# Returning the First Step to the Policy: Repairing the Multistart Bias of Neural Solvers on Selective Routing

**Anonymous authors**
**Paper under double-blind review**

## Abstract

Multistart decoding is central to state-of-the-art constructive neural routing solvers. On selective routing problems such as the Orienteering Problem (OP) and the Prize-Collecting TSP (PCTSP), a solver must decide not only how to order visits but which nodes are worth visiting at all. We identify a bias of POMO-style multistart decoding in this setting: the first node of every rollout is fixed outside the policy, assigned zero log-probability, and therefore excluded from learning. We repair it with selection-aware starts (SAS), a minimal change: one 128-parameter linear head that selects rollout starts and restores step-zero credit, leaving the POMO backbone and shared-baseline training unchanged. On OP50, SAS cuts the single-seed one-shot gap from 12.4% to 3.9% with full-budget performance unchanged; the repair transfers to PCTSP50 at the mechanism level, although the one-shot gap remains larger than on OP50. We then characterize in advance when the bias matters. Across five OP50 budgets, a model-free *start-stakes* measure, computed by classical local search as the share of achievable value lost when the start is forced to a uniformly random feasible node, tracks the realized one-shot benefit of the repair for this model family trained on the target distribution ($R^2 = 0.96$). Part of that correlation is structural; the empirical finding is the conversion rate: the realized benefit is a near-constant 0.72 to 0.75 of the available start value at every budget, a ratio nothing in the training objective guarantees, and it transfers unchanged to PCTSP. Within the measured range, the near-zero-benefit end of the spectrum is reached only through problem structure. Under the evaluated shifts the two factors separate: the measured start value persists, but the trained head converts less of it: the advantage halves on OP100 and reverses on OPLib, where every configuration decodes at gaps between 49% and 64%.

## 1 Introduction

Constructive neural solvers have become strong on routing problems where every node must be visited. On the traveling salesman problem, methods built on the multistart POMO recipe reach within a fraction of a percent of optimal. Yet the moment the problem changes from how to visit every node to which nodes are worth visiting at all, the same methods fall behind. On orienteering and prize-collecting routing, where a budget forces the solver to select a subset, the reported gaps of POMO-style models are several times larger than on TSP. This gap has been documented repeatedly, reported as a performance deficit, and left without a mechanism-level explanation; the failure that survives a careful protocol is narrower than those reports suggest, confined to low decoding budgets while a properly trained POMO is nearly at the classical reference given hundreds of rollouts per instance.

We identify the mechanism: the gap traces to an architectural bias at a specific point in the decoding path. The multistart paradigm behind these solvers hardcodes the assumption that every node is equally worth committing to first. On problems where every node must be visited, this is a sound symmetry: any node can open a tour. On selective routing, where whether to visit a node is itself the decision, the same assumption excludes the very choice that most needs to be learned. The first step is injected as a fixed rule and its

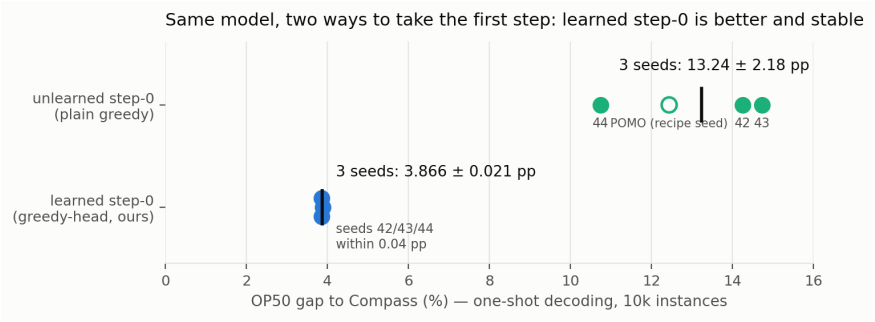


Figure 1: SAS is POMO with our learned start head (Section 2.3); the figure shows cross-seed variance of one-shot decoding for the same three SAS models. Bypassing the head (unlearned step zero, top row) scatters across seeds; decoding through the trained head (bottom row) is more stable (cross-seed spread 2.18 versus 0.021 gap points over three seeds).

log-probability is hardcoded to zero, so the decision of where to commit first never enters the policy gradient. Figure 1 shows what this costs: the same model, decoded from a learned first step, is both better and more stable than from a free, untrained first step, the only one-shot decode plain POMO has.

The repair addresses this mechanism directly. We leave the multistart backbone and its shared baseline untouched and change only who chooses the first step: a small head scores each node and selects the starts, and the log-probability that was hardcoded to zero is returned to the policy, restoring exactly the term POMO's own objective drops at step zero (exact in the sampled-start regime; the one shaping quantity, a temperature schedule, is a design constant fixed before training). The change is a single linear head, and it closes most of the gap where it is largest: at low decoding budgets, where a solver cannot afford to roll out from every node. On orienteering, it cuts the one-shot gap from 12.4% (a single-seed figure; the measured cross-seed band of free-first-step one-shot decoding is 10.7% to 14.7%) to 3.9%, while leaving full-budget performance unchanged. On prize-collecting routing the same repair transfers at the mechanism level, verified with the same control and the same geometric instrument, though its one-shot decode does not reach the one-shot greedy anchor there.

The repair, however, is the smaller half of this paper. Knowing that a bias exists does not tell a practitioner whether repairing it is worth anything on their distribution, so we characterize when the first step matters, quantitatively and in advance. A discrete control transplants the same head to TSP and CVRP, where symmetry makes starts equivalent, and finds the repair behaviorally inert there. A continuous budget sweep on orienteering then turns the contrast into a calibration: a model-free *start-stakes* measure, computable by classical local search in seconds per instance with no neural network in the loop, predicts the realized one-shot benefit of the repair across the entire sweep with $R^2 = 0.96$. Part of that correlation is expected by construction (Section 4.2); the empirical content is the conversion rate: the realized benefit is a near-constant 0.72 to 0.75 of the available start value at every budget. The prediction holds for this model family trained on the target distribution; off-distribution the two factors separate, with the stakes still measuring real start value that the trained head no longer converts (Section 4.2). The sweep also shows that, within the measured range, the near-zero-benefit end of the spectrum is a matter of problem structure, not of slack: only symmetry, not abundance, removes the decision.

We make four contributions. First, we locate a paradigm-level bias in multistart constructive solvers: the first step is hardcoded as a symmetry prior that never enters learning, which is sound on must-visit routing but excludes the decision that matters most on selective routing. Second, we characterize the bias predictively: a cheap, model-free measurement of a distribution's start-stakes predicts how much repairing the bias will be worth for models of this family trained on the target distribution ($R^2 = 0.96$ across five budgets spanning visit fractions 24% to 81%; part of the correlation is structural, Section 4.2). The conversion rate of available start value into realized value is near-constant at about three quarters and transfers unchanged to PCTSP. Third, we repair the bias with a minimal change that restores the first step's credit to the policy through one linear head of 128 parameters, an existence proof that the diagnosis suffices to act on: low-budget

decoding improves substantially with full-budget performance untouched. A reimplementation of the closest alternative mechanism from its published description, an auxiliary selection agent with about 1,700 times the selector parameters, corroborates the diagnosis by repairing the same bias through an independent design, reaching lower gaps on both problems at every budget while landing on the same sparse near-optimal starts. Fourth, we bound the findings with evidence across seeds, an exact randomized-start control, a geometric start-set instrument whose lower-bound character we quantify, and distribution-shift evaluations that locate where the learned policy's advantage inverts.

## 2 Method

### 2.1 Setting: selective routing under a decoding budget

An Orienteering Problem (OP) instance consists of a depot $x_0$, nodes $x_1, \ldots, x_n$ with prizes $\rho_1, \ldots, \rho_n$, and a length budget $T$; a solution is a depot-to-depot cycle that visits a subset of nodes within the budget and collects their prizes. The solver makes two coupled decisions, which subset to serve and in which order. We call such problems *selective routing*, in contrast to visit-all problems such as the TSP where the subset is fixed; the Prize-Collecting TSP (PCTSP) shares this structure.

Constructive neural solvers (Kool et al., 2019a; Kwon et al., 2020) encode an instance into node embeddings $h_0, \ldots, h_n$ and decode a solution autoregressively, one node per step, with a learned policy $\pi_\theta$. At inference the practical limit is the *decoding budget*: the number of rollouts a deployment can afford per instance. Reported benchmark numbers are typically produced with hundreds of rollouts per instance; a routing engine embedded in a planning loop often affords one. At the costs we measure (Appendix A), decoding a single instance one-shot takes about 6 ms on one CPU core or 10 ms on a GPU. A dispatch or re-planning loop that must score many candidate assignments per decision therefore operates at the one-to-few-rollouts end of this range.

POMO (Kwon et al., 2020) trains and decodes with $N$ parallel *multistart* rollouts. All rollouts share the encoder and decoder; rollout $s$ is forced to visit node $s$ first, and the $N$ trajectories are tied together by a shared baseline in the REINFORCE gradient,

$$\nabla_\theta \mathcal{L}_{\text{POMO}} = -\,\mathbb{E}\left[\frac{1}{N}\sum_{s=1}^{N}\big(R_s - b\big)\,\nabla_\theta \log \pi_\theta(\tau_s)\right], \qquad b = \frac{1}{N}\sum_{s=1}^{N} R_s, \tag{1}$$

where $R_s$ is the return of trajectory $\tau_s$. On the TSP this design is elegant: by cyclic symmetry every node begins some optimal tour, so forcing $N$ distinct first nodes is exploration at zero cost.

### 2.2 The first-step bias in the decoding stack

The failure of this recipe on selective routing has been reported as a performance number, without a mechanism-level account (Berto et al., 2025). It sits in three lines of the decoding stack of the reference implementation in RL4CO (Berto et al., 2025).[1]

**The first step is not a decision.** Under multistart decoding the first action of rollout $s$ is not sampled from $\pi_\theta$. It is computed outside the policy, written into the environment, and assigned log-probability zero. The policy's likelihood of a trajectory begins at step one. Whatever happens at step zero can never receive gradient, in training or anywhere else.

