QAP-SCP: LEARNING STRUCTURAL COMPATIBILITY PRIORS FOR QUADRATIC ASSIGNMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

The quadratic assignment problem (QAP) remains challenging under limited search budgets because solution quality depends strongly on which search basins are reached. Rather than replacing combinatorial search, we propose QAP-SCP, a learning framework that predicts where search should begin. Given a QAP instance, QAP-SCP constructs a compatibility-aware structural representation that captures interaction profiles, relative structural roles, and assignment-specific compatibility, and maps them to an additive assignment score matrix. The scorer is trained with staged search-aware supervision: exact local refinement provides both a refined assignment target and exchange-geometry signals, while a compatibility module is introduced as a zero-initialized correction to a learned structural prior. At inference, deterministic and perturbed decoding convert the learned score matrix into a diverse multi-start proposal portfolio, which is filtered using the exact QAP objective and passed to a replaceable search backend. This design preserves exact objective evaluation and backend search dynamics while using learning only to guide initialization. Experiments on QAPLIB, Taixxeyy, and synthetic QAP distributions show that QAP-SCP achieves competitive standalone construction quality and substantially reduces long-tail failures on basin-sensitive instances, while maintaining low final gaps on standard QAP benchmarks. Ablations further show the contributions of compatibility-aware representations, exchange-geometry supervision, and learned proposal diversification.

1 INTRODUCTION

The quadratic assignment problem (QAP) assigns n facilities to n locations while accounting for pairwise flow and distance. It is a classical NP-hard problem with a mature literature of tabu, memetic, and breakout local search methods (Taillard, 1991; Benlic & Hao, 2013). Strong search can produce high-quality solutions, but under limited wall-clock budgets its performance depends heavily on the quality and diversity of the initial search basins. This issue is especially pronounced on structured stress-test families such as Taixxeyy, where transposition-based metaheuristics can exhibit severe long-tail failures (Drezner et al., 2005).

Recent learning-based QAP solvers construct complete permutations, learn distributions over assignments, or score local improvement moves (Tan & Mu, 2024; Pan et al., 2026). These approaches demonstrate that learned models can capture useful assignment structure, but they also raise a broader design question: should learning replace combinatorial search, or should it learn where search should begin? Our diagnostics show that static construction quality and post-search quality are only weakly coupled, suggesting that a good initial assignment is not necessarily the one that leads to the best solution after refinement. We therefore treat learning as a mechanism for constructing a global proposal prior, while retaining exact QAP evaluation and conventional search for trajectory exploration, move acceptance, and final solution selection.

Based on this view, we develop QAP-SCP, a compatibility-aware framework for learning QAP proposal priors. The method first builds a structural representation that captures three complementary forms of assignment evidence: local interaction profiles, relative structural roles, and assignment-specific compatibility. A compatibility-guided scorer then converts these signals into a shared assignment score matrix by combining a structural assignment prior with an additive pair-specific correction. Training uses a staged search-aware procedure: exact local refinement provides both a

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.

search-refined assignment target and local exchange-geometry supervision, after which the compatibility module is fitted as a zero-initialized refinement of the learned structural prior. At inference, deterministic and perturbed decoding produce a diverse portfolio of complete assignments that is filtered and passed to a replaceable search backend. The learned model therefore does not replace the search dynamics; it learns where search should start. Our contributions are:

- We introduce a *compatibility-aware structural representation* for QAP that jointly captures interaction profiles, relative structural roles, and assignment-specific compatibility, providing a compact basis for scoring all facility–location pairs.
- We develop *compatibility-guided assignment scoring* with *staged search-aware learning*, first establishing a structural prior and then refining it with assignment-specific compatibility using search-refined supervision and exact exchange geometry.
- We formulate a *backend-agnostic multi-start proposal portfolio* that converts the learned score matrix into diverse complete assignments for downstream search. Across QAPLIB, Taixxeyy, and synthetic benchmarks, QAP-SCP achieves competitive construction quality and substantially reduces long-tail failures on basin-sensitive Taixxeyy instances while retaining low gaps on QAPLIB.

2 Related Work

Traditional QAP heuristics. QAP solvers combine exact or bounded optimization, fast move deltas, and diversification mechanisms; surveys cover branch-and-bound, construction, simulated annealing, tabu, genetic, and memetic methods (Koopmans & Beckmann, 1957; Loiola et al., 2007). Ro-TS randomizes tabu tenure around repeated 2-swap descent (Taillard, 1991), hybrid genetic search combines population diversity with local improvement (Drezner, 2003), and BLS adapts perturbations after stagnation (Benlic & Hao, 2013). QAPLIB is the standard heterogeneous benchmark (Burkard et al., 1997); Taixxeyy is a stress test for transposition-search failures (Drezner et al., 2005). We retain these search dynamics and learn only complete proposals for a replaceable heuristic backend.

End-to-end learning for assignments. Learning for combinatorial optimization spans direct construction, learned improvement, and learning-guided search (Bengio et al., 2021; Cappart et al., 2023). Neural QAP and graph-matching methods predict assignment matrices or permutation distributions: NGM explicitly builds an n^2-vertex association graph, while classical counterparts include IPFP, reweighted random walks, and FAQ-style doubly stochastic relaxations (Nowak et al., 2018; Wang et al., 2022; Leordeanu et al., 2009; Cho et al., 2010; Vogelstein et al., 2015). BiQAP learns an instance transformation around a differentiable lower level (Shi et al., 2025); PLMA uses cross-graph attention for additive permutation scores and warm-started test-time MCMC (Pan et al., 2026). QAP-SCP instead uses explicit low-dimensional descriptors, complete Hungarian decoding (Kuhn, 1955), and fixed deployment parameters. QAP instance-space analysis motivates the same emphasis on structural descriptors for algorithm selection and benchmark coverage (Christiansen & Smith-Miles, 2025); we use them directly as inputs to an assignment scorer.

Learning-guided construction and search. Neural methods may construct solutions autoregressively (Bello et al., 2016; Kool et al., 2019), learn local rewrites (Chen & Tian, 2019; Wu et al., 2022), or modify a problem before a conventional solver (Wang et al., 2021). SAWT scores QAP exchanges conditioned on the incumbent (Tan & Mu, 2024); active search, DIMES, and neural large-neighborhood search adapt or repair solutions at test time (Hottung et al., 2022; Qiu et al., 2022; Hottung & Tierney, 2022). POMO and Sym-NCO motivate diverse multi-start proposals (Kwon et al., 2020; Kim et al., 2022). In contrast, QAP-SCP learns an instance-conditioned assignment prior, decodes it into a diverse proposal portfolio, and delegates trajectory exploration and budget allocation to exact probes and the downstream search backend.

