $C_I(s)$, which is the least closed extension of s under $\preceq$. The map $[s]_{C_I} \mapsto C_I(s)$ is a bijection between $\mathcal{S}_I/\sim_{C_I}$ and $\mathcal{S}_I^{\mathrm{cl}}$.

Proof of Proposition 3. Idempotence makes $C_I(s)$ a closed member of $[s]_{C_I}$. Every closed z in that class satisfies $z = C_I(z) = C_I(s)$, proving uniqueness. For any closed extension t of s, monotonicity gives $C_I(s) \preceq C_I(t) = t$, proving leastness. Equal closed images define the same class, and each closed state represents its own class, establishing the bijection.

Staged commitments. Intermediate closure is compatible with accumulating the same admissible commitments. Let A be admissible at s, and let B assign only units unresolved in $C_I(s \vee A)$, with a jointly feasible final assignment. Then

$$C_I\big(C_I(s \vee A) \vee B\big) = C_I(s \vee A \vee B). \tag{11}$$

Put $q = s \vee A \vee B$ and $r = C_I(s \vee A) \vee B$. Joint feasibility supplies $x \in \Omega_I(q)$, and soundness ensures that x also extends r. Both states are valid. Extensivity gives $q \preceq r$. Monotonicity gives $C_I(s \vee A) \preceq C_I(q)$ and hence $r \preceq C_I(q)$. Applying closure to these inequalities proves the equality. The selected commitments are fixed in this identity. The neural query states remain those specified by the selection procedure.

B.3 Feasibility and Finite Propagation Chains

Proof of Proposition 2. Joint admissibility of $A_t = a_t \cup R_t$ and sound closure give

$$\Omega_I(z_{t+1}) = \Omega_I\big(C_I(z_t \vee A_t)\big) = \Omega_I(z_t \vee A_t) \neq \varnothing.$$

Idempotence makes z_{t+1} closed. These properties hold at every round by induction from z_0. A nonempty edit resolves at least one unit, and extensivity retains all earlier assignments, so $|\mathcal{U}(z_{t+1})| \leq |\mathcal{U}(z_t)| - 1$. Summing gives $T \leq |\mathcal{U}(z_0)|$.

A propagation chain retains its commitments. Reconstruction starts a new chain, while rejection leaves the retained state unchanged.

For MIS, the eligibility test maintains an independent set of selected vertices, which has a feasible completion obtained by excluding every remaining vertex. MCL has the same property with a clique of selected vertices. For MVC, excluded vertices form an independent set, and assigning every remaining vertex to the cover gives a feasible completion. These constructions establish admissibility for the corresponding selectors when eligibility is updated after each commitment.

B.4 Feasibility of Reconstructed Tours

TSP and ATSP reconstruction retains a set P of edges or arcs from a feasible source tour. It creates a partial assignment s^P with value 1 on P and -1 on all other decision units, then computes $z = C_I(s^P)$. Stored self-loops have value 0 and are outside U_I. The source tour belongs to $\Omega_I(s^P)$, so s^P and z are valid. All exclusions in z follow from the retained connections through closure.

TSP edge additions. On the complete undirected graph, the retained edges form vertex-disjoint paths, with isolated vertices treated as paths of length zero, or already form the full tour. An addition that respects the degree constraints and does not create a premature subtour joins two paths or completes the full tour. The TSP selector tests both edges of a residual bundle with updated degrees and components. An accepted bundle therefore leaves disjoint paths or a Hamiltonian cycle. The paths can be arranged in any cyclic order and joined at their available endpoints because the instance graph is complete. These joins respect the exclusions induced by saturated rules for degrees and premature subtours. The resulting cycle extends the edited state, and sound closure preserves its existence.

ATSP inclusion. The retained arcs form directed paths. Contract each path to a component whose remaining incoming and outgoing connections are at its first and last vertices. Before a residual exclusion, the available connections between distinct components form a complete directed graph. An inclusion passing the predecessor, successor, and subtour checks joins two directed paths or completes the tour. Connecting the remaining components in a cyclic order gives a Hamiltonian completion.

ATSP exclusion. The selector applies at most one signed residual assignment to a reconstructed state. If two directed path components remain, closure forces the two arcs joining them. A single component is likewise completed by closure. An unresolved exclusion therefore acts on at least three components. Removing one arc from a complete directed graph on $k \geq 3$ components preserves a Hamiltonian cycle: for a removed arc $A \to B$, order the components as $B, A, K_3, \ldots, K_k$ and close the cycle. Expanding each component back to its retained path gives a tour respecting the exclusion. Thus the single exclusion used after reconstruction is admissible, and subsequent sound closure preserves a feasible completion.

ATSP completion by assignment and cycle merging. The completion procedure in Appendix C.2 first computes a cycle cover while retaining the fixed paths. If the cover has more than one cycle, each cycle contains at least two contracted components because closure excludes arcs that would close a single component into a proper subtour. With one additional exclusion, three remaining components are completed by closure. For $k \geq 4$, every connection between distinct components is available except the single excluded arc.

Consider two cycles containing p and q components. Each has one unfixed connecting arc per component. Choosing one such arc from each cycle gives $pq \geq 4$ possible exchanges of their targets. Each exchange merges the cycles while retaining all fixed path arcs. A single forbidden connection rules out at most one exchange: its source fixes the removed arc in one cycle, and its target fixes the removed arc in the other. A legal exchange therefore exists. After merging, the same argument applies to the remaining cycles. Each exchange reduces the number of cycles by one, so the procedure terminates at a Hamiltonian tour. The inclusion case has all connections between the remaining components available and follows the same argument.

CVRP route additions. For route states with exclusions generated by the rules in Appendix C.3, each customer component is a path with total demand at most Q and at most two depot connections. The residual selector preserves these properties. A path with multiple customers and no depot connection can be attached at both endpoints. With one depot connection, its other endpoint remains available. With two, the route is complete. A singleton uses one stored depot marker. These attachments produce routes that satisfy the capacity constraint. In the representation of customer order, each singleton satisfies $q_i \leq Q$, so every completed order admits a feasible split.

B.5 Residual Progress and Expected Propagation Rounds

Let $m_t = |\mathcal{U}(z_t)|$, $A_t = a_t \cup R_t$, and $g_t = m_t - m_{t+1}$. Since $D_{A_t} \subseteq \mathcal{U}(z_t)$, applying the edits alone removes exactly $|D_{A_t}|$ unresolved units. Consequently, $|\mathcal{U}(z_t \vee A_t)| = m_t - |D_{A_t}|$, and subtracting m_{t+1} proves Equation (8). If $c_t = |\mathcal{U}(z_t \vee A_t)| - m_{t+1}$ denotes the closure contribution, then $g_t = |D_{A_t}| + c_t$ with $c_t \geq 0$. Edit sets assign one value to each domain element. Repeated identical proposals are identified before forming A_t. For staged closure, Equation (11) gives the same final closed state for the same compatible commitments.

Summing over any finite chain yields

$$\sum_{t=0}^{T-1} g_t = m_0 - m_T.$$

For $T > 0$, the average progress is $(m_0 - m_T)/T$, which equals m_0/T for a complete chain. An empty combined edit set has $g_t = 0$.

Additional progress from a residual. Closure of the primary edit alone resolves some of the assignments counted by g_t. The additional progress from the selected residual is

$$\Delta_t^{\mathrm{res}} = |\mathcal{U}(C_I(z_t \vee a_t))| - |\mathcal{U}(z_{t+1})| \geq 0. \tag{12}$$

The inequality follows from monotonicity of closure for the jointly admissible edits. This count includes the residual assignments and any further consequences they induce beyond closure of the primary edit. For edge-structured problems, $a_t = \varnothing$ and z_t is closed, so $\Delta_t^{\mathrm{res}} = g_t$. Both quantities measure resolved assignments. Solution quality is measured by f_I.

Expected round bound. Fix an initial state with $m = m_0 < \infty$. Let $\mathcal{F}_t$ contain the history before round t, and let T be the stopping time counting active rounds in a propagation chain. Extend the state constantly after stopping. Suppose a constant $\bar{g} > 0$ satisfies

$$\mathbb{E}[g_t \mid \mathcal{F}_t] \geq \bar{g}\, \mathbf{1}_{\{T>t\}} \quad \text{for every } t \geq 0. \tag{13}$$

For every $N \geq 1$, the tower property and the progress identity give

$$\bar{g} \sum_{t=0}^{N-1} \mathbb{P}(T > t) \leq \sum_{t=0}^{N-1} \mathbb{E}[g_t] = m - \mathbb{E}[m_{T\wedge N}] \leq m.$$

The unresolved count is bounded by m, so these expectations are finite. The tail-sum identity gives $\sum_{t<N} \mathbb{P}(T > t) = \mathbb{E}[T \wedge N]$. Monotone convergence then yields

$$\mathbb{E}[T] \leq \frac{m}{\bar{g}}. \tag{14}$$

The bound also gives $T < \infty$ almost surely and allows stopping with unresolved units remaining. For $m = 0$, $T = 0$. The quantity T counts rounds in one propagation chain. Appendix D.4 specifies outer reconstruction cycles and their model queries.

C Realizations for Individual Problems

For each problem, the representation in Equation (2) specifies the decision units and feasible set, sound closure C_I, admissible family $\mathcal{A}_I$, and completion operator D_I. The residual readout always selects through R_τ, while the encoding determines whether a candidate is a vertex assignment, a signed arc, a connection, or a bundle of edges. The feasibility arguments in Appendices B.3 and B.4 connect these selectors to admissible extensions of their partial states.

C.1 Traveling Salesman Problem

Encoding and objective. For $n \geq 3$ cities with symmetric distances d_{ij}, the units are unordered pairs $\{i, j\}$ with $i < j$. A feasible complete assignment is the incidence vector of the edges in a Hamiltonian cycle, with objective $f_I(x) = \sum_{i<j} d_{ij} x_{ij}$. The stored matrix is symmetric, has a zero diagonal, and uses -1 for unresolved connections. Let $d_i^+(s) = |\{j : s_{ij} = 1\}|$ be the selected degree, and let $P(s)$ be the components formed by selected edges.

Closure and edge additions. Closure repeatedly applies three rules. When $d_i^+(s) = 2$, all other unresolved edges incident to i are excluded. An unresolved edge joining the endpoints of a selected path is excluded if it would form a cycle before all n cities are included. If a city needs $2 - d_i^+(s) > 0$ additional edges and exactly that many eligible connections remain, those connections are selected, with degrees and components updated after each addition. Saturation stops when no assignment changes. An inclusion requires available degree at both endpoints and preserves the absence of premature subtours.

Residual commitments. TSP search starts from a reconstructed closed state, so $a_t = \varnothing$. For two nonadjacent edges (a, b) and (c, d) in the source tour, a residual candidate is the bundle $\{(a, c), (b, d)\}$. Both proposed connections must be unresolved. Its score is

$$r(a, b, c, d) = \ell^{\mathrm{res}}_{ac} + \ell^{\mathrm{res}}_{ca} + \ell^{\mathrm{res}}_{bd} + \ell^{\mathrm{res}}_{db}.$$

The selector shortlists bundles with the highest scores above τ and commits the first one passing the joint degree and subtour checks. A bundle passes when either insertion order is compatible, with degrees and components updated after the first edge before testing the second. Closure follows the selected bundle.