**The injected starts are blind.** The start of rollout $s$ is node $s$, enumerated as $\big(s \bmod n\big) + 1$ regardless of prizes, budget, or geometry. For OP the implementation adds a guard for budget-infeasible starts that resamples feasible ones *with* replacement, so scarce feasible starts can be drawn repeatedly and rollouts silently duplicated; for PCTSP there is no guard at all.

[1] rl4co v0.6.0. Start injection and its zero log-probability: `utils/decoding.py`, `DecodingStrategy.pre_decoder_hook`; the code comment reads "first logprobs is 0, so p = 1". Start enumeration: `utils/ops.py`, `select_start_nodes`. Shared baseline: `models/rl/reinforce/reinforce.py`. We verified the same structure on the current main branch.

**Bad starts bias the shared baseline.** Under Eq. (1) a forced start in a region the budget cannot profitably reach hurts twice. The rollout itself is wasted, and its low return drags the shared baseline $b$ down, which inflates the advantage of every sibling rollout and blurs the credit signal they train on.

Taken together this is a paradigm-level bias rather than a bug. The multistart paradigm hardcodes the assumption that every node is equally worth committing to first. On selective routing, where whether to visit a node is itself the decision, this prior excludes from learning the choice that most needs to be learned: which part of the instance deserves the budget. The most visible corollary is one-shot decoding. A POMO model has never trained, and can never train, a free first step, and its one-shot greedy gap on OP reflects that (12.4% for the official single-seed run; retrained models with a free first step span a measured 10.7% to 14.7% across seeds, Section 3).

### 2.3 Selection-aware starts

Our repair returns the first step to the policy while leaving the multistart skeleton, the shared baseline, the encoder, the decoder, and the training recipe unchanged. We call the result *selection-aware starts* (SAS).

**A start policy.** A single linear layer $w \in \mathbb{R}^d$ scores every node from its encoder embedding, $u_i = w^\top h_i$. The depot and all budget-infeasible nodes are masked to $-\infty$, and the scores define a start distribution $\pi_{\text{head}}(i) = \text{softmax}(u)_i$. Throughout, $\pi_\theta$ is the decoder policy, $\pi_{\text{head}}$ the start distribution just defined (its parameters $w$ are part of $\theta$), and $s_s$ the start assigned to rollout $s$. The head adds $d$ parameters (128 in our models), which is deliberate: the claim under test is that the missing piece is credit at step zero, not capacity. Throughout the paper, *the repair* refers to exactly this change: the start head's step-zero log-probability enters the policy gradient, and nothing else in POMO changes.

$N$ **distinct starts.** Training samples $N$ starts from $\pi_{\text{head}}$ without replacement by Gumbel-top-$k$ (Kool et al., 2019b), preserving POMO's variance-control skeleton of $N$ distinct trajectories. A temperature on the head logits follows a fixed schedule (5, 3, then 1 from the third epoch), set before training and never tuned. Because the credit term below reads the same tempered logits, the schedule scales the head's credit gradients during the first two epochs; with $N{=}n$ and every start feasible, as holds throughout training on the synthetic sets, it has no effect on which starts are taken.

**Restoring step-zero credit.** When starts are sampled from $\pi_{\text{head}}$, the log-likelihood of rollout $s$ is

$$\log \pi(\tau_s) \;=\; \log \pi_{\text{head}}(s_s) \;+\; \sum_{t \geq 1} \log \pi_\theta(a_t \mid \cdot). \tag{2}$$

Substituting Eq. (2) into the POMO gradient of Eq. (1) splits it into two parts,

$$\nabla \mathcal{L} = \underbrace{-\,\mathbb{E}\left[\frac{1}{N}\sum_s (R_s - b)\,\nabla \sum_{t \geq 1} \log \pi_\theta(a_t \mid \cdot)\right]}_{\nabla \mathcal{L}_{\text{POMO}} \text{ as implemented}} \underbrace{-\,\mathbb{E}\left[\frac{1}{N}\sum_s (R_s - b)\,\nabla \log \pi_{\text{head}}(s_s)\right]}_{\text{the term hardcoded to zero}}. \tag{3}$$

The first part is exactly the gradient that the existing implementation computes, because its trajectory likelihood starts at step one. The second part is exactly what the hardcoded zero deletes. We therefore train with

$$\mathcal{L} \;=\; \mathcal{L}_{\text{POMO}} \;-\; \frac{1}{N}\sum_{s=1}^{N} \underbrace{(R_s - b)}_{\text{detached}} \log \pi_{\text{head}}(s_s), \tag{4}$$

with the same shared baseline $b$ and all $N$ rollouts participating, which reproduces Eq. (3) term by term. In this sampled-start setting the added term carries no weight because Eq. (3) assigns it none; a coefficient other than 1 would compute the gradient of a different objective than Eq. (1).

**What the reported runs train.** The identity above describes the sampled-start regime, and the loss form is exact only there. Our reported runs do not operate in it: on the synthetic training sets every start is budget-feasible and training uses $N{=}n$, so the start set is exhaustive rather than sampled, and Eq. (4) reduces to an advantage-weighted likelihood of every start under $\pi_{\text{head}}$ (Appendix B details this regime). The loss introduces no weight to tune, and the temperature schedule, the one quantity that shapes the term, is a design constant fixed before training and never searched.

**Head-local credit.** The start head reads the encoder embeddings through a stop-gradient, so the credit term of Eq. (4) updates only the head's $d$ parameters and the encoder continues to be trained by the standard POMO loss alone; Appendix B records the degradation observed when this separation is removed.

### 2.4 Decoding at deployment

SAS decodes at any budget. With $N$ rollouts it takes the top-$N$ starts of the head deterministically. With a single rollout it takes the head's argmax start and decodes greedily; we call this one-shot mode *greedy-head* and use that name throughout the paper and in all tables and figures. Plain POMO has no learned counterpart at $N{=}1$: its deployable one-shot decode is greedy decoding with a free first step, which is precisely the step it never trained. This asymmetry, one model with a trained step zero and one without, is measured directly in Section 3.

All comparisons in this paper match decoding budgets exactly: $N$ multistart rollouts cost $N$, augmented multistart with 16 symmetries costs $16N$, and every evaluation path calls the policy with the target environment passed explicitly through one shared code path.

## 3 The repair on OP50 and PCTSP50

### 3.1 Setup

**Data and reference.** We evaluate on the canonical OP test set of Kool et al. (2019a): 10,000 instances with $n{=}50$ nodes, distance-based prizes, budget $T{=}3$, generated by the official script with test seed 1234. All gaps are relative to Compass, the released EA4OP solver (Kobeaga et al., 2018) and the strongest classical heuristic reported on this set (16.17 average prize, the same reference used across the literature).

**Training.** Every model is trained with the official POMO recipe for OP50 exactly as published (100 epochs, batch 64, learning rate $10^{-4}$ with the standard decay schedule), changing nothing but the start mechanism. We train three configurations. *enumerated* is unmodified POMO with the forced starts of Section 2.2. *uniform* replaces the start head by a uniform distribution over feasible starts, with the credit term of Eq. (4) removed; code-identical to SAS in every other respect, it is the control that isolates learned start selection from merely randomized starts. *SAS* is the full method. All configurations train with $N{=}n{=}50$ rollouts, so they see the same start coverage; they differ only in how starts are chosen when rollouts are scarce and in whether step zero receives credit. The uniform and SAS configurations train three seeds each (mean $\pm$ std reported).

**Decoding schemes.** We decode every configuration at matched budgets: one rollout (*greedy-head* for configurations with a start distribution, plain greedy for POMO), multistart with $N \in \{10, 25, 50\}$, and multistart with 16 symmetric augmentations ($50 \times 16 = 800$ rollouts), the scheme under which published full-budget numbers are produced. Significance is assessed with paired Wilcoxon signed-rank tests over the 10,000 instances, pairing configurations trained with the same seed. Comparison thresholds and statistical tests are fixed by this protocol and identical across configurations and budgets.

### 3.2 Main result: low-budget gains with full-budget performance intact

With a single rollout, plain POMO decodes at a 12.4% gap (Table 1). SAS decodes at 3.9%, and the uniform control shows that random free starts do not help (11.5%): the gain comes from *learned* starts, not from

Table 1: Attribution across three configurations on OP50 (gap % to Compass, 10,000 instances; mean ± std over 3 seeds; *enumerated* is the official single-seed POMO recipe). *uniform* differs from SAS only in the start mechanism, making it an exact control. One-shot rows decode through each configuration's start distribution (greedy-head); for the uniform configuration this draws one uniformly random feasible start per instance. Bold marks the best configuration at budgets where the paired difference is significant for every seed (Section 3.3); at 50 and 800 rollouts the configurations are statistically indistinguishable. AM and MDAM rows are published numbers on the same test set and reference; conversion details in Appendix A.

| decoding scheme (rollouts) | enumerated (POMO) | uniform | SAS (ours) |
|---|---|---|---|
| one-shot via start distribution (1) | — | $11.532 \pm 0.080$ | $\mathbf{3.866 \pm 0.021}$ |
| multistart 10 | 2.875 | $2.858 \pm 0.028$ | $\mathbf{1.293 \pm 0.024}$ |
| multistart 25 | 1.669 | $1.652 \pm 0.026$ | $\mathbf{1.094 \pm 0.019}$ |
| multistart 50 ($= n$) | 1.096 | $1.081 \pm 0.017$ | $1.078 \pm 0.022$ |
| multistart 50 × aug 16 (800) | 0.009 | $0.025 \pm 0.020$ | $0.014 \pm 0.016$ |
| one-shot, plain greedy (1) | 12.438 | $12.862 \pm 1.499$ | $13.244 \pm 2.176$ |
| published, same set (Kool et al., 2019a): AM greedy (1) 3.23; AM sampling (1280) 0.60 | | | |
| published, same set (Xin et al., 2021): MDAM greedy (5) 1.55; MDAM beam 50 (≈ 250) −0.25 | | | |

lifting the forced enumeration. A hand-crafted start rule narrows the gap but does not close it: starting each instance at its highest prize-to-depot-distance node, on the same trained backbones, decodes at 4.81%±0.04, still a point behind the learned head. The advantage persists through small budgets (1.3% versus 2.9% at ten rollouts, 1.1% versus 1.7% at twenty-five) and vanishes where it should, at $N{=}n$, where every start is taken and start selection has nothing left to choose. At the full published budget (800 rollouts) the three configurations are statistically indistinguishable, so restoring step-zero credit costs nothing where POMO was already strong.