3 Problem Formulation and Motivation

Let $F \in \mathbb{R}^{n\times n}$ be a facility-flow matrix and $D \in \mathbb{R}^{n\times n}$ a location-distance matrix. For a permutation $\pi \in \Pi_n$, the Koopmans–Beckmann QAP is

$$\min_{\pi\in\Pi_n} C(\pi) = \sum_{i=1}^{n}\sum_{j=1}^{n} F_{ij} D_{\pi(i)\pi(j)}. \tag{1}$$

For QAPLIB and Taixxeyy, we report the percentage gap to a best-known cost $C^\star$ as $100(C(\pi) - C^\star)/|C^\star|$. The synthetic benchmark does not provide common best-known values, so its main table uses the same-instance Ro-TS result under the designated protocol as the reference cost. Final comparisons always use the exact QAP objective rather than the learned score.

Assignment Score Matrices. An assignment score matrix $\Phi \in \mathbb{R}^{n\times n}$ assigns a score Φ_{ia} to placing facility i at location a, inducing the additive permutation score $S_\Phi(\pi) = \frac{1}{n}\sum_{i=1}^{n} \Phi_{i,\pi(i)}$. Although the QAP objective in Equation 1 is quadratic, the additive matrix provides a tractable proposal prior over complete assignments: Hungarian decoding is exact for S_Φ, Gumbel perturbations generate diverse permutations (Mena et al., 2018), and a swap changes only two terms of S_Φ.

From Static Scores to Search Basins. Let $\mathcal{A}_T(\pi;\xi)$ denote a fixed short-search procedure with budget T and random seed ξ, and define the basin value of an initial permutation as $B_T(\pi) = \mathbb{E}_\xi[C(\mathcal{A}_T(\pi;\xi))]$. Low $C(\pi)$ need not imply low $B_T(\pi)$: assignments with similar exact costs may lie in basins of very different depth, while a weaker initial assignment may improve more under the same backend. For a proposal portfolio $\mathcal{P}$, a more relevant downstream quantity is $Q_T(\mathcal{P}) = \mathbb{E}_\xi[\min_{\pi\in\mathcal{P}} C(\mathcal{A}_T(\pi;\xi))]$. This distinction motivates treating the learned score matrix as a proposal prior, generating a diverse portfolio of complete assignments, and delegating basin evaluation to exact short probes and the downstream search backend.

4 QAP-SCP

QAP-SCP learns an instance-conditioned proposal prior while leaving trajectory exploration and final acceptance to a replaceable QAP search backend. Given (F, D), we first construct a compatibility-aware structural representation that captures node-level interaction profiles and structural roles together with assignment-specific compatibility. These signals are integrated by a compatibility-guided scorer into an assignment matrix $\Phi_\theta \in \mathbb{R}^{n\times n}$, where Φ_{ia} scores assigning facility i to location a, and are decoded into a diverse set of complete permutations. Training uses search-refined assignment supervision together with exact exchange-geometry signals. A staged procedure first learns a structural assignment prior and then freezes it while a zero-initialized compatibility module learns an additive correction. Search is used only to construct supervision and select checkpoints, and is never differentiated through. At inference, exact objective values filter the proposal set and, when supported by the chosen backend, short probes allocate the remaining budget among promising trajectories. The model therefore does not replace the backend's move rule or search dynamics, but instead learns where search should begin. Objective evaluation and the returned incumbent remain governed by the exact QAP objective, allowing the learned proposal mechanism to interface with different search backends.

4.1 Compatibility-Aware Structural Representation

A QAP assignment depends not only on the properties of a facility and a location, but also on whether their structural roles are compatible under the pairwise flow–distance objective. We therefore organize assignment evidence at three levels: *interaction profiles*, *structural roles*, and *assignment compatibility*. For $X \in \{F, D\}$, the node representation is

$$x_i = \left[x_i^{\mathrm{prof}},\, x_i^{\mathrm{role}}\right],$$

while each candidate assignment (i, a) is associated with a pairwise representation r_{ia}. Row and column channels remain separate to preserve directional information in asymmetric QAPs.

Interaction Profile Encoding. For each node, we encode the local geometry of its incoming and outgoing interactions as

$$x_i^{\mathrm{prof}} = \psi_{\mathrm{prof}}(X_{i:}, X_{:i})\,,$$

where ψ_{prof} summarizes strength, dispersion, extrema, ℓ_1/ℓ_2 magnitudes, quantiles, and positive/nonzero occupancy. These descriptors characterize how interaction mass is distributed around a node and provide a common representational form for facility-flow and location-distance structure.

Structural Role Encoding. Local profiles do not capture a node's role relative to the rest of the instance. Let ρ_i collect six rank-based coordinates over row and column strength, variation, and absolute interaction magnitude. We additionally propagate interaction strength through the normalized matrix:

$$x_i^{\text{role}} = \left[\rho_i,\ \frac{(Xr)_i}{n},\ \frac{(X^\top c)_i}{n},\ \frac{(|X|r)_i}{n},\ \frac{(|X^\top|c)_i}{n}\right],$$

where r and c are the row- and column-strength vectors of X. These higher-order terms distinguish nodes with similar local profiles but different structural neighborhoods. Together, the profile and role encodings form a 39-dimensional node representation; continuous coordinates are standardized within each instance, with $\log(1+n)/\log(257)$ appended as a size coordinate.

Assignment Compatibility Encoding. Node representations describe the facility and location independently, whereas QAP decisions depend on their pairwise compatibility. We therefore define an eight-dimensional representation

$$r_{ia} = \left[r_{ia}^{\text{relax}},\ r_{ia}^{\text{spec}},\ r_{ia}^{\text{unc}},\ r_{ia}^{\text{align}}\right],$$

with four complementary channels.

(1) Relaxation compatibility. FAQ performs a fixed number of Frank–Wolfe updates over the Birkhoff polytope, and we retain the unrounded matrix P^{FAQ}. The standardized coordinate $r_{ia}^{\text{relax}} = (P_{ia}^{\text{FAQ}} - \mu_P)/(\sigma_P + \epsilon)$ provides a soft global assignment preference.

(2) Spectral compatibility. A spectral term measures alignment between facility and location structural embeddings, providing a complementary global correspondence signal.

(3) Assignment uncertainty. We encode row- and column-wise entropy together with top-two margins,

$$r_{ia}^{\text{unc}} = \left[H(P_{i:}^{\text{FAQ}}), H(P_{:a}^{\text{FAQ}}), M(P_{i:}^{\text{FAQ}}), M(P_{:a}^{\text{FAQ}})\right],$$

where $H(\cdot)$ denotes entropy and $M(\cdot)$ the largest-minus-second-largest margin, distinguishing confident from ambiguous relaxed correspondences.

(4) Structural-role compatibility. Finally, $r_{ia}^{\text{align}} = [\tilde{s}_i^F \tilde{c}_a^D,\ -|\rho_i^F - \rho_a^D|]$ compares facility strength and location centrality in both magnitude and relative rank.

The resulting representation separates *node-level structural evidence* from *assignment-specific compatibility evidence*. FAQ is used only as a deterministic structural encoder from (F, D): it accesses no reference solution, supplies no training labels, and does not directly determine the decoded proposal.