Completion. Completion joins selected paths into a Hamiltonian cycle using edit scores while respecting degree constraints and preventing premature subtours. The completed tour is then improved by 2-opt. Appendix D.4 gives the reconstruction and perturbation procedures.

C.2 Asymmetric Traveling Salesman Problem

Encoding and objective. For $n \geq 2$, units are ordered arcs (i, j) with $i \neq j$, with a separate value for each direction. A feasible solution has one predecessor and one successor per city and forms a single directed Hamiltonian cycle, with objective $f_I(x) = \sum_{i \neq j} d_{ij} x_{ij}$. Let $d^{\mathrm{out}}_i(s)$ and $d^{\mathrm{in}}_i(s)$ count committed outgoing and incoming arcs.

Closure. A selected outgoing arc excludes the other unresolved arcs in its row, and a selected incoming arc excludes the other unresolved arcs in its column. An arc closing a directed path into a proper subtour is excluded. If a city without a successor has exactly one eligible outgoing arc, that arc is selected. The corresponding rule holds for a missing predecessor. These rules repeat until no value changes.

Signed residual selection. With $a_t = \varnothing$, the selector examines unresolved arcs in the reconstructed closed state. A positive residual score proposes inclusion and a negative score proposes exclusion. It selects at most one assignment with score magnitude above τ, ranked by ℓ^{res}_{ij} for inclusion and $-\ell^{\mathrm{res}}_{ij}$ for exclusion. These are the raw residual scores. Appendix D.3 relates their thresholds to the scores used in supervision. Inclusion requires free predecessor and successor capacity and no premature subtour. Exclusion requires that the affected row and column each retain a selected or unresolved connection. Closure is applied after the assignment. Appendix B.4 establishes admissibility for a single signed assignment after reconstruction.

Completion. Completion first saturates the state. It masks excluded arcs, self-loops, and connections inconsistent with selected predecessors or successors, then solves a linear assignment using the Hungarian algorithm and the completion scores. The assignment gives each vertex one predecessor and one successor and forms a directed cycle cover.

To merge two cycles, the procedure replaces $a \to a'$ and $b \to b'$ by $a \to b'$ and $b \to a'$. Both removed arcs must be unfixed, and both new arcs must be permitted by the partial assignment. It selects the legal exchange with the largest score change, including a negative change, then updates successors and identifies the resulting cycles. These exchanges continue until one tour remains. Closure is computed before the assignment stage. Each exchange updates the cycle cover directly. Appendix B.4 proves termination for the states produced by reconstruction and residual selection. The complete tour is then improved by 2-opt using directed costs.

C.3 Capacitated Vehicle Routing Problem

Route representation. Let 0 be the depot and $1, \ldots, n$ the customers, with demands $0 < q_i \leq Q$ and vehicle capacity Q. A solution is a collection of routes that start and end at the depot, visit every customer exactly once, and satisfy the capacity bound. The objective sums all traversed distances. The route representation stores undirected connections between customers and binary markers for connections to the depot. A singleton route $0 - i - 0$ stores its depot marker once, while the route objective counts both traversals.

For each selected customer component K, maintain its demand $q(K) = \sum_{i \in K} q_i$ and the number $b(K)$ of selected depot connections. Customer degrees count stored selected connections. Connections between customers determine the customer components. The depot does not merge different routes into one component.

Closure and residual additions. An unresolved connection between customers is excluded if an endpoint has selected degree two, its endpoints belong to the same customer component, merging the components exceeds capacity, or the components together have more than two depot connections. An unresolved connection to the depot is excluded when the customer has degree two or its component already has two depot connections. Closure saturates these exclusion rules.

A residual inclusion passes the same checks. The selector chooses the eligible connection with the highest score above τ, commits it, and closes the state. It repeats with the same residual scores and the updated eligible set until no score passes the threshold or the commitment limit is reached. All these commitments use one residual query.

Route completion. Unresolved connections between customers are scanned in descending order of their edit scores and included while the degree, component, load, and depot conditions hold. Each remaining customer component is then connected to the depot at eligible endpoints. A component with multiple customers needs two depot connections, while a singleton needs one stored marker. The depot markers divide the resulting sequence into routes and thereby determine feasibility and cost.

Representation of customer order. A second representation stores partial adjacency among customers. Its completion uses the TSP procedure on the customer submatrix, followed by 2-opt on the tour over customers. Depot placement and route membership are determined by splitting this order. For a linear order $\pi_1, \ldots, \pi_n$, define

$$c(i, j) = d_{0,\pi_i} + \sum_{k=i}^{j-1} d_{\pi_k,\pi_{k+1}} + d_{\pi_j,0}.$$

The minimum cost among splits that satisfy the capacity constraint is given by

$$V(0) = 0, \qquad V(j) = \min_{\substack{0 \le i < j \\ \sum_{k=i+1}^{j} q_{\pi_k} \le Q}} \big\{ V(i) + c(i+1, j) \big\}. \tag{15}$$

Predecessors recover the route boundaries. The implementation evaluates both directions and every cyclic start of the completed customer tour, then retains the split with the lowest cost.

C.4 Maximum Independent Set

Encoding and closure. For an undirected graph $G = (V, E)$, units are vertices and $x_v = 1$ selects v. The feasible set and minimization objective are

$$\mathcal{X}_I = \{x \in \{0,1\}^V : x_u + x_v \le 1 \text{ for all } \{u, v\} \in E\}, \qquad f_I(x) = -\sum_{v \in V} x_v.$$

Every selected vertex forces its unresolved neighbors to 0. Applying this rule to all selected vertices closes the state. Primary selection includes the eligible unresolved vertex with the largest edit score. Residual selection applies the same eligibility test and adds at most one vertex whose residual score exceeds τ.

Completion. Completion returns an independent set that preserves the input commitments. Edit scores and a preference for low degree in the unresolved subgraph guide the selection of compatible vertices.

C.5 Maximum Clique

Encoding and closure. Here $x_v = 1$ includes v in a clique:

$$\mathcal{X}_I = \{x \in \{0,1\}^V : x_u + x_v \le 1 \text{ for distinct } u, v \text{ with } \{u, v\} \notin E\},$$
$$f_I(x) = -\sum_{v \in V} x_v.$$

Selecting u excludes every unresolved vertex not adjacent to u. An eligible addition must be adjacent to all selected vertices. Primary selection takes the largest edit score on this set, and residual selection takes at most one additional eligible vertex above τ.

Completion. Completion returns a clique containing the selected vertices and respecting committed exclusions. Edit scores and degree information guide the selection of mutually compatible vertices.

C.6 Minimum Vertex Cover

Encoding and closure. The value $x_v = 1$ includes v in the cover:

$$\mathcal{X}_I = \{x \in \{0,1\}^V : x_u + x_v \ge 1 \text{ for all } \{u, v\} \in E\}, \qquad f_I(x) = \sum_{v \in V} x_v.$$

An exclusion $x_u = 0$ forces each unresolved neighbor to 1. The excluded vertices consequently form an independent set. Primary selection chooses an eligible unresolved vertex with the lowest edit score and sets it to 0. Residual selection applies the same rule and accepts an additional exclusion only when its residual score is below $-\tau$. After the primary edit, its raw intermediate state is used for residual prediction and a closed copy is used for eligibility.

Completion. Completion constructs an independent set extending the committed exclusions, guided by edit scores and degree information. Its complement is a feasible vertex cover that preserves the input commitments.

D Learning and Search Details

D.1 Scorer Architecture

Instance and assignment representations. Edge-structured problems use an instance encoder and a decoder conditioned on the partial assignment. TSP supplies city coordinates, CVRP adds demands normalized by capacity, and ATSP supplies mean incoming and outgoing costs. Coordinates are centered and normalized by the mean distance from their center. Queries share the encoded instance features. The decoder incorporates the current commitments.

For edge-structured problems, the assignment enters attention through the bias

$$B_{ij}^{\text{state}} = \begin{cases} 1, & s_{ij} = 1, \\ -1, & s_{ij} = 0, \\ 0, & s_{ij} = -1. \end{cases} \tag{16}$$

ATSP and CVRP also use a bias B^{cost} given by negative costs normalized by the mean instance cost. TSP vertex features include selected degree, remaining degree availability, and the size of the selected component.

Vertex-assignment problems use a decoder-only Transformer with graph adjacency as an attention bias. Vertex features encode degree, neighborhood statistics, and the supplied vertex index. MCL also conditions on the selected clique and its eligible extensions. Distinct embeddings represent committed zero, committed one, and unresolved entries. The sampled retention fraction or normalized commitment count supplies progress information, according to the representation.

Prediction roles. Pre-normalized self-attention and SwiGLU blocks produce shared features for the edit and residual decision readouts. Readouts for edge-structured problems score pairs of vertices. ATSP uses distinct source and destination projections to represent direction. Readouts for vertex-assignment problems score individual vertices. The same features support the auxiliary assignment readout. When state guidance is enabled, mean pooling produces q_θ, with $\hat{q}_\theta = \sigma(q_\theta)$ used in the state loss and the optional acceptance rule in Appendix D.4.

TSP edit scores and the outputs for vertex-assignment problems use bounded transformations. ATSP edit scores combine a scaled neural prediction with the cost and assignment biases:

$$\ell^{\text{edit}} = \alpha_{\text{ATSP}} \ell^{\text{edit,raw}} + B^{\text{cost}} + B^{\text{state}}, \tag{17}$$

where $\alpha_{\text{ATSP}} > 0$ sets the neural score scale. The ATSP residual selector uses its readout directly. CVRP edit scores use dot products scaled by the feature dimension. Self-loops and padded entries are excluded from the decisions. Models are trained separately for each problem class. The two CVRP representations share parameters.

D.2 Training States and Supervision Targets

Partial contexts. A pair $(I, x^\star)$ yields multiple training contexts by retaining some reference decisions and leaving the rest unresolved. Sampling varies the retained fraction and the spatial pattern of missing decisions, and includes empty contexts. For these reference subsets, $s \preceq x^\star$, and sound propagation preserves $x^\star \in \Omega_I(s)$. Assignment labels remain those of $x^\star$.

TSP and ATSP remove random connections, contiguous segments, or costly connections from reference tours. ATSP also uses contexts from perturbed reference tours, supervised by the original reference. These examples train reference membership prediction from a perturbed context. A retained arc absent from the reference gives $x^\star \notin \Omega_I(s)$. The original reference is then a prediction target, while the inference completion must preserve the current commitments. ATSP training preprocessing applies a bounded number of degree and subtour propagation passes. Inference computes the saturated closure.

MIS and MCL contexts retain vertices selected in the reference. MVC retains exclusions from the independent set complementary to the reference cover. Vertices are sampled randomly or according to neighborhoods or degrees. Inputs also include contexts before propagation, as used for residual prediction.