The reference point for the one-shot number is AM, the sampled-start model family that never had the bias this paper repairs: its first step is an ordinary policy action. On this same test set and reference, AM's published one-shot greedy gap is 3.23% (Kool et al., 2019a), and SAS's 3.87% lands within 0.7 points of it. What AM gives up for that free first step is the full-budget end of the curve, 0.60% with 1280 samples against 0.014% for SAS at 800 rollouts. POMO holds the full-budget end and loses the one-shot end; AM holds the one-shot end and loses the full-budget end. SAS spans the budget curve while changing only 128 parameters of the multistart architecture, making the first step a learned decision inside that design rather than choosing between the two designs.

The same models also expose the free first step directly (Table 1, "one-shot, plain greedy"): across configurations it decodes at 12.4% to 13.2%, worse than starting from a uniformly random feasible node (11.5%), because the decoder's argmax at an unconditioned first step was never shaped by any training signal; and it scatters across seeds, landing anywhere from 10.7% to 14.7% (std 2.18 points), where the trained first step lands at 3.9% with a cross-seed std of 0.021 points (Figure 1). Learning step zero makes the one-shot decode simultaneously better and reproducible.

### 3.3 Budget dependence of the advantage

Figure 3 (Appendix A) plots the paired advantage of SAS over its uniform control as a function of the decoding budget. The advantage is largest where deployments are tightest: 7.7 points at one rollout, 1.6 at ten, 0.6 at twenty-five, and it reaches zero at $N{=}n{=}50$. Significance is assessed per seed and never pooled: paired Wilcoxon gives $p < 10^{-6}$ for every seed at every budget up to twenty-five rollouts (full per-seed statistics in Appendix A). Spending the same budget without choosing starts does not reproduce this: sampling and beam search on the same baseline checkpoint trail SAS at every matched budget (Table 8). Start selection is a budget allocation mechanism; once the budget covers every start there is nothing to allocate, and a method that still showed an advantage there would be claiming something other than what it does. The deployment regime where start selection matters, one to a few dozen rollouts per instance, is the planning-loop regime of Section 2.1.

Table 2: Attribution across three configurations on PCTSP50 (gap % to OR-Tools cost 4.48, 10,000 instances; mean ± std over 3 seeds). Same design as Table 1. AM anchors: as re-trained and reported by Berto et al. (2025), and as originally published against the same OR-Tools reference (Kool et al., 2019a).

| decoding scheme (rollouts) | enumerated (POMO) | uniform | SAS (ours) |
|---|---|---|---|
| one-shot via start distribution (1) | — | $11.900 \pm 0.040$ | $\mathbf{4.679 \pm 0.153}$ |
| multistart 10 | 4.189 | $4.206 \pm 0.022$ | $\mathbf{2.351 \pm 0.056}$ |
| multistart 25 | 2.841 | $2.852 \pm 0.008$ | $\mathbf{2.178 \pm 0.031}$ |
| multistart 50 ($= n$) | 2.141 | $2.160 \pm 0.007$ | $2.158 \pm 0.031$ |
| multistart 50 × aug 16 (800) | 1.011 | $1.025 \pm 0.004$ | $1.024 \pm 0.029$ |
| one-shot, plain greedy (1) | 9.241 | $10.250 \pm 0.281$ | $10.589 \pm 1.159$ |
| AM greedy (1): 2.46 re-trained (Berto et al., 2025); 2.66 original (Kool et al., 2019a) | | | |

**A classical constructor at matched latency.** The classical alternative at this latency is the construction stage of the reference solver itself, so we measure it. The same prize-per-detour greedy insertion that seeds the local-search reference, run without any descent, decodes at a 30.1% gap (best-of-one), 14.7% with its full five-criterion portfolio, and 11.3% at best-of-sixteen with noise, against SAS's one-shot 3.9%. The latencies are single-instance measurements, not batched throughput: one construction costs 6.7 ms on one CPU core and best-of-sixteen 152 ms, while one SAS one-shot decode of a single instance costs about 6 ms on one CPU and 10 ms on a GPU (Table 5, Appendix A). At matched per-instance latency, the repaired one-shot decode therefore beats classical construction by a factor of about eight in gap, and the classical side needs its full local search, at 0.3 seconds per restart, to pass the neural configuration; this ordering holds on the training distribution and reverses off it (Section 4.3).

### 3.4 What the head selects

We test geometrically whether the learned head selects the starts that near-optimal solutions use. For the first 1,000 test instances we run an independent local-search solver (16 restarts of a portfolio iterated local search, specified in Appendix E) and collect the set of nodes adjacent to the depot in any solution within 99.5% of the best found. This near-optimal start set is sparse: 2.2 nodes per instance on average, never more than 5. The head's argmax start falls inside it on 52.8% ± 5.6% of instances (three seeds; per-seed 95% binomial intervals are within ±3.1 points), against a 4.4% chance rate for a random feasible start, a factor of twelve. Because 16 restarts only sample the near-optimal solution space, the true hit rate can only be higher. Figure 4 (Appendix E) shows the first three test instances, including one where the head's pick lands just outside the sampled set.

The head's softmax is nearly a point mass (top-1 probability 0.99 on average). The sparsity measured above says why: near-optimal solutions use about two starts out of fifty, so a start policy that has learned the problem concentrates its mass there. Two diagnostics confirm this is what happens. Concentration is consistent across training runs, with cross-seed Jaccard overlap of 0.87 and 0.89 for top-10 start sets, so three independently trained heads select essentially the same instance-determined starts. And the selected starts are the right ones, as the hit rate above shows. A learned temperature floor could keep the head's entropy up if an application wanted calibrated start sampling; we leave this to future work.

### 3.5 The same repair on prize-collecting routing

We transplant the entire OP50 design to PCTSP50 with nothing re-tuned: the same official training recipe, the same three configurations with three seeds each, the same budget-matched schemes, the same criteria translated line by line to PCTSP's scale, and the same near-optimal start-set instrument, re-run with PCTSP's objective (Appendix C). Gaps are relative to the OR-Tools reference cost of 4.48 on the canonical test set (Berto et al., 2025; Kool et al., 2019a). A sensitivity analysis over the penalty factor (Appendix C) confirms that the selection pressure the repair needs exists at this benchmark's parameters.

**The mechanism transfers.** Every mechanism-level result of the OP study reproduces (Table 2). The paired advantage over the uniform control decays along the same curve, $+7.2 \pm 0.1$ points at one rollout, +1.9 at ten, +0.7 at twenty-five, zero at $N{=}n$ and at the full budget (OP: +7.7, +1.6, +0.6, zero, zero), with all nine seed-scheme pairings positive and per-seed paired Wilcoxon $p < 10^{-6}$ throughout (per-seed pairing, never pooled). The geometric evidence reproduces as well: PCTSP's near-optimal start sets are as sparse as OP's (2.1 nodes per instance), the head concentrates just as hard (top-1 probability 0.99), and it concentrates on the right starts, with hit@1 of $62.1\% \pm 6.5\%$ over 1,000 instances against a 4.3% chance rate, a factor of fourteen (OP: 52.8%, a factor of twelve). Full-budget performance is intact (1.02% versus the baseline's 1.01%). The same first-step bias exists on PCTSP, the same 128-parameter head removes it, and the same instrument shows the head learning the same kind of sparse start structure.

**What does not transfer.** SAS decodes one-shot at $4.68\% \pm 0.15$, short of the AM one-shot greedy anchors on this benchmark: the 2.46% reported by the benchmark re-implementation (Berto et al., 2025) and the 2.66% of AM's original run against the same OR-Tools reference (Kool et al., 2019a). The gap is nearly twice either anchor, and the miss is not a seed accident (all three seeds sit between 4.5% and 4.8%). The context is the family ceiling: on PCTSP this model family sits farther from the classical reference at every budget, with our retrained POMO still 1.0% from OR-Tools at the full 800-rollout budget where its OP counterpart reaches 0.009% of Compass, and AM's one-shot greedy (2.46% to 2.66%) lands between our baseline's multistart-25 and multistart-50 results. Start selection closes its share of the one-shot gap (11.9% to 4.7%, with the head picking near-optimal starts on 62% of instances); the distance that remains is consistent with the family ceiling visible at every budget. We therefore claim mechanism-level transfer, on the same evidence standard as OP, and we do not claim the budget-spanning result on PCTSP: the repaired model's one-shot decode does not reach the one-shot greedy anchor there.

## 4 When the first step matters

The repair matters only where the first step is a genuine decision, and this section maps that scope. A discrete control asks where the first step stops being a decision at all, a continuous budget sweep turns the answer into a quantitative calibration, and distribution-shift evaluations identify where the calibration ceases to apply.