4.2 Compatibility-Guided Assignment Scoring

The structural representations in Section 4.1 provide two complementary views of each candidate assignment: node-level evidence describes the roles of the facility and location independently, while assignment-specific compatibility evidence characterizes how well the two sides align. We integrate these signals through a compatibility-guided scoring function that first forms a structural assignment prior and then refines it with a pair-specific compatibility correction.

Structural Assignment Prior. Let x_i^F and x_a^D denote the structural representations of facility i and location a, respectively. Separate shared encoders map the two sides into latent embeddings,

$$h_i^F = f_F(x_i^F), \qquad h_a^D = f_D(x_a^D).$$

For each candidate pair (i, a), we combine the two embeddings using their direct representations, elementwise interaction, and absolute discrepancy:

$$\Phi_{ia}^{\text{struct}} = g\big([h_i^F,\ h_a^D,\ h_i^F \odot h_a^D,\ |h_i^F - h_a^D|]\big).$$

The resulting score captures whether the facility and location exhibit compatible structural profiles and roles. Because each node embedding is computed once and reused across all candidate pairs, this construction scores all n^2 assignments without explicitly forming an n^2-node association graph.

Compatibility-Guided Correction. The structural prior compares independently encoded facility and location representations, but does not directly consume the assignment-specific evidence r_{ia} introduced in Section 4.1. We therefore map the compatibility representation to an additive correction,

$$\Phi^{\mathrm{comp}}_{ia} = u(r_{ia}), \qquad \Phi_{ia} = \Phi^{\mathrm{struct}}_{ia} + \Phi^{\mathrm{comp}}_{ia}.$$

This decomposition allows explicit compatibility signals, including relaxed assignment preference, spectral alignment, uncertainty, and structural-role agreement, to refine the structural prior without replacing it. The final matrix $\Phi \in \mathbb{R}^{n\times n}$ assigns a score to every facility–location pair. For a complete permutation π, we define the corresponding additive assignment score as

$$S_\Phi(\pi) = \frac{1}{n}\sum_{i=1}^{n} \Phi_{i,\pi(i)},$$

The scalar $S_\Phi(\pi)$ ranks complete permutations, with larger values indicating better predicted objective quality. The two components capture complementary evidence: Φ^{struct} models broad facility–location correspondence from reusable node representations, while Φ^{comp} provides assignment-specific corrections. This decomposition supports staged training, where a structural prior is learned first and then refined by a zero-initialized compatibility module. The additive form is a deliberate modeling choice that aligns the representation hierarchy with the staged training procedure while keeping the two evidence sources identifiable.

4.3 Staged Search-Aware Assignment Learning

The assignment scorer is trained using supervision derived from exact local improvement. Rather than learning only from raw candidate permutations, we use search to expose both a refined assignment target and the local exchange geometry around that target. These two signals supervise the same assignment score matrix, while a staged optimization procedure first establishes the structural assignment prior and then introduces assignment-specific compatibility as an additive refinement.

For each synthetic training instance, we construct a candidate set $\mathcal{C} = \{\pi_k\}_{k=1}^{m}$ containing random permutations, inexpensive hand-crafted constructions, perturbations of those constructions, and samples decoded from an earlier score matrix. Let $c_k = C(\pi_k)$. Starting from the lowest-cost candidate, deterministic best-improvement 2-swap descent produces a locally optimal assignment $\pi^\star$, which serves as the reference state for both supervision signals.

Search-Refined Assignment Supervision. The refined assignment $\pi^\star$ identifies facility–location correspondences that survive exact local improvement. We supervise the score matrix in both assignment directions, encouraging each facility to select its target location and each location to recover its assigned facility. Let $p^{\mathrm{row}}_{ia} = \mathrm{softmax}_a(\Phi_{ia})$ and $p^{\mathrm{col}}_{ia} = \mathrm{softmax}_i(\Phi_{ia})$, and write $\sigma^\star = (\pi^\star)^{-1}$. The assignment loss is

$$\mathcal{L}_{\mathrm{assign}} = -\frac{1}{2n}\left[\sum_i \log p^{\mathrm{row}}_{i,\pi^\star(i)} + \sum_a \log p^{\mathrm{col}}_{\sigma^\star(a),a}\right].$$

Unlike supervision from the best raw candidate, this target incorporates the improvements discovered by exact local search and therefore reflects a locally refined assignment structure.

Exchange-Geometry Supervision. A refined permutation alone does not distinguish mildly incorrect assignments from exchanges that strongly degrade the QAP objective. We therefore augment assignment supervision with the exact local geometry around $\pi^\star$. For each facility pair $i < j$, we compute the objective increase induced by swapping their assigned locations,

$$\Delta_{ij} = C(\mathrm{swap}_{ij}(\pi^\star)) - C(\pi^\star) \geq 0.$$

Let ρ denote the median positive swap cost within the instance and define $d_{ij} = \mathrm{clip}(\Delta_{ij}/\max(\rho,\epsilon), 0, 3)$. The assignment score matrix induces the corresponding margin

$$m_{ij} = \Phi_{i,\pi^\star(i)} + \Phi_{j,\pi^\star(j)} - \Phi_{i,\pi^\star(j)} - \Phi_{j,\pi^\star(i)}.$$

We regress this margin toward the normalized exact swap cost,

$$\mathcal{L}_{\mathrm{regret}} = \frac{1}{\binom{n}{2}}\sum_{i<j} \mathrm{SmoothL1}(m_{ij}, d_{ij}).$$

This signal teaches the scorer not only which assignment is preferred, but also how strongly nearby exchanges should be separated according to exact QAP cost.

The two supervision signals are combined on the shared assignment matrix:

$$\mathcal{L}_{\mathrm{QAP}} = \lambda_{\mathrm{assign}}\mathcal{L}_{\mathrm{assign}} + \lambda_{\mathrm{regret}}\mathcal{L}_{\mathrm{regret}}.$$

The assignment term provides permutation-level correspondence supervision, while the exchange term preserves local objective sensitivity around the search-refined solution.

Staged Compatibility Refinement. Training follows the decomposition introduced in Section 4.2. In Stage 1, we optimize only the structural scorer Φ^{struct} using $\mathcal{L}_{\mathrm{QAP}}$, establishing an assignment prior from the node-level structural representation. In Stage 2, we freeze all Stage-1 parameters and introduce the compatibility scorer Φ^{comp}. Its final affine layer is initialized to zero, so that the combined score $\Phi = \Phi^{\mathrm{struct}} + \Phi^{\mathrm{comp}}$ exactly recovers the Stage-1 scorer at the start of refinement. Stage 2 then optimizes only the compatibility module under the same supervision, allowing assignment-specific evidence to refine the learned structural prior without rewriting it from a shared random initialization. The Stage-1 checkpoint is selected using standalone decoded assignment quality, whereas the Stage-2 checkpoint is selected using the best terminal objective after a fixed short search probe. These search procedures are used only to construct supervision and select checkpoints; no gradient is propagated through the search process.

4.4 Multi-Start Proposal Portfolio and Backend Interface

At inference, the learned score matrix is converted into a diverse portfolio of complete assignments. We apply four facility/location relabelings, and for each relabeling generate one Hungarian decoding together with nine Gumbel-perturbed decodings, yielding 40 learned proposals after mapping back to the original indices. The perturbations diversify assignments around the learned prior, while independent relabelings provide additional variation without changing the underlying QAP instance.