CVRP uses contexts in both its route representation and its representation of customer order. Route contexts remove individual connections, segments, or complete routes, then exclude connections ruled out by degree, capacity, and depot constraints. Contexts for customer order remove connections from the customer adjacency reference. Both representations train the scorer.

Auxiliary assignment supervision. Auxiliary labels identify consequences of the available implication tests. TSP labels exclusions implied by degree and subtour constraints, together with forced inclusions. ATSP uses the corresponding predecessor and successor rules. MIS labels neighbors of selected vertices as excluded, MCL labels incompatible vertices as excluded, and MVC labels neighbors of excluded vertices as included.

CVRP labels impossible connections and sole eligible connections required to satisfy customer degree. Its inference closure uses the exclusion rules in Appendix C.3. Auxiliary prediction trains the representation, while C_I performs deterministic propagation during inference.

State targets. The state target measures missing reference structure and the specified disagreement with that reference. For a reference binary assignment $x^\star$, define

$$\mathcal{D}_\star(s) = \{u : s(u) = 1,\ x^\star(u) = 0\} \cup \{u : x^\star(u) = 1,\ s(u) \neq 1\}. \tag{18}$$

For edge-structured problems, the state target is

$$y^q = \text{clip}\left(\frac{\sum_{u \in \mathcal{D}_\star(s)} w_u}{\max\{f_I(x^\star), 1\}},\ 0,\ 1 \right), \tag{19}$$

Table 8: Supervision terms for primary edits and residual commitments.

Problem	Edit supervision	Residual supervision
TSP	$\mathcal{R}_{\text{row}}$	$\mathcal{R}_{\text{global}},\ \mathcal{B}$
ATSP	$\mathcal{R}_{\text{rc}},\ \mathcal{B}$	$\mathcal{R}_{\text{rc}},\ \mathcal{B}$
CVRP route	$\mathcal{R}_{\text{rc}}$	$\mathcal{R}_{\text{global}},\ \mathcal{B}$
CVRP order	$\mathcal{R}_{\text{rc}}$	None
MIS/MCL	$\mathcal{R}_{\text{global}}$	$\mathcal{R}_{\text{global}},\ \mathcal{B}$
MVC	$\mathcal{R}_{\text{global}},\ \mathcal{B}$	$\mathcal{R}_{\text{global}},\ \mathcal{B}$

where w_u is the edge or arc cost and undirected units are counted once. MIS and MCL use unit weights and denominator $\max\{\sum_v x^\star_v, 1\}$. MVC applies the same count to the encoding of the complementary independent set, replacing known s_v by $1 - s_v$ and the reference by $1 - x^\star$.

The target for the CVRP customer order counts connections between customers: selected edges absent from the order reference and edges in the route reference that remain unresolved or excluded. Their sum, weighted by distance, is normalized by the reference route cost as in Equation (19).

D.3 Supervision Objectives

Ranking and assignment prediction. Ranking supervises priority among eligible commitments. Let M contain the supervised candidates, $P \subseteq M$ the entries labeled positive by the reference, and $\mathcal{G}$ the normalization groups. With a positive numerical stabilizer δ, define

$$\mathcal{R}_{\mathcal{G}}(\ell) = -\frac{1}{|P| + \delta} \sum_{G \in \mathcal{G}} \sum_{u \in P \cap G} \log \frac{\exp(\ell_u)}{\sum_{v \in M \cap G} \exp(\ell_v)}. \tag{20}$$

Groups without positives contribute zero. Normalization over rows, columns, and entire instances gives $\mathcal{R}_{\text{row}}$, $\mathcal{R}_{\text{col}}$, and $\mathcal{R}_{\text{global}}$. Positive counts are accumulated over the training batch.

Classification supervises individual assignment values, including the residual predictions used for thresholded selection:

$$\mathcal{B}_K(\ell, y; M) = -\frac{\sum_{u \in M}[w_+ y_u \log \sigma(\ell_u) + (1 - y_u) \log(1 - \sigma(\ell_u))]}{|M| + \delta}. \tag{21}$$

Here N_+ and N_- count positive and negative labels in M. The weight for positive examples is defined by

$$w_+ = \min\{K, N_-/(N_+ + \delta)\} \text{ when } N_- > 0; \quad w_+ = 1 \text{ when } N_- = 0. \tag{22}$$

Thus a mask containing only positive labels, as used for MVC closure labels, still contributes to supervision. Write $\mathcal{B} = \mathcal{B}_{K_{\text{cls}}}$ for edit/residual classification.

Supervision for each problem. Let

$$\mathcal{R}_{\text{rc}} = \frac{\mathcal{R}_{\text{row}} + \mathcal{R}_{\text{col}}}{2}.$$

Table 8 identifies the terms whose weighted combinations define $\mathcal{L}_{\text{edit}}$ and $\mathcal{L}_{\text{res}}$. Candidate masks for edge-structured problems enforce degree and connectivity conditions, with additional load and depot conditions for CVRP. TSP/CVRP residual ranking uses eligible undirected edges as one group per instance. MIS/MCL supervise unresolved vertices compatible with the selected set. MVC uses negated logits and labels from the complementary independent set on eligible exclusions.

The loss for the CVRP customer order uses the TSP eligibility rules on the customer submatrix. ATSP training uses the transformed edit scores from Appendix D.1 and applies the same transformation to residual scores:

$$\widetilde{\ell}^{\text{res}}_{ij} = \alpha_{\text{ATSP}} \ell^{\text{res}}_{ij} + B^{\text{cost}}_{ij} + B^{\text{state}}_{ij}. \tag{23}$$

The ATSP residual losses in Table 8 act on $\widetilde{\ell}^{\text{res}}$, while the inference selector uses ℓ^{res}. Writing $b_{ij} = B^{\text{cost}}_{ij} + B^{\text{state}}_{ij}$, the corresponding inclusion and exclusion conditions are

$$\ell^{\text{res}}_{ij} > \tau \iff \widetilde{\ell}^{\text{res}}_{ij} > b_{ij} + \alpha_{\text{ATSP}}\tau,$$
$$\ell^{\text{res}}_{ij} < -\tau \iff \widetilde{\ell}^{\text{res}}_{ij} < b_{ij} - \alpha_{\text{ATSP}}\tau.$$

Thus τ controls selection in the raw score scale, while its value in the supervised score scale depends on the arc.

Auxiliary and optional state objectives. For entries F labeled by the implication tests and unresolved neutral entries N outside F, define

$$\mathcal{L}_{\text{close}} = \mathcal{B}_{K_{\text{close}}}(\ell^{\text{close}}, y^{\text{close}}; F) + \eta_{\text{neutral}} \frac{\sum_{u \in N} (\sigma(\ell_u^{\text{close}}) - 1/2)^2}{|N| + \delta}. \tag{24}$$

The caps $K_{\text{cls}}, K_{\text{close}} > 0$ limit the weights assigned to positive examples, and $\eta_{\text{neutral}} \geq 0$ weights the neutral term. The first term learns the auxiliary assignments. The second keeps predictions on neutral entries near $1/2$. TSP/CVRP count each undirected unit once, and ATSP uses directed entries. The state loss averages $(\sigma(q_\theta) - y^q)^2$ over instances.

These objectives combine as in Equation (6). CVRP jointly trains its two representations:

$$\mathcal{L}_{\text{CVRP}} = \lambda_{\text{route}} \mathcal{L}_{\text{route}} + \lambda_{\text{order}} \mathcal{L}_{\text{order}}, \qquad \mathcal{L}_{\text{order}} = \mathcal{L}_{\text{edit}}^{\text{order}} + \lambda_{\text{state}} \mathcal{L}_{\text{state}}^{\text{order}}. \tag{25}$$

The nonnegative weights λ_{route} and λ_{order} balance the two representations. The route objective includes the decision, auxiliary, and state terms specified above. Both objectives train the model.

D.4 Search Procedures

Initialization and acceptance. Each of the S trajectories starts by scoring and completing an empty assignment. TSP, ATSP, and vertex-assignment problems close it before the query. CVRP begins from an empty route state. A trajectory retains a feasible solution and its associated state score, together with a partial state when further propagation uses it.

Algorithm 2: Test-time search with neural residual propagation

Input: I, scorer θ, operators defined for I, trajectories S, cycles C, threshold τ, weight β, tolerance ε

Initialize each trajectory's feasible pair (x_j, v_j).
$\varphi_j \leftarrow f_I(x_j) + \beta v_j$
for $c = 1, \ldots, C$ **do**
  for each active trajectory j in parallel **do**
    for each successive proposal (x', v') **do**
      $\varphi' \leftarrow f_I(x') + \beta v'$
      if $\varphi' < \varphi_j - \varepsilon$ **then**
        $(x_j, v_j, \varphi_j) \leftarrow (x', v', \varphi')$
        Retain its propagation state when used.
return x_j minimizing $f_I(x_j)$

Algorithm 2 processes proposals in their prescribed order. Each is compared against the current incumbent, including updates accepted earlier in the cycle. A candidate (x', v') contains a complete solution and the state score $v' = \hat{q}_\theta(I, s)$ attached to its partial context. With $\Phi_\beta(x, v) = f_I(x) + \beta v$, acceptance requires

$$f_I(x') + \beta v' < f_I(x_j) + \beta v_j - \varepsilon. \tag{26}$$

The default $\beta = 0$ gives Equation (7). A positive β adds a preference for states with a smaller predicted reference discrepancy, whose target is defined in Appendix D.2. Each accepted proposal replaces the incumbent and its attached score. The returned solution minimizes f_I among the final incumbents. Perturbed score rankings diversify completions across trajectories. For vertex-assignment problems, residual perturbations change score magnitudes while preserving the assignment indicated by each sign.

State scores and candidates. Each candidate carries the state score from a specified query. For vertex-assignment problems, candidates containing a residual commitment use the query after closure. MVC also completes a candidate with the primary edit alone from a closed copy. This candidate carries the score from the raw query after the primary edit. For edge-structured problems, reconstructed candidates use the query that produced their completion scores, and 2-opt retains that score. Perturbations of a complete solution retain the incumbent's state score.

Vertex-assignment problems. Each cycle selects a primary edit from the scores attached to the retained closed state and queries residual predictions on $s_t = z_t \vee a_t$. MVC closes a separate copy for residual eligibility. The selected residual is followed by exact closure and a further query for completion and state scoring. MVC compares completions both with and without the residual, using their respective state scores.

MIS continues its propagation chains until no primary edit remains. MCL and MVC also reopen incumbent solutions on a subset of trajectories. Reopening retains part of the incumbent clique or cover exclusions, respectively, then closes and scores the resulting partial state before selecting a primary edit. This permits earlier decisions to be reconsidered while each individual propagation chain retains its commitments.

Edge-structured problems. TSP and ATSP first perturb the incumbent tour and apply 2-opt. Reconstruction selects retained connections by removing contiguous, random, or multiple separated parts of that source tour, as well as parts containing connections with high cost. It creates a fresh state with entries outside the diagonal set to -1 and self-loops set to 0, assigns retained connections to 1, and computes closure. TSP writes both symmetric entries. ATSP writes the corresponding direction. The same reconstructed state and source tour produce a direct completion and a completion after residual selection, including when the direct candidate is accepted. Residual selection follows Appendices C.1 and C.2.