### 4.1 The must-visit control

If the diagnosis is right, the repair should matter only where the first step is a decision. We transplanted SAS unchanged to TSP50 and CVRP50 (official recipes; two seeds each, since a qualitative control needs consistency rather than tight intervals), expecting the head to flatten where symmetry makes starts equivalent: exactly on TSP by cyclic symmetry, nearly on CVRP with its fixed depot. Instead, the head concentrates on must-visit routing exactly as it does on selective routing (top-1 probability 0.97 to 0.99, against 0.02 for uniform; Appendix D). Concentration is therefore the head's universal behavior; what it lands on, and what it costs, differ by problem. On selective routing the concentration is aimed, landing on near-optimal starts at twelve to fourteen times chance (Section 3.4); on must-visit routing starts are equivalent or nearly so and the concentration is all but costless, with SAS within 0.2 points of plain POMO at every multistart budget on both problems and one-shot residuals inside the parity bands (Table 7, Appendix D). The bias we repair is architectural and problem-independent; what varies with the problem class is its consequence, absent under symmetry and largest where the first step carries a real decision. Section 4.2 makes this two-point contrast continuous, and shows that the boundary between the two regimes is set by problem structure rather than by budget.

### 4.2 Predicting the value of the repair

The must-visit control gives two ends of a spectrum: on selective routing the first step carries a large decision, on must-visit routing it carries none. This section makes the contrast continuous and asks the deployment question directly: given a problem distribution, can one predict how much restoring step-zero credit will be worth, before training anything?

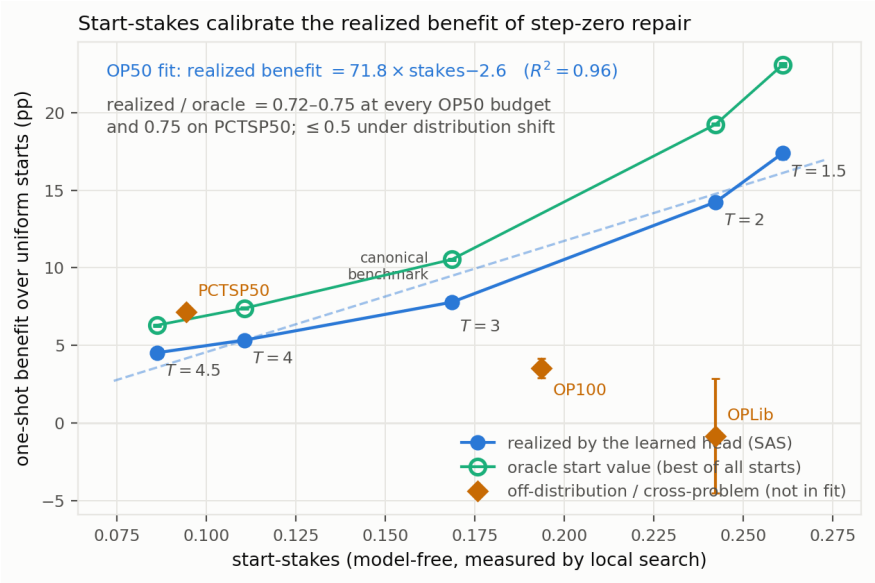


Figure 2: The calibration curve: start-stakes, measured on the instance distribution by local search with no model in the loop, track the realized one-shot benefit of the learned head (blue), with rank correlation 1.0 over the five budgets, linear fit $R^2 = 0.96$, and the oracle start value (rings) above at a near-constant ratio. Error bars are 3-seed standard deviations, mostly smaller than the markers. Diamonds are cross-problem and off-distribution points, excluded from the fit (see text). The PCTSP50 value follows the sweep's same-checkpoint convention; the OP100 and OPLib values are the zero-shot advantages of Section 4.3, measured against an independently trained uniform control (Appendix G).

Table 3: Budget sweep on OP50 (3 seeds each; benefit and oracle in percentage points of the per-budget reference; stakes and hit@1 as defined in the text; hit@1 values are lower bounds, see the metric characterization below).

| budget $T$ | visit frac | $\lvert S\rvert$ | stakes | realized benefit | oracle | realized/oracle | hit@1 ($\geq$) |
|---|---|---|---|---|---|---|---|
| 1.5 | 0.24 | 2.05 | 0.261 | $17.38 \pm 0.38$ | 23.09 | 0.75 | 0.57 |
| 2.0 | 0.34 | 2.10 | 0.242 | $14.24 \pm 0.27$ | 19.25 | 0.74 | 0.18 |
| 3.0 (canonical) | 0.55 | 2.21 | 0.169 | $7.78 \pm 0.06$ | 10.53 | 0.74 | 0.53 |
| 4.0 | 0.73 | 2.39 | 0.111 | $5.34 \pm 0.03$ | 7.39 | 0.72 | 0.63 |
| 4.5 | 0.81 | 2.48 | 0.086 | $4.52 \pm 0.08$ | 6.29 | 0.72 | 0.63 |

**Setup.** We sweep the OP50 length budget over five values, $T \in \{1.5, 2.0, 3.0, 4.0, 4.5\}$, moving the near-optimal visit fraction from 24% to 81% of nodes; $T$=3 is the canonical benchmark and anchors the sweep to every other number in this paper. Each budget trains SAS with the unchanged recipe (three seeds), and every quantity below is measured on one model: the *realized benefit* is the one-shot gain of the learned start over a uniformly random feasible start on the same checkpoint, so it isolates start selection from all training-side effects, and the *oracle start value* is the gain of the best of all fifty starts over the same random baseline. Both are normalized by a per-budget local-search reference on the same 1,000 test instances (Appendix G). Two candidate predictors, with all their thresholds, are fixed elements of the protocol: the cardinality of the near-optimal start sets of Section 3.4, and a *start-stakes* measure that is deliberately model-free, the value deficit of forcing the tour to begin at a random feasible node (full definition in Appendix G).

**Start-stakes predict the realized benefit.** Figure 2 is the central result of this section. The realized benefit falls from 17.4 points at the tightest budget to 4.5 points at the loosest, and the start-stakes measure predicts it: the rank correlation across budgets is 1.0 and a linear fit gives $R^2 = 0.96$ with slope 71.8 (95% confidence interval $[43, 100]$ from the five-point regression; bootstrap resampling of both the three seeds per

budget and the 200 stakes instances per budget keeps $R^2$ within $[0.93, 0.97]$, and refitting with any single budget left out keeps it within $[0.95, 0.98]$, so the fit quality is not carried by one tier or one lucky draw). The stakes themselves are measured to a standard error below 0.005 per budget, negligible against their 0.09 to 0.26 range. The measure costs a few CPU seconds per instance, involves no neural network, and can be evaluated on a target distribution before committing to training: it supplies the quantitative basis of the calibration, turning the qualitative contrast of the preceding sections into an instrument. The cardinality of near-optimal start sets, the second candidate predictor, is structurally pinned: it stays between 2.0 and 2.5 starts per instance across the entire sweep, a dynamic range too narrow to support a regression (its nominal $R^2$ of 0.89 is not meaningful on a span of one percent of the node count), and we report this as a finding about the problem rather than as a working predictor: OP's tours leave and return to a fixed depot, so the near-optimal depot-adjacent corridor stays sparse no matter how much budget is available.

**The correlation is not a normalization artifact.** The benefit and the stakes both divide by a local-search quantity, so one could suspect a ratio-correlation artifact from the shared style of denominator. Two sensitivity fits rule this out. Normalizing the benefit by the model's own uniform-start performance instead, so that no local-search quantity appears anywhere in the outcome, leaves the relation intact ($R^2 = 0.94$, rank correlation 1.0). Removing normalization altogether collapses the regression ($R^2 = 0.04$), which sharpens what the calibration is about: absolute prize scales differ across budgets by construction, and the calibration is a statement about relative value, the benefit as a share of achievable performance, the quantity gaps are stated in.

**The near-zero-benefit end is reached by structure, not by budget.** We expected the loosest budget to approach the must-visit regime, with the benefit falling inside the parity band of Section 4.1. It does not: at 81% visit fraction the benefit is still 4.5 points and the stakes are still 8.6%, far from the $\pm 0.5$-point parity band that TSP and CVRP exhibit. The comparison with the must-visit control then localizes the cause. What makes the first step irrelevant to tour quality on TSP is rotation symmetry, the structural fact that every start yields the same cycle; loosening a selective problem's budget never manufactures that symmetry, because the tour still starts and ends at the depot and the first hop remains a committed geometric decision. The near-zero-benefit end of the spectrum is reachable only through problem structure, not through budget.

**A near-constant conversion rate.** The oracle start value falls with the stakes exactly as the realized benefit does, and their ratio is nearly constant: 0.72 to 0.75 across a fourfold range of benefit (Table 3); pooled over the fifteen budget-seed cells the ratio is $0.734 \pm 0.016$, with every single cell inside $[0.706, 0.762]$. The trained head converts about three quarters of whatever start value exists into realized one-shot value, at every budget we measured. This ratio is computed as $(P_{\text{SAS}} - P_{\text{unif}})/(P_{\text{oracle}} - P_{\text{unif}})$ from three decodes of the same checkpoint; the local-search reference cancels algebraically, so the constancy is a property of the model family, measured independently of the instrument that defines the predictors and the hit rates.

**Where the calibration stops: cross-problem and off-distribution points.** The sweep says what the relation is; three further points, overlaid as diamonds in Figure 2 and never entered into the fit, say where it stops. We ran the identical stakes instrument, with fixed parameters and per-instance local-search anchors, on PCTSP50 (a minimization mirror of the same instrument), on OP100, and on the OPLib subset where the instrument is computable (Appendix G). PCTSP50 measures stakes of 0.094; the OP50 line predicts a 4.2-point benefit and the same-checkpoint measurement gives $7.2 \pm 0.2$, so the specific line under-predicts across problems. The conversion rate, however, transfers unchanged: the PCTSP head realizes 0.75 of its oracle start value, inside the OP50 band. The two shifted OP sets break in the opposite direction and more instructively. OP100 measures stakes of 0.19 and OPLib of 0.24, both at or above the canonical benchmark's 0.17, yet the realized zero-shot benefits are 3.5 points and $-0.8$ points against line predictions of 11.3 and 14.8. The stakes are not wrong about the start value. On the same checkpoints, decoding from every start still beats the head's one-shot choice by 6.8 points on OP100 and 14.1 points on OPLib, so the value the instrument measures persists off-distribution; what collapses is the trained head's conversion of it, from about three quarters in-distribution and cross-problem to about one half under moderate shift and to nothing under compound shift. Stakes measure the start inequality of a problem distribution; converting that inequality into realized benefit is the trained head's job, and that job is distribution-bound (Section 4.3).