To separate proposal generation from downstream search, we augment the learned portfolio with 40 uniformly random controls and evaluate all candidates using the exact QAP objective. Candidates are ranked by exact cost, and Hamming-diverse elites are greedily retained to balance solution quality and coverage. The selected permutations are then passed unchanged to a downstream heuristic $\mathcal{H}_T(\pi; \xi)$; the learned model does not alter its neighborhood, acceptance rule, memory mechanism, or perturbation strategy.

For the official-backend experiments, we instantiate $\mathcal{H}$ with BLS (Benlic & Hao, 2013). Four learned and four random elites receive equal short probes, after which the best trajectory from each source is allocated an equal share of the remaining budget. All proposal generation, filtering, probing, and refinement are counted within the reported wall-clock budget, and the network is never invoked inside the backend move loop. The returned solution is always the lowest-cost assignment observed under the exact QAP objective.

5 Experiments

5.1 Setup

We evaluate 256 instances per synthetic generator at $n \in \{20, 50, 100\}$, 20 Taixxeyy instances at each $n \in \{27, 45, 75, 125, 175\}$, and 135 QAPLIB instances. Ro-TS uses $5000n$ iterations, NGM 5,000 samples, and PLMA at most 200 test-time updates. BLS and QAP-SCP share 2/7/40-second synthetic and 2/5/20/30/40-second Taixxeyy tiers; QAPLIB uses 10/20/40 seconds for Tai, 10/20 seconds for Lipa/Sko/Tho/Wil, and 4/10 seconds for other families below/above $n = 100$.

All experiments use an AMD Ryzen 9 7945HX CPU and an NVIDIA V100 GPU with 32 GB memory. QAP-SCP and BLS share the official C++ backend and run serially on the CPU; Ro-TS adapts the official C code. QAP-SCP forms 40 learned and 40 random candidates, probes four elites per source, and counts all construction, filtering, probing, and refinement time. SAWT uses its released checkpoint and 10,000 decisions; NGM uses its released 5,000-sample protocol (134/135 QAPLIB instances); PLMA uses its checkpoint with $T = 200$ and seed 0. Other learning baselines follow their released protocols, including GPU-batched inference. Runtime is reported under each designated

Table 1: Synthetic results over 256 instances per distribution and size. Gap (%) is measured against the same-instance Ro-TS(5k) cost; time is mean runtime in seconds. BLS and QAP-SCP share 2-, 7-, and 40-second budgets at $n = 20, 50, 100$. Bold marks the lowest gap.

	QAP20		QAP50		QAP100	
Algorithm	Gap	Time	Gap	Time	Gap	Time
Geometrically Structured Datasets						
Ro-TS (5k)	**0.0000%**	0.44s	**0.0000%**	7.94s	0.0000%	71.29s
BLS	**0.0000%**	1.94s	0.0182%	6.93s	0.0251%	39.95s
SAWT (10k)	0.7545%	1.98s	1.2106%	2.05s	7.1679%	1.85s
NGM (5k)	11.9318%	1.66s	11.2704%	2.46s	9.2446%	12.10s
PLMA (T=200)	0.0923%	5.94s	0.0334%	14.88s	**-0.0036%**	39.08s
QAP-SCP	**0.0000%**	1.94s	0.0048%	6.95s	0.0157%	39.95s
Uniformly Random Datasets						
Ro-TS (5k)	0.0000%	0.51s	0.0000%	9.25s	**0.0000%**	89.49s
BLS	**-0.0067%**	1.95s	-0.1089%	6.93s	0.0280%	39.94s
NGM (5k)	13.7153%	1.66s	13.2815%	2.35s	10.7417%	12.15s
PLMA (T=200)	0.1674%	5.75s	0.0725%	14.39s	2.0315%	38.99s
QAP-SCP	**-0.0067%**	1.95s	**-0.1218%**	6.94s	0.0195%	39.94s

Table 2: Taixxeyy results over 20 instances per size. Entries report mean best-known gap (%) / runtime (s). PLMA ($T = 200$) uses seed 0 with amortized GPU batch time; Ro-TS uses $5000n$ iterations; SAWT 10,000 decisions; NGM 5,000 Gumbel–Sinkhorn samples; BLS and QAP-SCP use budgets of 2, 5, 20, 30, and 40 seconds.

n	Ro-TS (5k) *Gap / Time*	BLS *Gap / Time*	SAWT (10k) *Gap / Time*	NGM (5k) *Gap / Time*	PLMA (T=200) *Gap / Time*	QAP-SCP *Gap / Time*
27	47.2997% / 1.08s	0.2840% / 1.94s	1561.2729% / 10.07s	1686.2059% / 2.19s	**0.0000% / 0.32s**	0.2251% / 1.94s
45	111.1121% / 4.94s	234.2761% / 4.93s	4220.8822% / 10.39s	4047.6674% / 2.52s	56.7847% / 13.77s	**8.1679% / 4.94s**
75	116.0801% / 22.68s	163.2698% / 19.93s	5188.5247% / 10.86s	5033.5200% / 5.14s	**3.0122% / 23.89s**	13.2024% / 19.93s
125	98.8155% / 113.93s	131.6034% / 29.94s	9397.8665% / 11.65s	9537.5292% / 22.67s	69.7689% / 59.46s	**23.4300% / 29.94s**
175	67.2210% / 305.07s	211.4668% / 39.95s	12403.0203% / 12.46s	12296.4947% / 75.88s	94.0069% / 112.05s	**25.9735% / 39.95s**
All	88.1057% / 89.54s	148.1800% / 19.34s	6554.3133% / 11.09s	6520.2835% / 21.68s	44.7145% / 41.90s	**14.1998% / 19.34s**

protocol; BiQAP is omitted because its published QAPLIB metrics are not mean best-known gap (Shi et al., 2025). QAPLIB and Taixxeyy use best-known gaps, while synthetic gaps use same-instance Ro-TS cost; we also report paired advantage, W/T/L, and instance-bootstrap 95% CIs.

5.2 Main Results

Table 1 reports the per-size gaps and runtimes for both synthetic distributions.Synthetic $n = 20$ is saturated. At larger sizes, QAP-SCP stays within 0.016% of the Ro-TS reference on geometric data and reaches -0.1218% on uniform $n = 50$ versus -0.1089% for BLS. PLMA is strongest on geometric $n = 100$ but degrades on uniform $n = 100$; direct SAWT and NGM remain farther from the reference. Across all 1,536 paired instances, QAP-SCP improves over matched-budget BLS by 0.0073% (95% CI $[0.0015, 0.0132]\%$; W/T/L $= 507/612/417$).