CVRP reconstructs a route state, commits residual connections with closure between additions, and queries the resulting state for route completion. It then reconstructs a state in the customer order representation. One query produces two diversified tours over the customers, each followed by 2-opt and a split that satisfies the capacity constraint.

Neural computation. The cycle limit C and trajectory count S control search computation. After initialization, TSP and ATSP have two model query stages per cycle. CVRP has three: residual prediction for the route, route completion, and prediction of customer order. Vertex-assignment problems query the raw state after the primary edit and the closed state after residual selection, with an additional query for reopened states. Queries are batched across the applicable trajectories. Completion and 2-opt reuse the available predictions.

E Additional Experimental Details

E.1 Datasets, Splits, and Reference Solutions

We evaluate NRP4CO on three edge-structured problems and three vertex-assignment problems. We also report maximum cut (MCut) on BA-SMALL and BA-LARGE. The benchmark names, test-set sizes, reference solvers, and mean reference objectives are summarized in Table 9.

Table 9: Dataset statistics for the NRP4CO evaluation benchmarks.

Problem	Dataset	Reference Solver	Testing Data Size	Obj.
TSP	TSP-50	Concorde	1,280	5.688
	TSP-100	Concorde	1,280	7.756
	TSP-500	Concorde	128	16.546
	TSP-1000	Concorde	128	23.118
ATSP	ATSP-50	LKH(1000)	2,500	1.5545
	ATSP-100	LKH(1000)	2,500	1.5660
	ATSP-200	LKH(1000)	100	1.5647
	ATSP-500	LKH(1000)	100	1.5734
CVRP	CVRP-50	HGS	10,000	10.366
	CVRP-100	HGS	10,000	15.563
	CVRP-200	HGS	100	19.630
	CVRP-500	HGS	100	37.154
MIS	RB-[200-300]	KaMIS	500	20.090
	RB-[800-1200]	KaMIS	500	43.004
	ER-[700-800]	KaMIS	128	44.969
MCL	RB-SMALL	Gurobi	500	19.082
	RB-LARGE	Gurobi	500	40.182
MVC	RB-SMALL	Gurobi	500	205.764
	RB-LARGE	Gurobi	500	968.228
MCut	BA-SMALL	Gurobi	500	727.844
	BA-LARGE	Gurobi	500	2936.886

The edge-structured benchmarks use uniform Euclidean instances for TSP and CVRP and full directed cost matrices for ATSP. MIS uses the RB and ER graph families, while MCL and MVC use the RB family; in ER-[700-800], each edge is included independently with probability 0.15. The MCut experiment uses the BA-SMALL and BA-LARGE graph families. The training data sizes for each model are given in Table 10, and the test split is listed in Table 9.

Reference solutions are obtained using Concorde for TSP, LKH with 1,000 trials for ATSP, HGS for CVRP, KaMIS for MIS, and Gurobi for MCL, MVC, and MCut. Gurobi supplies the reference objectives under the time limits used in the main evaluation. The Obj. column reports the mean reference objective over the corresponding test set.

E.2 Training Data and Model Configurations

Training pairs consist of an instance and a reference solution, as described in Appendix D.2. For each pair, we sample retained reference decisions and unresolved entries to form multiple partial contexts, including empty contexts and states before propagation. ATSP additionally uses perturbed reference-tour contexts, and CVRP trains both its route and customer-order representations. The resulting training data sizes and model settings are listed in Table 10.

Table 10: Hyperparameters and datasets for training NRP4CO across the evaluated combinatorial optimization problems.

Problem	**Model**	**Pretrain**	**Training Parameters**					**Model Parameters**		
			Data Size	Epoch	Batch Size	LR	Warmup	Layers	Heads	Hidden Dim.
TSP	Uniform-50	–	1,280,000	600	1024	1e-3	100	16, 6	8	256
	Uniform-100	–	1,280,000	600	1024	1e-3	100	16, 6	8	256
	Uniform-500	Uniform-100	128,000	600	64	1e-3	100	16, 6	8	256
	Uniform-1000	Uniform-500	128,000	600	32	1e-3	100	16, 6	8	256
ATSP	Uniform-50	–	640,000	600	1024	1e-3	100	16, 6	8	256
	Uniform-100	Uniform-50	128,000	600	1024	1e-3	100	16, 6	8	256
	Uniform-200	Uniform-100	32,000	600	512	1e-3	100	16, 6	8	256
	Uniform-500	Uniform-200	6,400	600	128	5e-4	10	16, 6	8	256
CVRP	Uniform-50	–	1,280,000	600	1024	1e-3	100	16, 6	8	512
	Uniform-100	Uniform-50	1,536,000	600	512	1e-3	100	16, 6	8	512
	Uniform-200	Uniform-100	128,000	600	256	1e-3	50	16, 6	8	512
	Uniform-500	Uniform-200	12,800	600	64	1e-3	50	16, 6	8	512
MIS	RB-[200-300]	–	90,000	300	512	1e-4	20	0, 12	4	256
	RB-[800-1200]	RB-[200-300]	6,400	300	64	2e-4	100	0, 12	4	256
	ER-[700-800]	–	163,840	300	64	2e-4	100	0, 12	4	256
MCL	RB-SMALL	–	64,000	600	512	2e-4	20	0, 12	4	256
	RB-LARGE	RB-SMALL	6,400	600	64	1e-4	20	0, 12	4	256
MVC	RB-SMALL	–	128,000	600	512	1e-3	50	0, 12	4	256
	RB-LARGE	RB-SMALL	6,400	300	64	5e-4	50	0, 12	4	256
MCut	BA-SMALL	–	128,000	600	512	1e-3	50	0, 12	4	256
	BA-LARGE	BA-SMALL	128,000	600	64	2e-4	100	0, 12	4	256

NRP4CO uses a separate model for each problem and scale. Pretrain identifies the checkpoint used to initialize a row, while a dash denotes no pretraining. Training Parameters report data size, epoch count, batch size, learning rate, and warmup. The architecture is shared within the two problem families; Layers gives encoder and decoder depths for edge-structured models, and decoder depth for vertex-assignment models. Heads and Hidden Dim. give the attention-head count and hidden dimension.

F Additional Results and Analyses

F.1 Summary Comparison with Neural Baselines

Table 11 compares NRP4CO with the neural method that has the best gap on each included dataset for TSP, ATSP, CVRP, MIS, MCL, and MVC. Its per-dataset Gap Reduction and Speedup values un-

derlie the problem-level averages in Table 4. The last two columns use the definitions in Section 5.2. Across the 19 rows, the mean gap decreases from 1.4107% to 0.954%, corresponding to a 32.37% Gap Reduction. Mean runtime decreases from 2.9072s to 1.172s, which gives a $2.48\times$ speedup.

Table 11: Summary comparison against the neural baseline with the best gap for TSP, ATSP, CVRP, MIS, MCL, and MVC.

Problem	Dataset	Baseline			NRP4CO			Gap Reduction↑	Speedup↑
		Method	Gap↓	Time↓	Setting	Gap↓	Time↓		
TSP	TSP-50	StruDiCO	0.01%	0.06s	S=1,C=16	0.002%	0.015s	75.84%	3.93×
TSP	TSP-100	LEHD PRC 100	0.01%	0.64s	S=1,C=16	0.011%	0.025s	-12.34%	25.96×
TSP	TSP-500	NEXCO (D_s=7)	0.22%	3.26s	S=1,C=16	0.237%	0.285s	-9.09%	11.43×
TSP	TSP-1000	StruDiCO	0.54%	2.61s	S=1,C=16	0.315%	0.888s	41.59%	2.94×
ATSP	ATSP-50	COExpander	0.17%	0.20s	S=64,C=1	0.091%	0.050s	46.29%	3.97×
ATSP	ATSP-100	COExpander	0.95%	0.71s	S=64,C=1	0.126%	0.147s	86.71%	4.83×
ATSP	ATSP-200	COExpander	1.50%	2.64s	S=64,C=1	0.293%	0.644s	80.46%	4.10×
ATSP	ATSP-500	COExpander	1.57%	16.48s	S=64,C=1	0.553%	3.341s	64.78%	4.93×
CVRP	CVRP-50	NEXCO (D_s=5)	0.76%	0.18s	S=1,C=16	0.635%	0.040s	16.99%	4.54×
CVRP	CVRP-100	NEXCO (D_s=5)	1.24%	0.18s	S=1,C=16	1.118%	0.110s	10.02%	1.63×
CVRP	CVRP-200	NEXCO (D_s=5)	2.16%	0.97s	S=1,C=16	1.512%	0.679s	30.02%	1.42×
CVRP	CVRP-500	RL4CO	4.17%	0.88s	S=1,C=16	2.949%	4.250s	29.27%	0.21×
MIS	RB-[200-300]	NEXCO (D_s=7)	1.66%	0.14s	S=64,C=4	0.895%	0.298s	46.07%	0.47×
MIS	RB-[800-1200]	COExpander (S=4)	3.58%	3.82s	S=64,C=4	4.398%	3.169s	-22.86%	1.21×
MIS	ER-[700-800]	NEXCO (D_s=7)	4.20%	0.56s	S=64,C=4	3.746%	2.224s	10.80%	0.25×
MCL	RB-SMALL	COExpander (S=4)	0.45%	1.74s	S=32,C=1	0.044%	0.190s	90.16%	9.17×
MCL	RB-LARGE	COExpander (S=4)	3.08%	18.67s	S=32,C=1	0.993%	3.333s	67.78%	5.60×
MVC	RB-SMALL	COExpander (S=4)	0.37%	0.15s	S=64,C=1	0.058%	0.175s	84.27%	0.86×
MVC	RB-LARGE	COExpander (S=4)	0.16%	1.35s	S=64,C=1	0.149%	2.410s	6.70%	0.56×
Mean	–	–	1.4107%	2.9072s	–	0.954%	1.172s	32.37%	2.48×

F.2 Additional Component Ablations

The additional ablations in Table 12 separate the roles of the two decision outputs, the supervision signals, and the inference-time state updates. Removing residual selection increases the gap on all three problems, from 0.315% to 0.377% on TSP-1000, from 2.949% to 3.046% on CVRP-500, and from 0.993% to 1.074% on MCL RB-LARGE. The degradation is largest in relative terms on TSP-1000, while the same direction across edge-structured problems and the vertex-assignment problem shows that residual commitments contribute beyond the primary edit and final completion. The corresponding time decreases are small on TSP-1000 and CVRP-500, but larger on MCL RB-LARGE, where omitting residual selection reduces time from 3.333s to 1.075s. This difference separates the quality contribution of residual commitments from the computation required to score and apply them.

Table 12: Additional component ablations on TSP-1000, CVRP-500, and MCL RB-LARGE.