The calibration therefore predicts the realized benefit for this model family trained on the target distribution, and off-distribution it still bounds the available start value while saying nothing about how much of it an unadapted head will capture.

**What the hit rate measures, characterized.** Every hit@1 number in this paper is a lower bound, and the sweep lets us quantify how loose that bound can get. The near-optimal start sets are built from the restarts of one local-search procedure whose greedy prize-per-detour construction has a corridor preference: all sixteen restarts tend to enter the same basin, which is also why $|S|$ stays pinned near two, and where several corridors of nearly equal value exist the instrument reports only its own. The budget $T$=2.0 is the loosest point: the head's measured hit@1 there drops to 18% while its benefit (14.2 points) and conversion ratio (0.74) remain on the calibration curve. Forcing the solver to start at the head's chosen node resolves the tension (a post-hoc probe, designed only after all sweep outcomes were recorded): a quarter of the picks counted as misses reach 99.5% of the unconstrained optimum, so the corrected bound is at least 33%, and the median counted miss still reaches 97% of the optimum: the head works normally there, and the bound is loose because near-tied corridors are most prevalent at that budget. The same characterization applies, with a smaller discrepancy, to every hit rate in Sections 3.4 through 4.3.

**What this calibration does and does not claim.** Stakes and oracle value are structurally related, since both measure how unequal starts are; that part of the correlation is expected by construction, and we quantify it: regressing the oracle start value on the stakes gives the same tightness as the headline fit ($R^2 = 0.96$, slope 94.4). The realized line is that construction multiplied by the conversion rate; its slope of 71.8 is 0.76 of the oracle slope, just above the per-budget band (0.72 to 0.75) and inside the range of the fifteen budget-seed cells ($[0.706, 0.762]$), so "stakes predict the benefit" and "the head converts a near-constant three quarters" are two statements of one empirical fact, and we do not count them as two. The empirical content is exactly the second factor. The *realized* benefit tracks the stakes only because the trained head does capture the available value on every distribution; nothing in the training objective guarantees that, and the off-distribution points show it is a real degree of freedom that can fail: there the by-construction factor survives (the stakes keep tracking available start value) while the conversion is what breaks. Beyond that, a single scalar organizes the entire spectrum, including the canonical benchmark and the structural near-zero-benefit end. And the claim is scoped twice. It is per-distribution: within a budget, the per-instance cardinality $|S_i|$ does not predict the per-instance benefit (median within-budget rank correlations are indistinguishable from zero), so the calibration licenses distribution-level decisions, such as whether the repair is worth deploying, and not instance-level ones. And it is within-family: the prediction holds for this architecture trained on the target distribution, while for a shifted deployment the stakes bound only the available value.

### 4.3 Behavior under distribution shift

A learned start policy depends on the match between its training distribution and the test distribution, so we measure where that dependence breaks down. We stress the same OP50-trained models, zero-shot and unchanged, at two levels of distribution shift: OP100 (the same generator with size doubled and the budget at its canonical value of 4 instead of 3) and the classical OPLib benchmark, which shifts three axes at once: real city geometries instead of uniform squares, three prize schemes of which only one resembles the training prizes, and budgets between 2.0 and 22.1 in normalized units where training saw the single value 3. Table 4 summarizes the finding of this section: the advantage decays with shift, and under compound shift the learned start selection itself fails in a way we can describe precisely.

**Moderate shift: the advantage survives at half strength.** On OP100 the one-shot advantage over the uniform control is +3.5 points (14.0% versus 17.5%, positive for all three seeds; full ladder in Table 9), decays into seed noise by fifty rollouts, and is statistically zero at $N$=$n$=100 and at the full augmented budget, the same shape as in-distribution with roughly half the magnitude. Two caveats come with it. The cross-seed std of the one-shot decode grows from 0.02 points in-distribution to 1.0 points here, so the stability half of Figure 1 is an in-distribution claim and we do not extend it off-distribution. The head's absolute hit

Table 4: The one-shot advantage and the head's hit rate decay together under distribution shift (mean ± std over 3 seeds). Advantage = paired one-shot gap difference, uniform control minus SAS. hit@1 = fraction of instances where the head's argmax start lies in the near-optimal start set extracted by independent local search (16 restarts; a lower bound). OP50 hit@1 is measured on the first 1,000 test instances, OP100 on the first 200, and OPLib on the 102 instances with $n \leq 300$, where the extraction is computable.

| test set | one-shot advantage (pp) | head hit@1 | chance |
|---|---|---|---|
| OP50 (in-distribution) | $+7.67 \pm 0.09$ | $52.8\% \pm 5.6\%$ | 4.4% |
| OP100 (size + budget) | $+3.54 \pm 0.62$ | $47.2\% \pm 9.4\%$ | 2.5% |
| OPLib (compound) | $-0.84 \pm 3.71$ | $11.8\% \pm 4.3\%$ | 2.0% |

rate on the near-optimal start sets dips from 53% to 47%, tracking the halved advantage (the chance rate also halves as $n$ doubles, so the head stays well above random).

**Compound shift: start selection fails, and we can say why.** On OPLib the ordering reverses: unmodified POMO reaches a mean gap of 49.2% at fifty rollouts, the uniform control 51.2%, and SAS 53.5% (Table 10), with SAS worst at every budget and a cross-seed spread of 3 to 4 points; even the sign of the one-shot comparison flips between seeds (an enumeration-ordering caveat for the baseline configuration on this benchmark is recorded in Appendix A). The diagnosis is specific. The head still concentrates exactly as it does in-distribution (top-1 probability 0.98 to 0.99), and the target it should find still exists, since near-optimal start sets remain as sparse as on synthetic data (2.3 nodes per instance). It concentrates on the wrong nodes: the hit rate falls to 12%, and on 80% to 89% of instances the head places over 0.9 probability on a start outside the near-optimal set. The shift axes separate cleanly, with real geometry alone roughly halving the hit rate and the shifted prize schemes destroying it, and cross-seed agreement of the heads' top starts collapses alongside, which is the variance blow-up seen in the gap table; the per-axis hit rates and agreement numbers are in Appendix F.

One factor must be separated from the start mechanism entirely. All three configurations collapse identically with instance size: POMO's own gap grows from 15% at $n \leq 100$ to 78% beyond 1000 nodes (Table 11), with the uniform control and SAS within a few points of it at every bracket. The size axis is a property of this model family as a whole. The start choice fixes only the first node; what degrades with scale is the stepwise construction quality of the entire tour, which no start rule touches, and accordingly the three start rules we compare collapse in lockstep.

**The boundary, decomposed by the calibration instrument.** The stakes measurement of Section 4.2 separates what fails here. Both shifted sets measure high stakes (0.19 on OP100, 0.24 on the OPLib subset, at or above the canonical benchmark's 0.17), and the value those stakes point to is present: on the same checkpoints, decoding from every start still beats the head's one-shot choice by 6.8 points on OP100 and 14.1 points on OPLib. The start inequality survives the shift intact; the head's conversion of it is what decays, from three quarters in-distribution to about one half here and below zero under compound shift. A baseline with no training distribution to carry makes the cost concrete: the reference solver's construction stage alone, the same prize-per-detour greedy insertion of Section 3.3 with no local search, reaches a mean gap of 27% (best-of-one) and 17% (best-of-sixteen) on the same OPLib subset, where every neural configuration decodes one-shot between 42% and 46% and reaches only 27% to 32% with fifty rollouts. On the training distribution the neural one-shot beats that constructor by a factor of about eight; off it, the constructor wins by fifteen points or more. Latency-comparable classical construction is the appropriate deployment reference point on shifted distributions, and the neural advantage this paper measures is, on present evidence, a within-distribution property.

**What this boundary means.** These results bound the contribution: restoring step-zero credit buys its large low-budget gains where what the start policy learned matches the instance distribution, retains about half of them under moderate shift, and turns negative under compound shift, where a confidently wrong start is worse than a random feasible one. This is consistent with the diagnosed mechanism rather than in tension

with it. The repair makes the first step a learned decision, and any learned decision, ours or anyone's, carries its training distribution with it; a learned start policy is no exception, and the value of making the first step learnable comes with the same distribution dependence as every other learned component. What is specific to this paper is that we can measure the boundary and say precisely what breaks. Beyond that boundary lies a problem that belongs to the field rather than to any single method: synthetic-to-classical transfer is unsolved for neural OP solvers as a class. The supervised evaluation of Yao et al. (2025) reports zero-shot OPLib gaps of 13% to 72%, and every configuration we test degrades with the same size-bracket shape; their numbers and ours are samples of the same open problem. Making the start head robust to distribution shift, whether through test-time input normalization, training-time distribution augmentation, or head features invariant to prize and geometry rescaling, is future work aligned with the axis decomposition above.

## 5 Related Work

**Constructive neural routing and multistart decoding.** Attention-based constructive solvers (Kool et al., 2019a) decode routing solutions autoregressively from learned node embeddings. POMO (Kwon et al., 2020) added the multistart scheme this paper examines: $N$ rollouts forced to begin at $N$ distinct nodes, trained against a shared mean baseline, with instance augmentation on top. On the TSP the forced starts are justified by cyclic symmetry, and later work deepened the same idea; Sym-NCO (Kim et al., 2022) regularizes with solution symmetries. This line of work treats the first step as a free source of diversity. Our observation is that on selective routing it is not free: it silently removes the subset decision from the learned policy at the one step where it binds.