Table 2 reports the per-size gaps and runtimes on Taixxeyy. QAP-SCP reduces single-start BLS mean gap from 148.18% to 14.20% (paired advantage 30.61%, 95% CI $[23.44, 37.72]\%$, W/T/L $= 57/18/25$). This supports the complete multi-candidate racing system, not attribution of the full gain to learning alone. PLMA reaches 44.71% ; Ro-TS reaches 88.11% and exhibits severe single-start tails. Frozen SAWT with zero-shot MDS and direct NGM do not transfer to this non-geometric stress test.

Table 3 reports the family-wise gaps and runtimes on QAPLIB. QAP-SCP, BLS, and Ro-TS obtain mean gaps of 0.0611%, 0.0847%, and 0.1089%. The paired advantage over BLS is $+0.0233\%$ with CI $[-0.0024, 0.0540]\%$ and W/T/L $= 17/110/8$; hence this saturated panel supports low gap, not a stable learned advantage. SAWT and NGM transfer poorly (35.39% and 50.13% over 134 completed instances), while PLMA reaches 0.2304%. NGM rows are direct released-model inference rather than a shared-backend attribution experiment.

Table 3: QAPLIB results grouped by instance family. Entries are mean reference gap (%) / mean runtime (s). Ro-TS uses $5000n$ iterations, SAWT uses 10,000 decisions, NGM uses 5,000 samples, and PLMA (T=200) uses its fixed update budget. NGM covers 134/135 instances; † marks its affected Tai and overall means.

Family (size range)	Ro-TS (5k) *Gap / Time*	BLS *Gap / Time*	SAWT (10k) *Gap / Time*	NGM (5k) *Gap / Time*	PLMA (T=200) *Gap / Time*	QAP-SCP *Gap / Time*
Bur (26)	**0.0000% / 0.88s**	**0.0000% / 3.92s**	3.9120% / 24.81s	3.2691% / 1.65s	**0.0000% / 0.33s**	**0.0000% / 3.94s**
Chr (12–25)	0.5685% / 0.33s	0.0507% / 3.92s	124.3679% / 31.11s	252.8167% / 1.48s	**0.0000% / 2.72s**	**0.0000% / 3.94s**
Els (19)	**0.0000% / 0.36s**	**0.0000% / 3.93s**	47.3708% / 144.53s	53.7791% / 1.97s	**0.0000% / 2.22s**	**0.0000% / 3.94s**
Esc (16–128)	**0.0000% / 7.37s**	**0.0000% / 4.26s**	60.9052% / 30.10s	61.7006% / 3.04s	**0.0000% / 1.87s**	**0.0000% / 4.27s**
Had (12–20)	**0.0000% / 0.24s**	**0.0000% / 3.93s**	4.3957% / 37.14s	3.7746% / 1.67s	**0.0000% / 1.95s**	**0.0000% / 3.94s**
Kra (30–32)	**0.0000% / 1.45s**	**0.0000% / 3.93s**	34.2057% / 23.50s	32.7437% / 1.79s	0.4755% / 6.31s	**0.0000% / 3.94s**
Lipa (20–90)	**0.0000% / 12.82s**	0.0615% / 9.93s	1.3920% / 48.21s	14.4711% / 4.38s	0.2288% / 23.10s	0.0284% / 9.93s
Nug (12–30)	**0.0000% / 0.52s**	**0.0000% / 3.92s**	18.5976% / 68.32s	16.2569% / 2.25s	**0.0000% / 2.55s**	**0.0000% / 3.94s**
Rou (12–20)	**0.0000% / 0.24s**	**0.0000% / 3.92s**	12.5658% / 20.11s	12.8071% / 1.92s	**0.0000% / 2.83s**	**0.0000% / 3.94s**
Scr (12–20)	**0.0000% / 0.23s**	**0.0000% / 3.93s**	28.0012% / 20.11s	34.6297% / 1.70s	**0.0000% / 2.83s**	**0.0000% / 3.94s**
Sko (42–100)	0.0465% / 32.30s	0.0753% / 9.93s	15.6046% / 86.18s	15.7957% / 11.10s	0.4169% / 52.11s	**0.0317% / 9.93s**
Ste (36)	**0.0000% / 2.37s**	**0.0000% / 3.93s**	117.4086% / 60.98s	114.6438% / 2.53s	**0.0000% / 13.27s**	**0.0000% / 3.94s**
Tai (10–256)	**0.2145% / 45.73s**	0.2831% / 11.36s	23.7699% / 52.76s	22.8057% / 7.13s†	0.6778% / 28.59s	0.2561% / 11.36s
Tho (30–150)	**0.0339% / 59.72s**	0.2568% / 13.27s	24.9514% / 56.14s	23.3541% / 32.51s	0.4064% / 42.24s	0.0554% / 13.27s
Wil (50–100)	**0.0115% / 28.35s**	0.0322% / 9.94s	8.9986% / 31.80s	8.4911% / 12.21s	0.2008% / 35.51s	0.0253% / 9.93s
All (10–256)	0.1089% / 17.10s	0.0847% / 7.10s	35.3862% / 48.19s	50.1258% / 5.11s†	0.2304% / 16.64s	**0.0611% / 7.11s**

Table 4: Standalone QAPLIB construction by size, without local search. Entries report mean reference gap (%) / mean construction time (s); lower is better. PLMA-CG, derived from the PLMA cross-graph scorer (Pan et al., 2026), and QAP-SCP use matched best-of-80 decoding on CPU. SAWT uses its released 10,000-decision GPU-batched protocol; NGM uses 5,000 Gumbel–Sinkhorn samples under its released GPU protocol. † marks incomplete NGM coverage.

Size (instances)	SAWT (10k) *Gap / Time*	NGM (5k) *Gap / Time*	PLMA-CG *Gap / Time*	QAP-SCP *Gap / Time*
$n \leq 20$ (51)	40.2205% / 31.24s	76.7011% / 1.72s	60.3481% / 0.0033s	**46.4743%** / 0.0055s
$21 \leq n \leq 49$ (47)	42.3399% / 51.44s	39.5687% / 1.92s	56.0627% / 0.0049s	**21.8812%** / 0.0106s
$50 \leq n \leq 99$ (24)	13.1980% / 70.66s	20.5857% / 5.16s	22.0947% / 0.0120s	**9.7993%** / 0.0382s
$n \geq 100$ (13)	32.2438% / 61.51s	37.6094%† / 31.97s	33.6228% / 0.0328s	**10.6576%** / 0.1448s
All	35.3862% / 48.19s (135)	50.1258%† / 5.11s (134)	49.4820% / 0.0083s (135)	**27.9432%** / 0.0265s (135)

5.3 Ablations and Attribution

We isolate standalone construction, representation and loss terms, and start sources. Static experiments use 80 decoded permutations without local search.

Standalone Construction. Table 4 compares standalone construction quality on QAPLIB (no local search) under matched best-of-80 decoding, isolating QAP-SCP's construction gap against SAWT, NGM, and PLMA-CG.PLMA-CG is our local reproduction of PLMA's cross-graph scorer, not the released checkpoint or full solver (Pan et al., 2026). MCMC and refinement are disabled, and it shares best-of-80 decoding with QAP-SCP. SAWT uses its released 10k-decision protocol. Family-wise results appear in Appendix B. Under matched best-of-80 decoding, QAP-SCP improves on PLMA-CG in every size group and reduces overall gap from 49.48% to 27.94%. SAWT and NGM are shown under their released 10k- and 5k-sample protocols, respectively; NGM covers 134/135 instances. The reported times follow the corresponding execution protocols.