Method	TSP-1000		CVRP-500		MCL RB-LARGE	
	Gap↓	Time↓	Gap↓	Time↓	Gap↓	Time↓
NRP4CO	0.315%	0.888s	2.949%	4.250s	0.993%	3.333s
w/o residual selection	0.377%	0.852s	3.046%	4.160s	1.074%	1.075s
w/o constraint propagation	0.370%	1.912s	3.106%	4.178s	1.057%	1.072s
shared edit/residual head	0.353%	0.879s	3.017%	4.234s	1.026%	3.264s
w/o residual assignment supervision	0.385%	0.890s	3.073%	4.195s	1.109%	3.346s
w/o auxiliary closure loss	0.336%	0.887s	2.988%	4.272s	1.034%	3.291s
w/o primary-edit conditioning	N/A	N/A	N/A	N/A	1.090%	3.394s
w/o state loss	0.327%	0.890s	2.968%	4.159s	0.998%	3.306s

The score fields also need to remain specialized. Sharing the edit and residual head raises the gap to 0.353%, 3.017%, and 1.026% on TSP-1000, CVRP-500, and MCL RB-LARGE, respectively, with nearly unchanged inference time. Edit scores rank choices for completion, whereas residual scores must retain an assignment scale for thresholded commitment. A shared head therefore couples two prediction tasks with different uses of score magnitude. Removing residual assignment supervision produces a larger increase, reaching 0.385% on TSP-1000 and 1.109% on MCL RB-LARGE. The result follows the role of the residual objective: assignment labels anchor the score scale used by R_τ, while ranking losses alone do not specify which candidates should cross the threshold. On MCL, removing primary-edit conditioning further increases the gap from 0.993% to 1.090%. This variant is defined for the vertex-assignment setting, where the residual query receives the selected primary edit; edge-structured problems use reconstruction with $a_t = \varnothing$.

Deterministic constraint propagation affects both solution quality and the amount of downstream work. Replacing C_I with the identity mapping raises the gap on TSP-1000, CVRP-500, and MCL RB-LARGE to 0.370%, 3.106%, and 1.057%. The time effect depends on the problem realization: TSP-1000 increases from 0.888s to 1.912s, CVRP-500 remains near 4.25s, and MCL RB-LARGE decreases to 1.072s. The timing changes follow the problem-specific completion and search procedures, while the gap increase is consistent across all three benchmarks. The auxiliary closure loss has a smaller but consistent effect, with gaps of 0.336%, 2.988%, and 1.034%. Since exact propagation is still applied at inference, this ablation isolates the representation signal that teaches the shared encoder to predict forced assignments.

The state loss has the weakest effect among the additional training ablations: removing it changes the gaps to 0.327%, 2.968%, and 0.998%, while leaving inference time essentially unchanged. This pattern is consistent with its role as optional search guidance, separate from the NRP transition. Taken together, the results distinguish three sources of the final quality: residual commitments determine which unresolved assignments are retained, the separate supervised score fields make those commitments usable, and propagation exposes their deterministic consequences before completion. The auxiliary and state objectives refine this process through the shared representation and search score, respectively.

F.3 Cross-scale Generalization

We evaluate cross-scale generalization by testing each model on the reported instance sizes of the same problem. The tables cover edge-structured problems (TSP, ATSP, and CVRP) and vertex-assignment problems (MIS, MCL, and MVC). Rows indicate test sizes, columns indicate training sizes, and model initialization is given in Appendix E.2.

Table 13: Cross-scale generalization on TSP (gap, %).

Testing \ Training		TSP-50	TSP-100	TSP-500	TSP-1K
TSP-50	DIFUSCO (T_s=50) (Sun & Yang, 2023)	0.09%	0.25%	2.55%	2.71%
	Fast T2T (T_s=5,T_g=5) (Li et al., 2024)	0.09%	0.36%	1.02%	1.26%
	StruDiCO (T_s=5,T_g=5) (Wang et al., 2025)	0.01%	0.01%	0.31%	0.80%
	NRP4CO (S=16,C=64)	**0.000%**	**0.000%**	**0.000%**	**0.000%**
TSP-100	DIFUSCO (T_s=50) (Sun & Yang, 2023)	1.44%	0.23%	3.44%	3.31%
	Fast T2T (T_s=5,T_g=5) (Li et al., 2024)	0.12%	0.02%	0.40%	0.55%
	StruDiCO (T_s=5,T_g=5) (Wang et al., 2025)	0.16%	0.02%	0.34%	0.55%
	NRP4CO (S=16,C=64)	**0.004%**	**0.000%**	**0.008%**	**0.011%**
TSP-500	DIFUSCO (T_s=50) (Sun & Yang, 2023)	4.16%	3.04%	1.40%	1.85%
	Fast T2T (T_s=5,T_g=5) (Li et al., 2024)	2.67%	1.77%	0.38%	0.95%
	StruDiCO (T_s=5,T_g=5) (Wang et al., 2025)	3.46%	2.96%	0.33%	0.49%
	NRP4CO (S=16,C=64)	**1.839%**	**1.751%**	**0.168%**	**0.177%**
TSP-1K	DIFUSCO (T_s=50) (Sun & Yang, 2023)	4.54%	3.98%	2.65%	2.21%
	Fast T2T (T_s=5,T_g=5) (Li et al., 2024)	3.46%	3.08%	1.06%	0.58%
	StruDiCO (T_s=5,T_g=5) (Wang et al., 2025)	4.31%	3.22%	0.75%	0.54%
	NRP4CO (S=16,C=64)	**2.709%**	**2.977%**	**0.459%**	**0.183%**

TSP. NRP4CO achieves the lowest reported gap in every TSP comparison (Table 13). Models trained on TSP-50 and TSP-100 maintain gaps below 0.005% on the two smaller test sets, but their gaps increase to 2.709% and 2.977%, respectively, on TSP-1000. In contrast, the TSP-500 model reduces the TSP-1000 gap to 0.459%, while the TSP-1000 model attains a 0.177% gap on TSP-500. Since reconstruction and deterministic constraint propagation are shared across problem sizes, these results suggest that the learned edit and residual guidance transfers more effectively between larger-scale instances.

Table 14: Cross-scale generalization on ATSP ($S = 512, C = 1$).

Testing \ Training	ATSP-50		ATSP-100		ATSP-200		ATSP-500	
	Obj.↓	Gap↓	Obj.↓	Gap↓	Obj.↓	Gap↓	Obj.↓	Gap↓
ATSP-50	1.5548	0.021%	1.5551	0.037%	1.5553	0.050%	1.5550	0.034%
ATSP-100	1.5684	0.152%	1.5667	0.044%	1.5673	0.080%	1.5681	0.132%
ATSP-200	1.5729	0.517%	1.5701	0.342%	1.5672	0.158%	1.5690	0.274%
ATSP-500	1.5902	1.070%	1.5946	1.344%	1.5867	0.843%	1.5813	0.501%

ATSP. On ATSP, the largest test size produces the clearest difference between models (Table 14). The ATSP-50 and ATSP-100 models reach gaps of 1.070% and 1.344% on ATSP-500, while the ATSP-200 model reduces the gap to 0.843%. The ATSP-500 model stays below 0.3% on smaller test sizes and inherits the smaller models through the training sequence. With the same completion procedure at each size, the lower gap of the ATSP-200 model on ATSP-500 shows the value of training edit and residual scores on larger instances.

Table 15: Cross-scale generalization on CVRP ($S = 1, C = 64$).

Testing \ Training	CVRP-50		CVRP-100		CVRP-200		CVRP-500	
	Obj.↓	Gap↓	Obj.↓	Gap↓	Obj.↓	Gap↓	Obj.↓	Gap↓
CVRP-50	10.4108	0.421%	10.4133	0.444%	10.4116	0.429%	10.4105	0.416%
CVRP-100	15.7364	1.102%	15.6770	0.724%	15.7505	1.191%	15.7529	1.205%
CVRP-200	20.1049	2.406%	20.1317	2.557%	19.8319	1.026%	20.1516	2.640%
CVRP-500	38.6964	4.174%	38.7305	4.272%	38.6297	3.995%	38.0093	2.280%

CVRP. CVRP shows asymmetric transfer across scales (Table 15). On CVRP-50, all four training scales yield gaps between 0.416% and 0.444%, so models trained on larger instances retain their solution quality when evaluated on smaller ones. Transfer in the opposite direction deteriorates with size: the CVRP-50 model reaches gaps of 1.102%, 2.406%, and 4.174% on CVRP-100, CVRP-200, and CVRP-500, respectively. At the two largest test sizes, matched-scale training reduces the best transferred gap from 2.406% to 1.026% and from 3.995% to 2.280%. Training at the largest scale also gives the largest transferred gaps on CVRP-100 and CVRP-200, at 1.205% and 2.640%. NRP4CO uses the same route reconstruction, deterministic constraint propagation over degree, capacity, and depot constraints, and capacity-feasible completion at every scale. The asymmetric degradation is therefore consistent with scale sensitivity in the learned guidance, since larger instances require edit and residual scores to rank more eligible connections across more route components.

MIS. On MIS, the model trained on RB-[200-300] reaches a 6.431% gap on RB-[800-1200], compared with 2.942% for the model trained on larger RB graphs (Table 16). Testing the latter on RB-[200-300] changes the gap from 0.575% to 0.780%. The MIS decision encoding and deterministic constraint propagation rules are the same at both sizes. Edit and residual scores trained on smaller graphs provide less effective guidance as the number of eligible vertices grows.

Table 16: Cross-scale generalization on MIS ($S = 64, C = 16$).

Testing \ Training	RB-[200-300]		RB-[800-1200]	
	Obj.↑	Gap↓	Obj.↑	Gap↓
RB-[200-300]	19.970	0.575%	19.928	0.780%
RB-[800-1200]	40.186	6.431%	41.702	2.942%

MCL and MVC. On MCL, the model trained on RB-SMALL generalizes to RB-LARGE with an objective of 39.982 and a gap of 0.457%, compared with 39.860 and 0.662% for the model trained on RB-LARGE. On RB-SMALL, the two models attain gaps of 0.000% and 0.071%, respectively, so the RB-SMALL model has the lower gap on both test sizes in this comparison.

Table 17: Cross-scale generalization on MCL ($S = 64, C = 8$).

Testing \ Training	RB-SMALL		RB-LARGE	
	Obj.↑	Gap↓	Obj.↑	Gap↓
RB-SMALL	19.082	0.000%	19.070	0.071%
RB-LARGE	39.982	0.457%	39.860	0.662%

For MVC, training and testing on the same scale gives the lower gap: the models trained on RB-SMALL and RB-LARGE obtain gaps of 0.024% and 0.090% on their corresponding test sets, while cross-scale testing increases these values to 0.060% on RB-SMALL and 0.129% on RB-LARGE. The objective values follow the same ordering, with 205.814 versus 205.886 on RB-SMALL and 969.098 versus 969.474 on RB-LARGE. Both problems use the same decision encoding and deterministic constraint propagation rules at the two scales. All four MCL gaps remain below 0.7%, and MVC gaps remain below 0.13%.

Table 18: Cross-scale generalization on MVC ($S = 64, C = 8$).