**Neural solvers on selective routing.** OP and PCTSP appear as secondary benchmarks from the start (Kool et al., 2019a; Xin et al., 2021; Kim et al., 2022), evaluated on synthetic instances against classical heuristic anchors, Compass for OP (Kobeaga et al., 2018) and iterated local search for PCTSP. The RL4CO benchmark (Berto et al., 2025) reports that POMO underperforms on these problems, and this reading, a strong architecture doing poorly on a problem family, is the departure point of our work.[2] Outside the constructive-RL line, BQ-NCO (Drakulic et al., 2023) reports strong OP results in an imitation-learning framework aimed at out-of-distribution size generalization. In the supervised line, SL-DABL (Yao et al., 2025) is to our knowledge the only neural OP work that reports zero-shot results on the classical OPLib collection, observing gaps of 13% to 72%, which documents how brittle synthetic-trained OP models are off distribution. None of these works locates a mechanism behind the selective-routing gap; they report it as a score.

**Diversity through policies versus diversity through starts.** A parallel line replaces POMO's start-based diversity with learned policy diversity. Poppy (Grinsztajn et al., 2023) trains a population with a winner-takes-all objective, and PolyNet (Hottung et al., 2025) conditions a single decoder on $K$ binary vectors, trains with a Poppy-style best-of-$K$ loss, and reports that free first steps then work better than forced ones.[3] SAS differs on all three axes: the diversity source is the start set itself (one policy, no latent vectors), the loss is POMO's original full-$N$ shared-baseline objective with one deleted term restored rather than a best-of-$K$ selection, and the first step remains a designated multistart step whose designator becomes a trained head with its log-probability in the gradient. The two approaches are complementary in principle, since a start head could sit under a latent-conditioned decoder; our results show that on selective routing the start decision alone, 128 parameters, closes most of the low-budget gap. Closest to our repair, the POMO+ preprint (Jakubicz et al., 2025) attaches an auxiliary agent that learns to choose POMO's starting nodes, evaluated on the must-visit CVRP. Our work differs in problem class (selective routing, where the forced start suppresses a decision rather than exploits a symmetry), in mechanism (128 parameters and no training

[2] Trained with the officially recommended recipe (RL4CO v0.6.0; full configuration in Appendix B and in our released code), our POMO baseline reaches substantially stronger multistart performance than the corresponding table row. We therefore compare against our own retrained baseline throughout, under one budget-matched protocol, and report the discrepancy so it can be checked against the released configuration.

[3] PolyNet's code is not released. In RL4CO v0.6.0 (`rl4co/utils/decoding.py`), multistart decoding applies environment-selected forced starts whenever more than one rollout is requested, unless a custom start-selection function is passed. We take the reported free-start result as the authors state it; our own free-start configurations are implemented and released with this paper.

loop of its own, against 214,913 and a separate agent), and in evidence (an exact control and a start-set analysis that attribute the gain and locate its boundary); Table 12 details the axes. Since its code is not released, we reimplemented POMO+ from its published description and ran it head-to-head on OP50 and PCTSP50 (Appendix H). The result corroborates the diagnosis: an independent design repairs the same bias, reaches lower gaps with more capacity (one-shot 2.60% against our 3.87% on OP50 and 3.32% against 4.68% on PCTSP50, ahead at every budget at about 1,700 times the selector parameters, Table 13), and lands its selections in the same sparse near-optimal start sets ours does. Where the larger selector and its training loop are affordable, they buy about one further point of one-shot gap; restoring step-zero credit alone, with 128 parameters, already closes most of the gap. A further line spends test-time budget on axes orthogonal to start selection: efficient active search adapts a subset of parameters per instance (Hottung et al., 2022), simulation-guided beam search couples the policy with rollouts (Choo et al., 2022), and COMPASS searches a latent policy space (Chalumeau et al., 2023).[4] These are complementary consumers of the same budget; our experiments include sampling and beam-search baselines at matched budgets (Section 3.3, Table 8), where learned starts dominate both in the deployment range.

**Classical orienteering.** The OP has a mature classical literature (Vansteenwegen et al., 2011), with strong heuristics (EA4OP (Kobeaga et al., 2018)), an exact branch-and-cut line (Kobeaga et al., 2024), and the OPLib instance collection. We use this infrastructure as measurement, Compass as the gap reference on synthetic sets and independent local search to extract near-optimal start sets in Section 3.4, rather than as competitors, and we also measure the classical side at matched latency: a single-instance neural decode costs 6 to 15 ms on one CPU core or one GPU, the same class as one classical greedy construction at 6.7 ms, while one local-search restart costs 0.34 s (Table 5); Section 3.3 reports what each buys at that latency in-distribution, and Section 4.3 where the ordering reverses.

## 6 Conclusion

The multistart paradigm behind constructive neural solvers hardcodes its first step as a symmetry prior that never enters learning. We located this bias at a specific point in the decoding path, showed that on selective routing it removes the decision that binds, and characterized when it matters: a model-free start-stakes measurement tracks the realized benefit of repairing it, for this model family trained on the target distribution, across a budget sweep with $R^2 = 0.96$; the trained head converts a near-constant 0.72 to 0.75 of the available start value at every budget, a rate that transfers unchanged to prize-collecting routing; and within the measured range the near-zero-benefit end of the spectrum is reached only through problem structure, not through budget slack. One linear head restores first-step credit and improves low-budget decoding on OP50 without degrading full-budget performance. The randomized-start control, the near-optimal start-set analysis, and the POMO+ reimplementation following its published description provide convergent evidence for this mechanism. The repair's advantage is bounded by its training distribution, and the calibration instrument separates what that means: off-distribution the start value the stakes measure persists, while the trained head's conversion of it halves under moderate shift and reverses under compound shift, a boundary we measure and report. Making the conversion, not the measurement, robust to that shift is the natural next step.

### Reproducibility and code availability

The supplementary material contains the training and evaluation code for all configurations including the POMO+ reimplementation, the local-search instruments (near-optimal start-set extraction, the start-stakes measure and its PCTSP mirror, the constructor baseline), the budget-sweep and shift-study drivers, the per-run outputs behind every table and figure, and a README that maps each reported number to the script and command that produced it. Test sets are the canonical public benchmarks, referenced by checksum. The code will be released publicly upon publication.

[4]The COMPASS method of Chalumeau et al. (2023) is unrelated to the classical OP solver Compass (Kobeaga et al., 2018) used as our reference.

### Broader Impact Statement

This work analyzes and repairs a training bias of neural combinatorial optimization solvers and characterizes when the repair is worth deploying. Its foreseeable applications are efficiency improvements in routing and logistics planning. We do not identify societal risks specific to this method beyond those of route optimization generally; the calibration methodology may help practitioners avoid spending compute on repairs that a cheap measurement indicates would provide negligible benefit.

## References

Federico Berto, Chuanbo Hua, Junyoung Park, Laurin Luttmann, Yining Ma, Fanchen Bu, Jiarui Wang, Haoran Ye, Minsu Kim, Sanghyeok Choi, Nayeli Gast Zepeda, André Hottung, Jianan Zhou, Jieyi Bi, Yu Hu, Fei Liu, Hyeonah Kim, Jiwoo Son, Haeyeon Kim, Davide Angioni, Wouter Kool, Zhiguang Cao, Qingfu Zhang, Joungho Kim, Jie Zhang, Kijung Shin, Cathy Wu, Sungsoo Ahn, Guojie Song, Changhyun Kwon, Kevin Tierney, Lin Xie, and Jinkyoo Park. RL4CO: an extensive reinforcement learning for combinatorial optimization benchmark. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2025.

Felix Chalumeau, Shikha Surana, Clément Bonnet, Nathan Grinsztajn, Arnu Pretorius, Alexandre Laterre, and Thomas D. Barrett. Combinatorial optimization with policy adaptation using latent space search. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2311.13569.

Jinho Choo, Yeong-Dae Kwon, Jihoon Kim, Jeongwoo Jae, André Hottung, Kevin Tierney, and Youngjune Gwon. Simulation-guided beam search for neural combinatorial optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. arXiv:2207.06190.

Darko Drakulic, Sofia Michel, Florian Mai, Arnaud Sors, and Jean-Marc Andreoli. BQ-NCO: Bisimulation quotienting for efficient neural combinatorial optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2301.03313.

Nathan Grinsztajn, Daniel Furelos-Blanco, Shikha Surana, Clément Bonnet, and Thomas D. Barrett. Winner takes it all: Training performant RL populations for combinatorial optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2210.03475.

André Hottung, Yeong-Dae Kwon, and Kevin Tierney. Efficient active search for combinatorial optimization problems. In *International Conference on Learning Representations (ICLR)*, 2022. arXiv:2106.05126.

André Hottung, Mridul Mahajan, and Kevin Tierney. PolyNet: Learning diverse solution strategies for neural combinatorial optimization. In *International Conference on Learning Representations (ICLR)*, 2025.

Szymon Jakubicz, Karol Kuźniak, Jan Wawszczak, and Paweł Gora. POMO+: Leveraging starting nodes in POMO for solving capacitated vehicle routing problem, 2025. arXiv:2508.08493.

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

Gorka Kobeaga, María Merino, and Jose A. Lozano. An efficient evolutionary algorithm for the orienteering problem. *Computers & Operations Research*, 90:42–59, 2018. doi: 10.1016/j.cor.2017.09.003.

Gorka Kobeaga, Jairo Rojas-Delgado, María Merino, and Jose A. Lozano. A revisited branch-and-cut algorithm for large-scale orienteering problems. *European Journal of Operational Research*, 313(1):44–68, 2024. doi: 10.1016/j.ejor.2023.07.034.

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

Wouter Kool, Herke van Hoof, and Max Welling. Stochastic beams and where to find them: The Gumbel-top-$k$ trick for sampling sequences without replacement. In *International Conference on Machine Learning (ICML)*, 2019b. arXiv:1903.06059.

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 (NeurIPS)*, 2020.