Representation and Loss Ablation. Table 5 ablates QAP-SCP's components to measure each part's contribution. The encoding row masks the Assignment Compatibility Encoding tensor in a frozen model and is therefore a reliance diagnostic; loss rows retrain the stated branch. The strict no-Search-Refined Assignment Supervision arm removes assignment supervision from both stages while retaining their fixed Exchange-Geometry Supervision weights.

The model relies on the Assignment Encoding input, and training its branch gives the largest gain. The isolated Exchange-Geometry Supervision effect is smaller and inconclusive on QAPLIB; removing the Search-Refined Assignment Supervision degrades both panels with CIs above zero.

Table 5: Best-of-80 construction ablations without local search. Values are paired objective advantages (%) of the complete model over the indicated ablation, with 95% instance-bootstrap CI; positive favors the complete component. These rows use 135 QAPLIB instances. The no-Compatibility Branch row follows the strict staged protocol.

Component	Synthetic (75)	QAPLIB
no-Search-Refined Assignment Compatibility Encoding	$+1.960\%$ $[+1.224, +2.739]\%$	$+7.926\%$ $[+5.713, +10.371]\%$
no-Assignment Supervision	$+2.957\%$ $[+2.054, +3.977]\%$	$+1.553\%$ $[+0.753, +2.494]\%$
no-Exchange-Geometry Supervision	$+0.322\%$ $[+0.066, +0.579]\%$	$+0.641\%$ $[-0.071, +1.515]\%$
no-Compatibility-Guided Correction	$+16.742\%$ $[+13.978, +19.686]\%$	$+13.622\%$ $[+9.893, +17.897]\%$

Start Ablation. Table 6 compares single and multi-start sources.Under the current tiered budgets, a single learned start reduces the mean Taixxeyy gap from 148.1800% to 74.2848%. The paired mean gap reduction (random minus learned) is 73.8952 percentage points, with a 95% instance-bootstrap CI of $[38.4342, 108.8256]$ and W/T/L of $54/22/24$. On QAPLIB, the corresponding gaps are 0.0847% and 0.0594%; the paired reduction is 0.0253 percentage points, CI $[-0.0009, 0.0591]$, with W/T/L $12/111/12$.

Table 6: Start-source ablation on Taixxeyy and QAPLIB. Entries are mean best-known gap (%); lower is better. Parenthesized values are candidate counts. Column headers give the current main-table tiers: 2/5/20/30/40 seconds for Taixxeyy and the 4–40-second QAPLIB tiers in Section 5.1. All entries use the same official C++ BLS backend and the same main-table budgets. Candidate counts, probe allocation, and random seeds are fixed before each comparison and held constant across the compared source methods.

Start source and backend	Taixxeyy	QAPLIB
Random (1) $\rightarrow$ BLS	148.1800%	0.0847%
QAP-SCP (1) $\rightarrow$ BLS	74.2848%	**0.0594%**
Random (40+40) $\rightarrow$ BLS	22.1024%	0.1318%
PLMA-CG (40) + Random (40) $\rightarrow$ BLS	20.1884%	0.0985%
SAWT (40) + Random (40) $\rightarrow$ BLS	20.7249%	0.1200%
QAP-SCP (40) + Random (40) $\rightarrow$ BLS	**14.1998%**	0.0611%

For multiple starts, replacing one random source with QAP-SCP changes the reported mean gaps from 22.1024% to 14.1998% on Taixxeyy and from 0.1318% to 0.0611% on QAPLIB. In the separate fixed-budget, matched source-replacement experiment, the mean relative objective advantages are 3.2736% (CI $[1.4533, 5.6835]\%$; W/T/L $47/31/22$) and 0.0490% (CI $[0.0102, 0.0972]\%$; W/T/L $21/107/7$), respectively. These latter intervals describe the fixed-budget experiment, not the differences between the tiered-budget means in Table 6.

Cross-dataset Interpretation. QAPLIB is near a ceiling: single-start BLS reaches 0.0847% and ties the full system on 110/135 instances, while random multi-start does not improve it. Taixxeyy is basin-sensitive: random multi-start reduces mean gap from 148.18% to 22.10%, and learned-plus-random proposals to 14.20%. Thus coverage and probe allocation explain most of the reduction, with learning adding a proposal signal. No full-system gap exceeds 50%, but means of $13.20\%/23.43\%/25.97\%$ at $n = 75/125/175$ show residual error. The matched protocol isolates the effect of changing source composition under the same search backend and allocation rules.

6 Conclusion

We presented QAP-SCP, a compatibility-aware framework that learns an assignment proposal prior for QAP while leaving trajectory exploration and final acceptance to a replaceable search backend. The method combines a compatibility-aware structural representation, compatibility-guided assignment scoring, staged search-aware assignment learning, and a multi-start proposal portfolio. Experiments with official search backends show that QAP-SCP substantially reduces severe long-tail failures on Taixxeyy and improves over Ro-TS on QAPLIB, while remaining statistically tied with BLS on QAPLIB and showing no stable advantage across every synthetic setting. Ablations further show that assignment-specific compatibility improves standalone construction, while its downstream benefit depends on the search landscape. Overall, the results support treating the learned score matrix as a proposal prior and delegating basin evaluation to exact inference-time probes and downstream search.

AI USE STATEMENT

Generative AI tools were used to assist with code development, experiment bookkeeping, literature discovery, and drafting. The authors reviewed the AI-assisted material against source code, logged experimental artifacts, and primary references. The authors take responsibility for the final content, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The supplementary material will include feature definitions, model and loss hyperparameters, dataset generation seeds, per-instance results, objective recomputation checks, and executable evaluation scripts. The main text reports all construction and search time inside the stated wall-clock budgets. An anonymized repository containing the minimal runnable code, configurations, and evaluation scripts for reproducing the three main tables is available at `https://anonymous.4open.science/r/qap-fl-C34D`.

REFERENCES

Irwan Bello, Hieu Pham, Quoc V. Le, Mohammad Norouzi, and Samy Bengio. Neural combinatorial optimization with reinforcement learning. *arXiv preprint arXiv:1611.09940*, 2016.

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.

Una Benlic and Jin-Kao Hao. Breakout local search for the quadratic assignment problem. *Applied Mathematics and Computation*, 219(9):4800–4815, 2013. doi: 10.1016/j.amc.2012.10.106.

Rainer E. Burkard, Stefan E. Karisch, and Franz Rendl. QAPLIB—a quadratic assignment problem library. *Journal of Global Optimization*, 10(4):391–403, 1997. doi: 10.1023/A:1008293323270.

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

Xinyun Chen and Yuandong Tian. Learning to perform local rewriting for combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Minsu Cho, Jungmin Lee, and Kyoung Mu Lee. Reweighted random walks for graph matching. In *European Conference on Computer Vision*, pp. 492–505, 2010.