Testing \ Training	RB-SMALL		RB-LARGE	
	Obj.↓	Gap↓	Obj.↓	Gap↓
RB-SMALL	205.814	0.024%	205.886	0.060%
RB-LARGE	969.474	0.129%	969.098	0.090%

F.4 Generalization to TSPLIB

We evaluate NRP4CO on 49 TSPLIB (Reinelt, 1991) instances with 51–1002 nodes. The model trained on random TSP-100 instances is used for the 51–200-node group, and the model trained on random TSP-500 instances is used for the 225–1002-node group. For NRP4CO, input coordinates follow the normalization in Appendix D.1 and the setting is S=16,C=64. Tables 19 and 20 report the gap for each instance using the definition in Section 5.1.

Table 19: Solution quality for methods trained on random TSP-100 instances and evaluated on TSPLIB instances with 51–200 nodes.

Instances	DIFUSCO	T2T	Fast T2T	COExpander	StruDiCO	NEXCO	GenSCO	NRP4CO
eil51	2.82%	0.14%	0.00%	0.000%	0.00%	0.00%	0.000%	0.000%
berlin52	0.00%	0.00%	0.00%	0.000%	0.00%	0.00%	0.000%	0.000%
st70	0.00%	0.00%	0.00%	0.000%	0.00%	0.00%	0.000%	0.000%
eil76	0.34%	0.00%	0.00%	0.000%	0.00%	0.00%	0.000%	0.000%
pr76	1.12%	0.40%	0.00%	0.000%	-0.00%	0.00%	0.000%	0.000%
rat99	0.09%	0.09%	0.00%	0.000%	0.00%	0.00%	0.000%	-0.000%
kroA100	0.10%	0.00%	0.00%	0.000%	0.00%	0.00%	0.000%	0.000%
kroB100	2.29%	0.74%	0.65%	0.742%	0.00%	0.00%	0.000%	0.000%
kroC100	0.00%	0.00%	0.00%	0.335%	0.00%	0.00%	0.000%	-0.000%
kroD100	0.07%	0.00%	0.00%	0.000%	0.00%	0.00%	0.000%	0.000%
kroE100	3.83%	0.27%	0.00%	0.000%	2.15%	0.00%	0.000%	-0.000%
rd100	0.08%	0.00%	0.00%	0.000%	0.11%	0.00%	0.000%	-0.000%
eil101	0.03%	0.00%	0.00%	0.000%	0.00%	0.00%	0.000%	0.000%
lin105	0.00%	0.00%	0.00%	0.000%	0.18%	0.54%	0.000%	0.000%
pr107	0.91%	0.61%	0.62%	0.100%	0.18%	0.08%	0.000%	-0.000%
pr124	1.02%	0.60%	0.08%	0.783%	-0.00%	0.58%	0.000%	0.000%
bier127	0.94%	0.54%	1.50%	0.000%	0.04%	0.66%	0.000%	0.000%
ch130	0.29%	0.06%	0.00%	0.000%	0.24%	0.00%	0.000%	-0.000%
pr136	0.19%	0.10%	0.01%	0.597%	0.04%	0.00%	0.000%	0.010%
pr144	0.80%	0.50%	0.39%	0.146%	0.00%	0.00%	0.000%	0.000%
ch150	0.57%	0.49%	0.00%	0.000%	0.04%	0.00%	0.000%	0.000%
kroA150	0.34%	0.14%	0.00%	0.893%	0.24%	0.00%	0.000%	0.000%
kroB150	0.30%	0.00%	0.07%	0.100%	0.00%	0.02%	0.340%	0.051%
pr152	1.69%	0.83%	0.19%	0.191%	0.69%	0.68%	0.187%	-0.000%
u159	0.82%	0.00%	0.00%	0.000%	0.00%	0.00%	0.000%	-0.000%
rat195	1.48%	1.27%	0.79%	0.419%	-0.00%	0.11%	0.194%	0.340%
d198	3.32%	1.97%	0.86%	0.488%	-0.00%	0.00%	0.751%	0.210%
kroA200	2.28%	0.57%	0.49%	0.056%	0.00%	0.00%	0.160%	0.000%
kroB200	2.35%	0.92%	2.50%	0.851%	-0.00%	0.00%	0.098%	0.003%
Mean	0.97%	0.35%	0.28%	0.197%	0.14%	0.09%	0.060%	0.021%

NRP4CO obtains the best mean gap in both groups: 0.021% for 51–200 nodes and 0.283% for 225–1002 nodes. The corresponding means for GenSCO (Li et al., 2025), the best competing method in each table, are 0.060% and 0.373%.

Table 20: Solution quality for methods trained on random TSP-500 instances and evaluated on TSPLIB instances with 225–1002 nodes.

Instances	DIFUSCO	T2T	Fast T2T	COExpander	GenSCO	NRP4CO
ts225	4.95%	2.24%	1.37%	1.115%	0.517%	-0.000%
tsp225	3.25%	1.69%	0.81%	0.200%	0.000%	0.719%
pr226	4.22%	0.84%	0.34%	0.551%	0.211%	-0.000%
gil262	2.18%	0.96%	0.18%	0.166%	0.000%	0.217%
pr264	0.92%	0.92%	0.73%	0.000%	0.000%	-0.000%
a280	1.39%	1.39%	0.10%	0.000%	0.000%	0.551%
pr299	1.46%	1.46%	1.40%	1.305%	0.011%	0.444%
lin318	2.95%	1.73%	1.21%	0.025%	0.025%	0.229%
rd400	1.18%	0.44%	0.08%	0.100%	0.030%	0.088%
fl417	3.30%	3.30%	2.01%	1.095%	0.596%	0.385%
pr439	2.73%	1.63%	0.50%	0.822%	0.106%	0.258%
pcb442	2.59%	1.70%	0.61%	0.467%	0.015%	0.085%
d493	1.81%	1.81%	1.43%	1.223%	3.872%	0.697%
u574	2.50%	1.85%	0.94%	1.044%	0.109%	0.096%
rat575	2.32%	1.29%	1.43%	0.509%	0.000%	0.502%
p654	7.49%	1.19%	1.67%	0.714%	1.023%	0.339%
d657	4.86%	2.40%	0.64%	0.676%	0.457%	0.284%
u724	2.05%	2.05%	1.41%	1.137%	0.040%	0.404%
rat783	3.04%	1.88%	1.03%	0.502%	0.068%	0.072%
pr1002	–	–	–	0.618%	–	0.476%
Mean	2.90%	1.62%	0.94%	0.613%	0.373%	0.283%

F.5 Generalization to CVRPLIB

We evaluate NRP4CO on 70 CVRPLIB (Uchoa et al., 2017) instances with 101–513 nodes. The model trained on random CVRP-100 instances is used for the 101–148-node group, the CVRP-200 model for the 153–351-node group, and the CVRP-500 model for the 359–513-node group. For NRP4CO, inputs follow the normalization in Appendix D.1 and the setting is S=16,C=64. Tables 21 to 23 list the reference objective, the objective returned by NRP4CO, and the resulting gap.

Table 21: Solution quality for NRP4CO trained on random CVRP-100 instances and evaluated on CVRPLIB instances with 101–148 nodes.

Instances	Exact	Obj.↓	Gap↓
X-n101-k25	27.905	28.285	1.359%
X-n106-k14	26.362	26.609	0.938%
X-n110-k13	14.979	14.978	-0.008%
X-n115-k10	12.933	12.943	0.078%
X-n120-k6	13.437	13.440	0.022%
X-n125-k30	56.164	57.272	1.973%
X-n129-k18	29.005	29.183	0.617%
X-n134-k13	11.351	11.463	0.981%
X-n139-k10	13.678	13.697	0.138%
X-n143-k7	15.776	15.879	0.653%
X-n148-k46	43.671	44.287	1.410%
Mean	24.115	24.367	0.742%

F.6 Runtime Breakdown

Table 24 decomposes the average inference time per instance on two edge-structured problems, TSP and CVRP, and two vertex-assignment problems, MIS and MCL. The hardware follows Section 5.1, and the components correspond to the search procedure in Appendix D.4.

Table 22: Solution quality for NRP4CO trained on random CVRP-200 instances and evaluated on CVRPLIB instances with 153–351 nodes.

Instances	Exact	Obj.↓	Gap↓
X-n153-k22	21.572	21.808	1.093%
X-n157-k13	17.572	17.720	0.846%
X-n162-k11	14.412	14.460	0.334%
X-n167-k10	20.766	20.821	0.263%
X-n172-k51	46.974	47.653	1.444%
X-n176-k26	47.816	48.993	2.463%
X-n181-k23	26.809	27.021	0.792%
X-n186-k15	24.251	24.521	1.111%
X-n190-k8	17.037	17.244	1.214%
X-n195-k51	44.274	44.880	1.370%
X-n200-k36	60.084	61.533	2.412%
X-n204-k19	19.588	19.870	1.438%
X-n209-k16	30.691	31.177	1.583%
X-n214-k11	10.879	11.183	2.790%
X-n219-k73	117.601	118.492	0.758%
X-n223-k34	40.478	41.192	1.762%
X-n228-k23	25.747	26.192	1.731%
X-n233-k16	19.297	19.652	1.840%
X-n237-k14	27.078	27.390	1.154%
X-n242-k48	82.757	84.318	1.886%
X-n247-k50	37.966	39.051	2.857%
X-n251-k28	38.885	39.611	1.869%
X-n256-k16	18.942	19.098	0.820%
X-n261-k13	26.591	27.245	2.458%
X-n266-k58	75.489	77.189	2.252%
X-n270-k35	35.432	36.053	1.751%
X-n275-k28	22.924	23.651	3.175%
X-n280-k17	33.503	34.173	1.997%
X-n284-k15	20.421	21.014	2.907%
X-n289-k60	95.351	97.699	2.462%
X-n294-k50	47.315	48.174	1.817%
X-n298-k31	34.247	35.109	2.518%
X-n303-k21	21.961	22.366	1.844%
X-n308-k13	25.980	26.469	1.880%
X-n313-k71	94.437	96.848	2.553%
X-n317-k53	78.456	79.151	0.886%
X-n322-k28	29.846	30.555	2.377%
X-n327-k20	27.681	28.358	2.448%
X-n331-k15	31.155	31.956	2.571%
X-n336-k84	139.283	142.859	2.568%
X-n344-k43	42.194	43.203	2.389%
X-n351-k40	25.948	26.630	2.628%
Mean	40.945	41.728	1.841%

The largest component on TSP-500 is local search at 43.4%, followed by scorer forward passes at 39.7%. On CVRP-100, scorer forward passes account for 66.9% and local search for 18.7%. For MIS and MCL, scorer forward passes take 56.0% and 60.8%, respectively, while completion takes 34.7% and 27.4%. Completion is the second largest component on both vertex-assignment problems.

Residual selection accounts for 0.1%–1.3% of total time across the four settings, and deterministic constraint propagation accounts for 0.4%–3.2%. Their combined share is at most 4.5%.

Table 23: Solution quality for NRP4CO trained on random CVRP-500 instances and evaluated on CVRPLIB instances with 359–513 nodes.