Pieter Vansteenwegen, Wouter Souffriau, and Dirk Van Oudheusden. The orienteering problem: A survey. *European Journal of Operational Research*, 209(1):1–10, 2011. doi: 10.1016/j.ejor.2010.03.045.

Liang Xin, Wen Song, Zhiguang Cao, and Jie Zhang. Multi-decoder attention model with embedding glimpse for solving vehicle routing problems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

Shunyu Yao, Xi Lin, Jiashu Wang, Qingfu Zhang, and Zhenkun Wang. Rethinking supervised learning-based neural combinatorial optimization for routing problem. *ACM Transactions on Evolutionary Learning and Optimization*, 5(4), 2025. doi: 10.1145/3694690.

## A Evaluation protocol

**Test data provenance.** The OP50 test set is rebuilt bit-exactly from the official generation script of Kool et al. (2019a), fetched verbatim and run with test seed 1234, producing 10,000 instances (distance-based prizes, $T$=3). We pin the file by checksum (`md5 47a828139a9fcf1e642a060f40837d9e`) so that every number in this paper refers to one immutable dataset, the same one the published Compass reference of 16.17 anchors to.

**Gap definition.** Gap $= (P_{\text{ref}} - P_{\text{model}})/P_{\text{ref}}$, where $P$ is the mean collected prize over the test set and $P_{\text{ref}} = 16.17$ (Compass). For PCTSP, a minimization problem, gap $= (C_{\text{model}} - C_{\text{ref}})/C_{\text{ref}}$ with $C_{\text{ref}} = 4.48$ (OR-Tools). This is the aggregate-mean convention used by the benchmark literature on this set; we verified the convention by reproducing a published gap from its printed prize values. Supplementary statistics (per-instance distributions, paired Wilcoxon tests, per-seed rows) use the same underlying prizes and are never mixed into the headline convention.

**Budget accounting.** Each decoding scheme is charged its exact rollout count per instance: greedy and greedy-head cost 1; multistart with $N$ starts costs $N$; sampling costs the number of samples (1280); instance augmentation costs the number of augmentations (1280); multistart combined with 16 symmetric augmentations costs $16N$ (800 at $N$=50). Comparisons are made scheme against scheme at equal budget only.

**Multistart evaluation path.** All multistart-family evaluations call the policy directly with the target environment passed explicitly, and best-of-$N$ reduction is done over an explicit start-major layout. This rule exists because the library's evaluation utilities, when called without an environment, silently instantiate a default one sized for 20 nodes and take start indices modulo 20, so on 50-node instances multistart with 25 and with 50 starts evaluate the same 20 starts. A consistency check (bit-identical results between the two schemes) exposed this before any of the three configurations was evaluated; the error direction would have underestimated the unmodified baseline's multistart performance, biasing the comparison in our favor.

**Published anchors.** AM values in Table 1 are the published numbers of Kool et al. (2019a) on the same test set and reference; AM samples its first step from the policy and never had the forced-start bias. MDAM gaps are recomputed from its published objectives (15.92 greedy, 16.21 beam 50) against the same Compass reference; MDAM's greedy decodes once per each of its five decoders, so its cheapest decode costs five rollouts.

**Wall-clock costs.** Table 5 reports two kinds of measured cost, which answer different questions and are labeled separately. *Batched throughput* (GPU rows, single RTX 4090, full 10,000-instance set at protocol batch sizes) answers how much a large offline evaluation costs; it amortizes fixed per-forward overheads and must not be read as deployment latency. *Single-instance latency* (batch size one, mean over 100 instances

after warmup, measured on a single RTX A4000 and one core of its host CPU) answers what one decision costs in a planning loop: one SAS one-shot decode takes 10.3 ms on the GPU and 6.1 ms on one CPU core (5.5 ms with four threads), and a batch-one multistart-50 decode 14.5 ms. At that latency class the classical comparison point is the constructor of Section 3.3 at 6.7 ms; one restart of the full ILS instrument costs 0.34 s on one EPYC 7352 core. Compass is cited as published (five minutes for the same set (Berto et al., 2025), about 30 ms per instance on different hardware; we make no cross-hardware comparison beyond orders of magnitude). The start head itself adds no measurable inference cost (POMO and SAS throughput rows are equal).

Table 5: Measured decoding cost per instance, OP50. Throughput rows amortize batching over 10,000 instances; latency rows decode one instance at a time (batch 1, mean of 100). ILS = the instrument solver of Appendix E; constructor = its construction stage only.

| decode | rollouts | cost / instance |
|---|---|---|
| *batched throughput (GPU, batch $\geq$ 200)* | | |
| POMO greedy | 1 | 0.051 ms |
| SAS greedy-head | 1 | 0.053 ms |
| SAS multistart 50 | 50 | 0.41 ms |
| SAS multistart $\times$ aug | 800 | 4.2 ms |
| *single-instance latency (batch 1)* | | |
| SAS greedy-head, GPU | 1 | 10.3 ms |
| SAS greedy-head, 1 CPU core | 1 | 6.1 ms |
| SAS multistart 50, GPU | 50 | 14.5 ms |
| constructor best-of-1 (1 CPU core) | — | 6.7 ms |
| constructor best-of-16 (1 CPU core) | — | 152 ms |
| Compass (published, other hardware) | — | $\sim$30 ms |
| ILS, 1 restart (1 CPU core) | — | 0.34 s |
| ILS, 16 restarts (1 CPU core) | — | 5.5 s |

**Hardware and determinism.** All OP training and all OP50 evaluations ran on a single NVIDIA RTX 4090, where the wall-clock rows above were also measured. The zero-shot OP100 and OPLib evaluations ran on a single NVIDIA RTX A4000 with reduced batch sizes, the PCTSP50 training and evaluations on a single NVIDIA RTX A5000, and the near-optimal start-set extractions of Appendix E are CPU-only. Evaluation batch shapes are fixed per scheme and recorded with every output, since batch-shape dependent kernel numerics can shift greedy decoding results; greedy schemes are deterministic given the checkpoint, and sampling schemes record their seeds.

**Budget-curve statistics.** At one rollout the median paired improvement of SAS over the uniform control is +6.4 to +6.6 points per seed (IQR roughly 1.9 to 12.5), with about 85% of instances improved against 14% worsened; the paired Wilcoxon signed-rank statistic is $W \approx 4.6 \times 10^7$ at one rollout, with $p < 10^{-6}$ for every seed at every budget up to twenty-five rollouts. Per-seed medians are +1.24 to +1.30 at ten rollouts and +0.31 at twenty-five. Figure 3 plots the full curve.

## B Training details

Every configuration trains with the official POMO recipe for OP50, unchanged: 100 epochs of 160,000 sampled instances, batch 64, Adam with learning rate $10^{-4}$ and weight decay $10^{-6}$, learning-rate decay by 0.1 at epochs 80 and 95, embedding dimension 128, mixed 16-bit precision. The baseline configuration uses the recipe's seed; the uniform and SAS configurations train three seeds each (42, 43, 44). SAS adds the start head (one linear layer, 128 parameters), the Gumbel-top-$k$ start sampler with the fixed temperature schedule stated in Section 2.3, and the credit term of Eq. (4); the head path is computed in float32 inside the

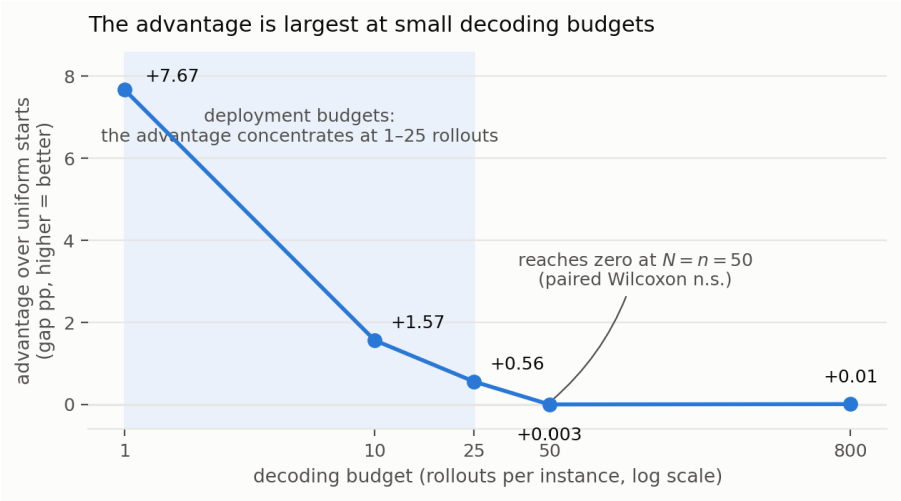


Figure 3: Advantage of SAS over the uniform control (gap percentage points, mean ± std over three seed pairs) versus decoding budget. The advantage decays from $+7.666 \pm 0.088$ at one rollout to zero at $N{=}n{=}50$ (paired Wilcoxon not significant) and stays at zero through the full 800-rollout scheme. Error bars are smaller than the markers beyond $N{=}1$.

mixed-precision forward, because the Gumbel sampler's noise clamp underflows to zero in 16-bit precision and $-\log(u)$ then produces infinities. Nothing else differs between configurations.

**The exhaustive-start regime of the reported runs.** On the synthetic training sets every start is budget-feasible (on the unit square the depot round trip is at most $2\sqrt{2} < T$), and training uses $N{=}n$, so Gumbel-top-$k$ selects all $n$ starts in every batch and the start set is exhaustive rather than sampled. In that regime the realized returns do not depend on the head, and Eq. (4) reduces to an advantage-weighted likelihood of every start under $\pi_{\text{head}}$: each of the $n$ starts pushes its own log-probability up or down by its advantage in every batch. This is a denser training signal for the head than the sampled form, and it is consistent with the sharp concentration the trained head exhibits (Section 3.4).