Jeffrey Christiansen and Kate Smith-Miles. Instance space analysis for the quadratic assignment problem. *arXiv preprint arXiv:2506.20172*, 2025. URL `https://arxiv.org/abs/2506.20172`.

Zvi Drezner. A new genetic algorithm for the quadratic assignment problem. *INFORMS Journal on Computing*, 15(3):320–330, 2003. doi: 10.1287/ijoc.15.3.320.16076.

Zvi Drezner, Peter M. Hahn, and Éric D. Taillard. Recent advances for the quadratic assignment problem with special emphasis on instances that are difficult for meta-heuristic methods. *Annals of Operations Research*, 139(1):65–94, 2005. doi: 10.1007/s10479-005-3444-z.

André Hottung and Kevin Tierney. Neural large neighborhood search for routing problems. *Artificial Intelligence*, 313:103786, 2022. doi: 10.1016/j.artint.2022.103786.

André Hottung, Yeong-Dae Kwon, and Kevin Tierney. Efficient active search for combinatorial optimization problems. In *International Conference on Learning Representations*, 2022.

Minsu Kim, Junyoung Park, and Jinkyoo Park. Sym-NCO: Leveraging symmetricity for neural combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

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

Tjalling C. Koopmans and Martin Beckmann. Assignment problems and the location of economic activities. *Econometrica*, 25(1):53–76, 1957. doi: 10.2307/1907742.

Harold W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1–2):83–97, 1955. doi: 10.1002/nav.3800020109.

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

Marius Leordeanu, Martial Hebert, and Rahul Sukthankar. An integer projected fixed point method for graph matching and map inference. In *Advances in Neural Information Processing Systems*, volume 22, 2009.

Eliane Maria Loiola, Nair Maria Maia de Abreu, Paulo Oswaldo Boaventura-Netto, Peter Hahn, and Tania Querido. A survey for the quadratic assignment problem. *European Journal of Operational Research*, 176(2):657–690, 2007. doi: 10.1016/j.ejor.2005.09.032.

Gonzalo Mena, David Belanger, Scott Linderman, and Jasper Snoek. Learning latent permutations with gumbel-sinkhorn networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Byt3oJ-0W.

Alex Nowak, Soledad Villar, Afonso S. Bandeira, and Joan Bruna. Revised note on learning quadratic assignment with graph neural networks. In *IEEE Data Science Workshop*, pp. 229–233, 2018. doi: 10.1109/DSW.2018.8439919.

Yicheng Pan, Ruisong Zhou, Haijun Zou, Tianyou Li, and Zaiwen Wen. Learning to solve the quadratic assignment problem with warm-started MCMC finetuning. *arXiv preprint arXiv:2604.20109*, 2026. URL https://arxiv.org/abs/2604.20109.

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

Liangliang Shi, Haoran Zhang, Shuheng Shen, Changhua Meng, Weiqiang Wang, and Junchi Yan. BiQAP: Neural bi-level optimization-based framework for solving quadratic assignment problems. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 2550–2561, 2025. doi: 10.1145/3711896.3736859.

Éric Taillard. Robust taboo search for the quadratic assignment problem. *Parallel Computing*, 17 (4–5):443–455, 1991. doi: 10.1016/S0167-8191(05)80147-4.

Zhentao Tan and Yadong Mu. Learning solution-aware transformers for efficiently solving quadratic assignment problem. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 47627–47648. PMLR, 2024. URL https://proceedings.mlr.press/v235/tan24d.html.

Joshua T. Vogelstein, John M. Conroy, Vince Lyzinski, Louis J. Podrazik, Steven G. Kratzer, Eric T. Harley, Donniell E. Fishkind, R. Jacob Vogelstein, and Carey E. Priebe. Fast approximate quadratic programming for graph matching. *PLOS ONE*, 10(4):e0121002, 2015. doi: 10.1371/journal.pone.0121002.

Runzhong Wang, Zhigang Hua, Gan Liu, Jiayi Zhang, Junchi Yan, Feng Qi, Shuang Yang, Jun Zhou, and Xiaokang Yang. A bi-level framework for learning to solve combinatorial optimization on graphs. In *Advances in Neural Information Processing Systems*, volume 34, pp. 21453–21466, 2021.

Runzhong Wang, Junchi Yan, and Xiaokang Yang. Neural graph matching network: Learning lawler's quadratic assignment problem with extension to hypergraph and multiple-graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):5261–5279, 2022. doi: 10.1109/TPAMI.2021.3078053.

Yaoxin Wu, Wen Song, Zhiguang Cao, Jie Zhang, and Andrew Lim. Learning improvement heuristics for solving routing problems. *IEEE Transactions on Neural Networks and Learning Systems*, 33 (9):5057–5069, 2022. doi: 10.1109/TNNLS.2021.3068828.

A FEATURE GROUPS

Tables 7 and 8 give the complete feature specification used by the released model. Each statistic is computed independently for the facility-flow matrix and the location-distance matrix; the same feature map and normalization are used on both sides.

Table 7: Complete 39-dimensional node signature. Here X denotes the max-absolute-normalized flow or distance matrix and X_i its ith row. Continuous coordinates (all but ranks and size) are standardized within an instance.

Group	Dim.	Coordinates
Row summaries	14	$\sum X_i$, mean, standard deviation, minimum, maximum, ℓ_1 norm, ℓ_2 norm, quantiles at $0.10/0.25/0.50/0.75/0.90$, positive-entry fraction, and nonzero-entry fraction.
Column summaries	14	The same 14 statistics applied to column i of X (equivalently, row i of $X^\top$).
Relative-order summaries	6	Average ranks of row sum, row standard deviation, row ℓ_1 norm, column sum, column standard deviation, and column ℓ_1 norm.
Two-hop structural context	4	$(Xr)_i/n$, $(X^\top c)_i/n$, $(\lvert X\rvert r)_i/n$, and $(\lvert X^\top\rvert c)_i/n$, where r and c are row and column ℓ_1-strength vectors.
Size context	1	$\log(1+n)/\log(257)$.
Total	39	Per facility or location node.

Table 8: Complete eight-dimensional facility–location compatibility tensor for candidate pair (i, a). FAQ is run for a fixed number of Frank–Wolfe updates and is used only as a deterministic feature transform.

Index	Feature	Definition and role
1	FAQ assignment	Standardized soft-assignment coordinate P^{FAQ}_{ia}.
2	Spectral compatibility	Negative standardized distance between aligned facility and location spectral embeddings.
3	FAQ row entropy	Standardized entropy of facility row $P^{\mathrm{FAQ}}_{i:}$, broadcast over locations.
4	FAQ column entropy	Standardized entropy of location column $P^{\mathrm{FAQ}}_{:a}$, broadcast over facilities.
5	FAQ row margin	Standardized top-one minus top-two probability margin of $P^{\mathrm{FAQ}}_{i:}$.
6	FAQ column margin	Standardized top-one minus top-two probability margin of $P^{\mathrm{FAQ}}_{:a}$.
7	Strength–centrality product	Product of standardized facility total absolute flow and location centrality (negative total absolute distance).
8	Strength–centrality rank match	Negative absolute difference between the average ranks of facility strength and location centrality.