Instances	Exact	Obj.↓	Gap↓
X-n359-k29	51.512	52.726	2.358%
X-n367-k17	24.741	25.464	2.921%
X-n376-k94	147.881	149.049	0.790%
X-n384-k52	66.086	67.818	2.621%
X-n393-k38	38.386	39.326	2.449%
X-n401-k29	66.204	67.127	1.395%
X-n411-k19	19.741	20.171	2.182%
X-n420-k130	108.137	110.139	1.852%
X-n429-k61	65.777	67.284	2.291%
X-n439-k37	36.484	37.195	1.949%
X-n449-k29	55.282	57.052	3.203%
X-n459-k26	25.279	26.068	3.119%
X-n469-k138	222.364	228.862	2.922%
X-n480-k70	89.842	91.966	2.364%
X-n491-k59	66.591	68.517	2.891%
X-n502-k39	69.262	70.459	1.729%
X-n513-k21	24.273	24.913	2.636%
Mean	69.285	70.832	2.334%

Table 24: Runtime breakdown of NRP4CO inference.

Component	TSP-500 S=1,C=64	CVRP-100 S=1,C=64	MIS RB-[800-1200] S=64,C=4	MCL RB-SMALL S=64,C=8
Total	0.997s	0.418s	3.169s	0.825s
Scorer forward	0.396s (39.7%)	0.279s (66.9%)	1.774s (56.0%)	0.501s (60.8%)
Primary edit selection	–	–	0.021s (0.7%)	0.006s (0.8%)
Residual selection	0.013s (1.3%)	0.000s (0.1%)	0.025s (0.8%)	0.008s (0.9%)
Constraint propagation	0.032s (3.2%)	0.002s (0.4%)	0.025s (0.8%)	0.008s (0.9%)
Completion	0.058s (5.8%)	0.007s (1.8%)	1.100s (34.7%)	0.226s (27.4%)
Local search	0.433s (43.4%)	0.078s (18.7%)	–	–
Reconstruction	0.013s (1.3%)	0.003s (0.7%)	–	0.006s (0.8%)
Objective and incumbent update	0.004s (0.4%)	0.001s (0.2%)	0.001s (0.0%)	0.001s (0.1%)
Other overhead	0.049s (5.0%)	0.047s (11.2%)	0.225s (7.1%)	0.069s (8.4%)

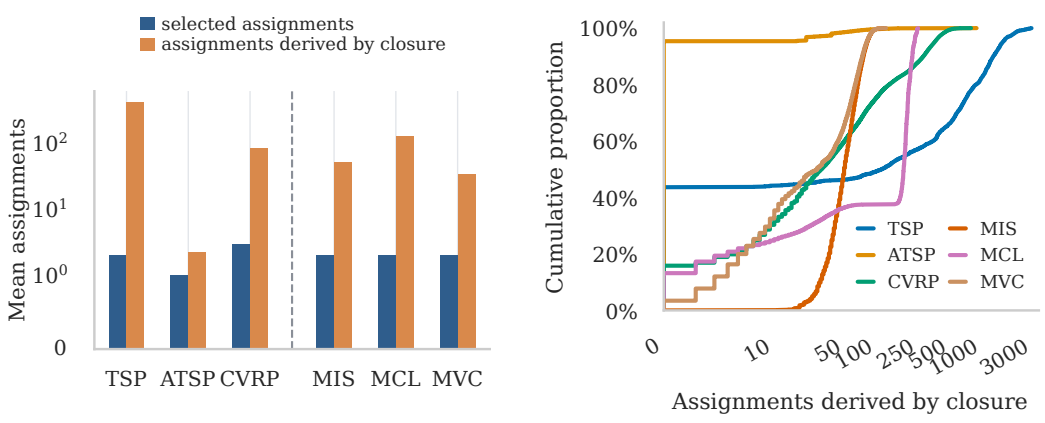
F.7 Assignment Resolution Across Problems

We analyze assignment resolution at the level of each active transition. For a transition, selected assignments are the commitments in A_t, while assignments derived by closure are the additional units resolved when C_I is saturated. Figure 3 reports the mean attribution and the distribution of closure assignments across the six problems.

Figure 3 (a) shows that closure resolves more assignments per active transition than the selected set on TSP, CVRP, MIS, MCL, and MVC. The difference is smaller on ATSP, where the two sources contribute similar counts. Figure 3 (b) exposes the corresponding distributional differences: TSP has the widest right tail, ATSP is concentrated at low closure counts, and CVRP, MIS, MCL, and MVC occupy intermediate ranges. The two panels together separate the assignments chosen by the learned transition from the additional assignments recorded by deterministic propagation.

F.8 Influence of Local Search

NRP4CO applies 2-opt after tour completion and after perturbing an incumbent tour before reconstruction. For CVRP, 2-opt is applied to the customer-order tour before the feasible split. Each enabled local-search call uses the round schedule in Table 25. To isolate its contribution, we dis-



(a) (b)

Figure 3: Assignment resolution across the six problems. (a) shows mean selected assignments and assignments derived by closure per active transition. (b) shows the empirical cumulative distribution of assignments derived by closure; the horizontal scale uses $\log(1 + x)$ for readability while tick labels show assignment counts. The first three problems are edge-structured problems and the last three are vertex-assignment problems.

able these 2-opt calls while keeping the trained model, trajectory count S, cycle budget C, and the remaining search procedure unchanged. Tables 26 and 27 report the objective, optimality gap, and average inference time per instance on TSP and ATSP.

Table 25: Default local-search rounds by test size.

Problem	**Test size**				
	50	**100**	**200**	**500**	**1000**
TSP	32	64	–	256	512
ATSP	32	64	128	256	–
CVRP	32	64	128	256	–

The schedule increases the local-search budget with test size: 32 rounds are used at size 50, followed by 64, 128, 256, and 512 rounds at the reported larger scales. The table covers the tested sizes for each edge-structured problem, so a dash marks a scale that is not part of that benchmark family. For TSP and ATSP, the rounds refine completed tours and reconstructed incumbents; ATSP evaluates exchanges with directed costs. In CVRP, they refine the customer order before route splitting. Local-search time is included in the inference times reported throughout the paper.

Table 26: Influence of local search on TSP. "− Local Search" disables 2-opt during inference.

Method	**TSP-50**			**TSP-100**			**TSP-500**			**TSP-1000**		
	Obj.↓	Gap↓	Time↓	Obj.↓	Gap↓	Time↓	Obj.↓	Gap↓	Time↓	Obj.↓	Gap↓	Time↓
NRP4CO (S=1,C=16)	5.688	0.0024%	0.015s	7.757	0.0112%	0.025s	16.585	0.2372%	0.285s	23.191	0.3154%	0.888s
− Local Search	5.689	0.0318%	0.011s	7.761	0.0642%	0.016s	16.820	1.6582%	0.257s	23.891	3.3433%	0.491s
NRP4CO (S=1,C=64)	5.688	0.0013%	0.039s	7.757	0.0089%	0.080s	16.575	0.1755%	0.997s	23.165	0.2028%	2.710s
− Local Search	5.688	0.0044%	0.039s	7.758	0.0336%	0.041s	16.660	0.6898%	0.882s	23.518	1.7285%	1.528s
NRP4CO (S=16,C=64)	5.688	-0.0000%	0.429s	7.756	0.0003%	1.301s	16.574	0.1676%	9.896s	23.160	0.1827%	38.173s
− Local Search	5.688	0.0005%	0.417s	7.758	0.0250%	1.030s	16.651	0.6364%	8.986s	23.506	1.6748%	19.724s

On TSP, local search lowers the gap at every problem size and search budget, with larger gains on TSP-500 and TSP-1000. For $S = 1, C = 16$, 2-opt reduces the TSP-1000 gap from 3.3433% to 0.3154%, while inference time increases from 0.491s to 0.888s. The larger settings exhibit the

same pattern: the gap falls from 1.7285% to 0.2028% for $S = 1, C = 64$, and from 1.6748% to 0.1827% for $S = 16, C = 64$. At the latter setting, the corresponding runtime grows from 19.724s to 38.173s. The improvement comes from applying edge exchanges to completed tours throughout search. Their runtime cost becomes most visible on the largest instances.

Table 27: Influence of local search on ATSP. "− Local Search" disables 2-opt during inference.

Method	ATSP-50			ATSP-100			ATSP-200			ATSP-500		
	Obj.↓	Gap↓	Time↓	Obj.↓	Gap↓	Time↓	Obj.↓	Gap↓	Time↓	Obj.↓	Gap↓	Time↓
NRP4CO (S=64,C=1)	1.5559	0.0913%	0.050s	1.5680	0.1262%	0.147s	1.5693	0.2931%	0.644s	1.5821	0.5529%	3.341s
− Local Search	1.5560	0.0962%	0.057s	1.5682	0.1357%	0.159s	1.5694	0.2934%	0.610s	1.5821	0.5529%	3.416s
NRP4CO (S=256,C=1)	1.5550	0.0350%	0.179s	1.5670	0.0631%	0.520s	1.5679	0.2008%	2.287s	1.5816	0.5203%	12.103s
− Local Search	1.5551	0.0377%	0.177s	1.5671	0.0660%	0.529s	1.5679	0.2021%	2.239s	1.5818	0.5318%	12.201s

Local search has a smaller effect on ATSP. With $S = 256, C = 1$, the gap decreases from 0.5318% to 0.5203% on ATSP-500. The reductions on ATSP-50/100/200 range from 0.0013 to 0.0029 percentage points. For $S = 64, C = 1$, the largest reduction is 0.0095 percentage points on ATSP-100, while the ATSP-500 gap remains 0.5529%. The measured runtimes differ by at most 0.098s across all paired settings. ATSP completion solves a constrained linear assignment and merges its cycles before 2-opt, so local search provides a small refinement to each completed proposal.

F.9 Residual Threshold Sensitivity

We sweep the residual threshold τ on TSP-100 with $S = 1, C = 16$, CVRP-100 with $S = 1, C = 16$, and MCL RB-SMALL with $S = 32, C = 1$. The "off" setting disables residual selection, and the dotted line marks the reference threshold $\tau = 0.25$. Figure 4 reports the resulting Gap and the fraction of residual queries that accept at least one assignment.

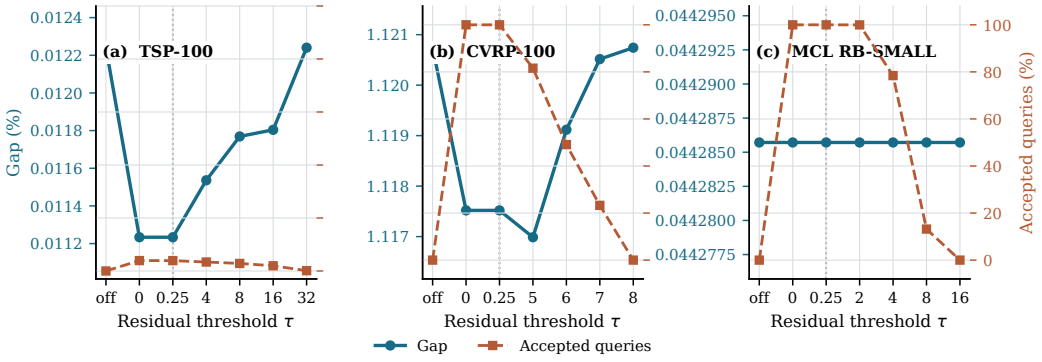


Figure 4: Residual threshold sensitivity on TSP-100, CVRP-100, and MCL RB-SMALL.