B FAMILY-WISE STANDALONE QAPLIB RESULTS

Table 9: Standalone construction on QAPLIB without an external local-search backend, grouped by family and observed size range. Entries report mean best-known gap (%); lower is better. PLMA-CG, derived from the PLMA cross-graph scorer (Pan et al., 2026), and QAP-SCP use identical best-of-80 assignment decoding, shared perturbations, and two seeds. SAWT uses 10,000 improvement decisions; NGM uses its released inference protocol. † marks family rows with incomplete NGM coverage.

Family (size range)	SAWT (10k) *Gap*	NGM (5k) *Gap*	PLMA-CG *Gap*	QAP-SCP *Gap*
Bur (26)	3.9120%	3.2691%	4.8590%	**3.2526%**
Chr (12–25)	**124.3679%**	252.8167%	181.3867%	151.2715%
Els (19)	**47.3708%**	53.7791%	56.5077%	66.8027%
Esc (16–128)	60.9052%	61.7006%†	100.2060%	**16.4577%**
Had (12–20)	4.3957%	**3.7746%**	5.1015%	4.9443%
Kra (30–32)	34.2057%	32.7437%	30.4233%	**18.9883%**
Lipa (20–90)	**1.3920%**	14.4711%	14.8842%	2.4632%
Nug (12–30)	18.5976%	16.2569%	18.0959%	**11.8587%**
Rou (12–20)	**12.5658%**	12.8071%	14.3517%	13.8386%
Scr (12–20)	**28.0012%**	34.6297%	36.1510%	28.1450%
Sko (42–100)	15.6046%	15.7957%	14.9914%	**8.7990%**
Ste (36)	117.4086%	114.6438%	114.4551%	**31.3299%**
Tai (10–256)	23.7699%	22.8057%†	30.0030%	**20.5301%**
Tho (30–150)	24.9514%	23.3541%†	22.8177%	**15.9154%**
Wil (50–100)	8.9986%	8.4911%	8.4536%	**4.7882%**
All	35.3862% (135)	50.1258%† (134)	49.4820% (135)	**27.9432%** (135)

C WALL-CLOCK-MATCHED MAIN TABLES

The main text follows each baseline's original stopping protocol. Before that comparison was adopted, we also evaluated all available implementations under fixed wall-clock limits. Tables 10, 11, and 12 preserve those results. Methods share instances and nominal time limits; neural baselines run on the V100 while BLS, Ro-TS, and QAP-SCP run on the Ryzen 9 7945HX. For PLMA-local, a dash marks a run that reached the deadline before completing one test-time-update epoch. Such a fallback is not reported as an algorithm result.

Table 10: Fixed-20-second Taixxeyy results over 20 instances per size. Entries are mean best-known gap (%). † denotes a partial 80/100-instance mean.

n	Ro-TS	BLS	SAWT	NGM	PLMA-local	QAP-SCP
27	23.9478%	**0.0000%**	1951.7325%	1383.9262%	**0.0000%**	**0.0000%**
45	154.2924%	213.0631%	4711.3649%	3856.9604%	56.9008%	**1.3584%**
75	101.0536%	163.2370%	5459.4890%	4793.3027%	23.3858%	**12.5064%**
125	74.3801%	134.4608%	9891.0395%	9247.8664%	141.2595%	**25.5005%**
175	221.1227%	213.9696%	12825.6213%	–	–	**27.4238%**
All	114.9593%	144.9461%	6967.8494%	4820.5139%†	55.3865%†	**13.3578%**

Table 11: Fixed-budget QAPLIB results grouped by family. Entries are mean best-known gap (%). Budgets are 4 seconds for $n \leq 100$, 10 seconds for $101 \leq n \leq 175$, and 60 seconds above 175. NGM covers 131/135 instances; † marks affected partial means.

Family (size range)	Ro-TS	BLS	SAWT	NGM	PLMA-local	QAP-SCP
Bur (26)	**0.0000%**	**0.0000%**	4.9424%	3.4241%	**0.0000%**	**0.0000%**
Chr (12–25)	0.1732%	0.0507%	169.5122%	247.8767%	0.6031%	**0.0000%**
Els (19)	**0.0000%**	**0.0000%**	47.3708%	51.7832%	4.2090%	**0.0000%**
Esc (16–128)	0.6944%	**0.0000%**	59.4177%	43.3714%†	**0.0000%**	**0.0000%**
Had (12–20)	**0.0000%**	**0.0000%**	7.1420%	4.2894%	**0.0000%**	**0.0000%**
Kra (30–32)	**0.0000%**	**0.0000%**	40.4906%	32.5189%	0.8995%	**0.0000%**
Lipa (20–90)	0.1003%	**0.0615%**	1.5285%	14.5881%	3.6047%	0.0635%
Nug (12–30)	**0.0000%**	**0.0000%**	23.7882%	16.0018%	0.0044%	**0.0000%**
Rou (12–20)	**0.0000%**	**0.0000%**	15.1093%	12.8992%	0.0593%	**0.0000%**
Scr (12–20)	**0.0000%**	**0.0000%**	36.4115%	26.8373%	**0.0000%**	**0.0000%**
Sko (42–100)	0.1857%	0.0976%	17.6972%	15.6881%	1.8267%	**0.0805%**
Ste (36)	**0.0000%**	**0.0000%**	122.9812%	106.6608%	0.5113%	**0.0000%**
Tai (10–256)	0.4731%	**0.3099%**	29.3917%	22.6587%†	1.2322%	0.3130%
Tho (30–150)	0.4052%	0.2931%	28.0915%	26.2352%†	0.7845%	**0.0445%**
Wil (50–100)	0.0692%	**0.0322%**	10.5631%	8.8081%	0.8190%	0.0516%
All (10–256)	0.2485%	0.0932%	42.5926%	45.7893%†	1.0151%	**0.0820%**

Table 12: Fixed-budget synthetic results over 256 instances per distribution and size. Entries are mean gap (%) relative to same-instance Ro-TS. Budgets are 2, 5, and 20 seconds for $n = 20, 50, 100$. PLMA-local is unavailable because its batched runs completed zero test-time-update epochs before each deadline.

Distribution	n	Ro-TS	BLS	SAWT	NGM	PLMA-local	QAP-SCP
Geometric	20	**0.0000%**	**0.0000%**	2.0921%	12.6538%	–	**0.0000%**
Geometric	50	**0.0000%**	0.0202%	2.0668%	11.2496%	–	0.0037%
Geometric	100	**0.0000%**	0.0059%	7.5717%	9.0444%	–	0.0004%
Uniform	20	**0.0000%**	**0.0000%**	–	14.2357%	–	**0.0000%**
Uniform	50	0.0000%	-0.1166%	–	13.2749%	–	**-0.1470%**
Uniform	100	0.0000%	0.0082%	–	10.5740%	–	**-0.0075%**