On TSP-100, the reference threshold gives a Gap of 0.01123%, matching the result at $\tau = 0$. Increasing the threshold reduces the rate of accepted queries from 3.91% at $\tau = 0$ to 0.11% at $\tau = 32$, while the Gap increases to 0.01224%. CVRP-100 shows a similar change in residual activity. The rate of accepted queries decreases from 100% at $\tau = 0.25$ to 23.28% at $\tau = 7$ and reaches zero at $\tau = 8$. The Gap varies from 1.11752% at the reference threshold to 1.11699% at $\tau = 5$ and 1.12074% when residual selection is disabled.

For MCL RB-SMALL, the rate of accepted queries decreases from 100% at $\tau = 0.25$ to 13.19% at $\tau = 8$ and zero at $\tau = 16$, while the Gap remains 0.04429% throughout the sweep. The threshold changes the amount of residual propagation recorded in a transition, with a smaller effect on the final objective for this setting. We use $\tau = 0.25$ as the reference point for the reported models, and the sweep shows the range in which residual selection changes the search trajectory.

G VISUALIZATION OF THE NRP4CO SOLVING PROCESS

G.1 TRAVELING SALESMAN PROBLEM

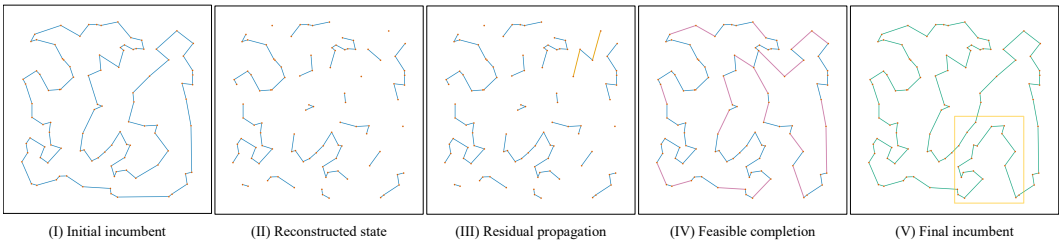


Figure 5: Visualization of TSP solving process.

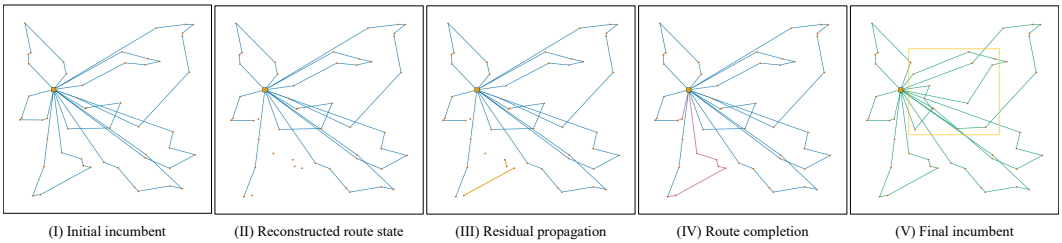


Figure 6: Visualization of CVRP solving process.

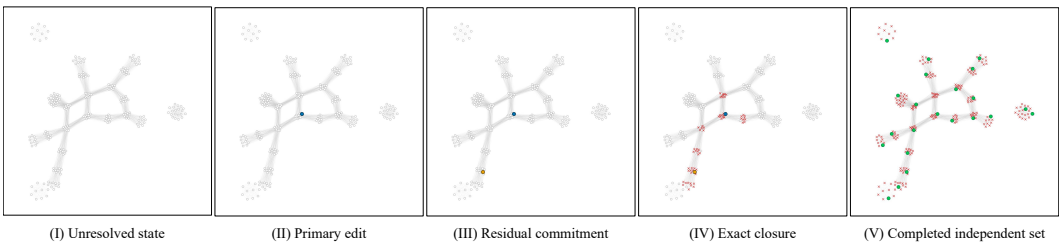


Figure 7: Visualization of MIS solving process.

In Figure 5, orange-red points denote cities. Blue edges denote the current or retained structure, orange edges are residual commitments, pink edges are added during completion, and green edges form the final tour. At Step 0, the model predicts edge scores from an empty state, constructs a feasible tour, and applies 64 rounds of 2-opt. The resulting blue tour visits all 100 cities and has length 7.906.

At Step 1, NRP4CO reconstructs a partial state by retaining 80 edges from the incumbent. The retained blue segments remain fixed, while unresolved regions are reopened for reconnection. At Step 2, the residual selector commits two orange edges. TSP closure then applies degree and subtour rules and forces an edge when the remaining eligible connections exactly satisfy a city's required degree.

At Step 3, edit scores complete the retained paths into a tour. Pink edges mark the new connections. Its length is 7.948, which is larger than the incumbent length 7.906, so the strict improvement test rejects this candidate for the current cycle. After later reconstruction, residual propagation, closure, completion, and 2-opt cycles, Step 4 gives the final incumbent of length 7.842. The final tour improves on the initial incumbent by 0.064, or approximately 0.81%.

G.2 CAPACITATED VEHICLE ROUTING PROBLEM

In Figure 6, orange-red circles denote customers, and the orange square with a dark border denotes the depot. Blue connections denote the current or retained route structure, orange connections are residual commitments, pink connections are added during capacity-feasible completion, and green

connections form the final route set. Multiple connections meet at the depot because every route starts and ends there.

At Step 0, the model constructs an initial feasible solution and applies local search. The blue route set covers 50 customers with 8 routes and has total cost 11.354. Step 1 retains 50 connections from this incumbent. Isolated customers and open route endpoints mark the portions made unresolved for the next prediction. At Step 2, the residual selector commits two orange connections. CVRP closure updates customer degrees, route components, component demand, capacity feasibility, and depot connections after each commitment. It records infeasible connections as excluded assignments.

At Step 3, the completion procedure adds pink connections while preserving single visitation, depot return, and vehicle capacity. The candidate cost is 11.354, equal to the initial incumbent, so the strict improvement rule keeps the current solution. After subsequent search cycles, Step 4 retains 8 routes with reorganized connections and total cost 11.147. The final cost improves by 0.207, or approximately 1.82%.

G.3 Maximum Independent Set

In Figure 7, hollow light-gray circles are unresolved vertices, blue filled circles are selected by the primary edit, orange filled circles are residual commitments, red crosses are excluded vertices, and bright-green circles form the final independent set. The pale-gray edges are the input graph. Peripheral clusters are separate connected components of the fixed graph, and their positions remain unchanged across steps.

At Step 0, the graph contains 240 vertices and 3127 edges, and every vertex is unresolved. Step 1 selects one primary vertex in blue. Step 2 conditions the residual prediction on that selection and commits one additional compatible vertex in orange; the two selected vertices have no edge between them. Step 3 applies MIS closure: every unresolved neighbor of a selected vertex is forced to the excluded state. This resolves 85 vertices in total. Vertices that remain hollow are still compatible with the current commitments and can participate in completion.

At Step 4, edit scores complete the independent set over the remaining feasible vertices. Both the green set and the reference solution contain 22 vertices. The graph can contain multiple maximum independent sets of this size, so their selected vertices can differ.

H Additional Experiments on Maximum Cut

We additionally evaluate NRP4CO on maximum cut (MCut), using the vertex-assignment model and search procedure.

H.1 Problem Formulation and Inference

Encoding and closure. Each $x_v \in \{0,1\}$ specifies one side of a partition. All binary assignments are feasible, and

$$\mathcal{X}_I = \{0,1\}^V, \qquad f_I(x) = -\sum_{\{u,v\}\in E} \mathbf{1}\{x_u \neq x_v\}.$$

The MCut objective is the unweighted cut cardinality. Exact closure is the identity, $C_I(s) = s$, so $c_t = 0$ in Equation (8). Primary selection chooses the unresolved vertex with largest absolute edit score and assigns it to the side indicated by the sign. The residual selector does the same with residual scores, adding at most one assignment when its absolute score exceeds τ.

Completion and search. Completion assigns each unresolved vertex to a partition using edit scores and cut contributions. The resulting complete partition preserves all input assignments. Search follows the vertex-assignment procedure in Appendix D.4. A subset of trajectories reopens the incumbent partition, retaining some partition assignments before selecting the next primary edit.

Propagation progress. A primary edit and one residual commitment resolve two units. If no residual score passes the threshold, $R_t = \varnothing$ and the transition uses the primary edit alone. An empty combined edit set has $g_t = 0$.

H.2 Training Objectives

Partial contexts and assignment supervision. Training contexts retain assignments on both sides of the reference partition. Edit and residual outputs use unweighted binary classification on all unresolved vertices, denoted by $\mathcal{B}_{\text{plain}}$, the unweighted form of Equation (21). Auxiliary supervision also uses reference assignment labels on unresolved vertices and omits the neutral term in the auxiliary objective of Appendix D.3.

State target. Let $E^\star_{\text{cut}} = \{\{u, v\} \in E : x^\star_u \neq x^\star_v\}$. The optional state target is

$$y^q = \frac{|\{\{u, v\} \in E^\star_{\text{cut}} : s_u = -1 \text{ or } s_v = -1\}|}{\max\{|E^\star_{\text{cut}}|, 1\}}. \tag{27}$$

H.3 Experimental Setup and Results

We use BA-SMALL and BA-LARGE from the benchmark families in ML4CO-Bench-101 (Ma et al., 2025b), with 500 test instances per dataset. Gurobi (Gurobi Optimization, 2023) supplies reference objectives under a time limit. The metrics and hardware follow Section 5.1. Table 28 reports the results for two search configurations, $(S, C) = (64, 4)$ and $(32, 32)$.

Table 28: Additional results on MCut. * denotes reported results from (Ma et al., 2025a).

Method	BA-SMALL			BA-LARGE		
	Obj.↑	Gap↓	Time↓	Obj.↑	Gap↓	Time↓
Gurobi* (Gurobi Optimization, 2023)	727.84	0.00%	60.61s	2936.89	0.00%	300.21s
DiffUCO: CE (S=4,F=1)* (Sanokowski et al., 2024)	727.53	0.06%	0.61s	2989.46	-1.77%	2.70s
COExpander (S=4,D_s=1,T_s=20) (Ma et al., 2025a)	728.32	-0.05%	0.25s	2960.66	-0.80%	0.67s
NRP4CO (S=64,C=4)	725.89	0.28%	0.65s	2941.21	-0.14%	8.14s
NRP4CO (S=32,C=32)	727.91	0.00%	2.46s	2963.69	-0.90%	19.68s

COExpander (Ma et al., 2025a) has a smaller gap on BA-SMALL. On BA-LARGE, NRP4CO reaches a gap of -0.90% compared with -0.80% for COExpander, a 12.5% Gap Reduction with a speedup of $0.03\times$. DiffUCO (Sanokowski et al., 2024) is stronger and faster on this dataset, reaching -1.77% in 2.70s.