**Head-local credit, quantified.** In our first implementation the credit term backpropagated through the encoder embeddings and degraded the whole policy, raising the full-budget gap to 3.21% against the baseline's 0.009%; the start-credit signal, orders of magnitude sparser than the per-step objective, is not a useful encoder training signal. With the stop-gradient of Section 2.3, training curves are indistinguishable from plain POMO.

## C PCTSP: penalty sensitivity, protocol, and results

**Data and reference.** The PCTSP50 test set is rebuilt bit-exactly from the official generation script of Kool et al. (2019a) with test seed 1234 (10,000 instances, checksum-pinned as for OP). Gaps are relative to the OR-Tools reference cost of 4.48 on this set (Berto et al., 2025); rewards were cross-checked at load time by recomputing tour cost plus unvisited penalties from raw coordinates.

**Penalty sensitivity.** PCTSP's selection pressure rests on one published design choice, the penalty factor of 3 (Kool et al., 2019a), with no sensitivity study in the literature. We therefore measured, over the factor grid $[0.5, 8]$ with two calibration configurations, the two properties that pre-existing statements describe: the design intent of visiting about half the nodes, and the source paper's own liveness claim that the prize constraint is "sometimes binding and sometimes not". Both hold at the standard setting with margin: visit rate 0.54 at factor 3, quota binding on 33% of instances, and visit-all saturation only beyond the grid.

**Results against the OP protocol.** Every OP comparison was translated line by line to PCTSP's scale. The paired SAS–uniform comparisons at one-shot, ten, and twenty-five rollouts are significant with per-seed paired Wilcoxon $p < 10^{-6}$. At the full budget SAS reaches 1.02%, below the 1.51% comparison value (the baseline's own full-budget gap plus the same half-point margin used on OP). The negative result: SAS's one-shot decode, at 4.68% ± 0.15, does not reach the 2.46% AM greedy of the benchmark re-implementation (Berto et al., 2025), nor AM's originally published 2.66% on the same reference (Kool et al., 2019a). The near-optimal start-set instrument is the same solver as the sensitivity study above, run under Appendix E conventions with the minimization mirror of the near-optimal rule (objective within 0.5% of the best restart).

Table 6: PCTSP50, all schemes (gap % to OR-Tools cost 4.48; mean ± std over 3 seeds where applicable).

| decoding scheme (rollouts) | enumerated (POMO) | uniform | SAS (ours) |
|---|---|---|---|
| one-shot via start distribution (1) | — | $11.900 \pm 0.040$ | $4.679 \pm 0.153$ |
| multistart 10 | 4.189 | $4.206 \pm 0.022$ | $2.351 \pm 0.056$ |
| multistart 25 | 2.841 | $2.852 \pm 0.008$ | $2.178 \pm 0.031$ |
| multistart 50 ($= n$) | 2.141 | $2.160 \pm 0.007$ | $2.158 \pm 0.031$ |
| multistart 50 × aug 16 (800) | 1.011 | $1.025 \pm 0.004$ | $1.024 \pm 0.029$ |
| one-shot, plain greedy (1) | 9.241 | $10.250 \pm 0.281$ | $10.589 \pm 1.159$ |
| sampling 1280 | 0.823 | $0.817 \pm 0.019$ | $0.823 \pm 0.020$ |
| aug 1280 | 2.507 | $3.227 \pm 0.527$ | $2.858 \pm 0.051$ |

## D The must-visit control: setup and full numbers

SAS was transplanted unchanged to TSP50 and CVRP50: the official POMO recipe per problem, the ITER-1 head configuration, and no tuning of any kind (on TSP, where there is no depot, node 0 is an ordinary city and is not masked from the start distribution; this is an environment semantics fix, not a parameter). Configurations: plain POMO (recipe seed) and SAS with two seeds per problem. Uniformity is measured on 1,000 freshly sampled instances (head softmax over feasible first actions): mean entropy against $\ln n$, KL to the uniform distribution, and mean top-1 probability. Parity is measured on a fixed 10,000-instance set per problem as mean decoding cost, reported as the relative difference to plain POMO. We expected the head to flatten under symmetry; it did not, and Section 4.1 reports the finding.

Table 7: Must-visit control, full numbers. Uniformity: entropy ratio $H/\ln n$, KL to uniform (nats), mean top-1 (uniform = 0.02). Parity: relative cost difference to plain POMO in points (bands: ±0.5 one-shot, ±0.2 multistart). Column heads: gh = greedy-head one-shot; ms10/ms50 = multistart with 10/50 rollouts; msaug = multistart with 16 augmentations.

| problem | seed | $H/\ln n$ | KL | top-1 | gh | ms10 | ms50 | msaug |
|---|---|---|---|---|---|---|---|---|
| TSP50 | 42 | 0.017 | 3.85 | 0.972 | −0.195 | +0.083 | +0.006 | +0.001 |
| TSP50 | 43 | 0.015 | 3.85 | 0.976 | −0.178 | +0.012 | −0.015 | 0.000 |
| CVRP50 | 42 | 0.004 | 3.90 | 0.994 | +0.263 | −0.059 | +0.006 | −0.004 |
| CVRP50 | 43 | 0.005 | 3.89 | 0.991 | +0.314 | −0.024 | +0.022 | +0.001 |

The one-shot residuals show a small asymmetry: on TSP the learned one-shot is marginally better than plain greedy, on CVRP marginally worse, with the sign flip consistent with CVRP's fixed depot and mildly asymmetric customers; both sit well inside the ±0.5 band. The persistence of concentration under exact symmetry is itself consistent with the exhaustive credit signal of Section 2.3: nothing in the objective pushes the head toward uniform, and accumulated advantage noise is enough for softmax dynamics to settle on an arbitrary winner.

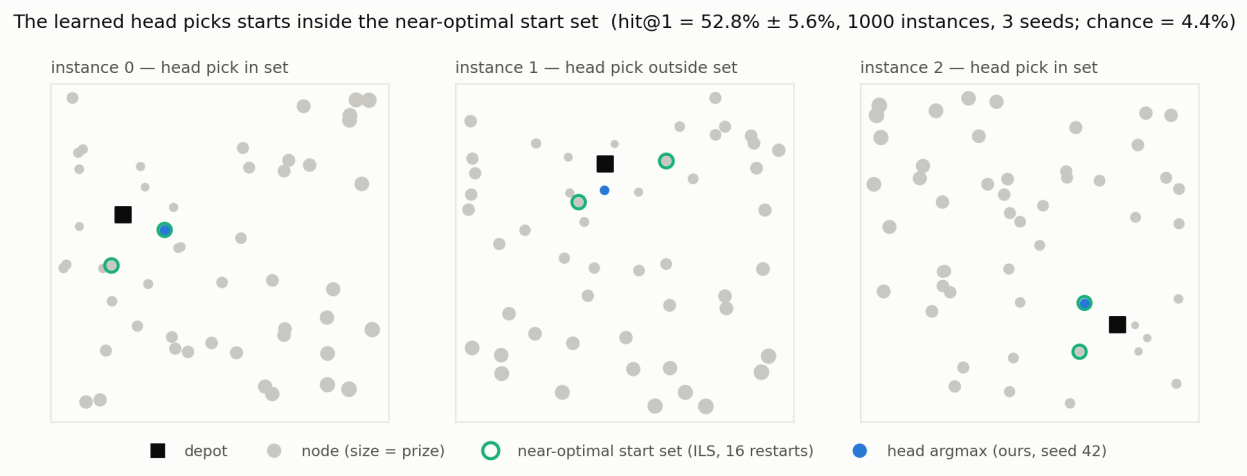


Figure 4: Head start distribution versus the near-optimal start set (first three test instances, fixed selection, seed 42). Rings mark the near-optimal start set extracted from ILS solutions; the filled marker is the head's argmax. Aggregate over 1,000 instances: hit@1 = 52.8% ± 5.6% versus 4.4% chance, a lower bound since the ILS samples the near-optimal space.

## E The near-optimal start-set instrument

Every hit-rate number in the paper is measured against near-optimal start sets extracted by an independent local-search solver, specified here. Construction is randomized greedy insertion under the score $\rho_i^{\gamma_p}/\text{detour}_i^{\gamma_d}$, run once for each of five fixed criterion shapes $(\gamma_p, \gamma_d) \in \{(1,1), (0.5,1), (2,1), (4,0.5), (1,0.2)\}$, keeping the best. Descent alternates tour compression (2-opt and Or-opt), greedy insertion of further nodes, and prize-improving 1-1 swaps to a fixed point. Iterated local search then runs 160 perturbation rounds that cycle small random or contiguous drops (10–30% of visited nodes), targeted eviction of the worst prize-to-saving nodes, large drops (40–60%), and a fresh reconstruction every sixth round, with an acceptance rule that records the elite solution and otherwise walks within 2% of it. Each instance receives 16 restarts with fixed per-restart seeds; the near-optimal start set is the union of the two depot-adjacent nodes over all restarts whose prize reaches 99.5% of the best restart. The implementation is pure NumPy on CPU. Because 16 restarts sample the near-optimal solution space, all reported hit rates are lower bounds. Figure 4 shows the head's start distribution against these sets on the first three test instances.

## F Full results

Table 8 extends Table 1 with the sampling and augmentation-only schemes. Tables 9, 10 and 11 give the complete zero-shot numbers behind Section 4.3. Per-seed rows and raw per-instance outputs for every cell ship with the code release.

The bottom block gives orthogonal budget mechanisms on the baseline checkpoint at matched budgets: sampling and beam search spend rollouts without choosing starts, and both trail learned starts at every budget in the deployment range.

**OP100 protocol.** Zero-shot: OP50-trained checkpoints, unchanged, on the canonical OP100 test set (10,000 instances, seed 1234, checksum-pinned); gaps relative to the published Compass value of 33.19 on this set, from the same source table as the OP50 anchor. Because $n \neq 50$, both multistart at 50 starts and at $N$=$n$=100 are reported; no selection is made between them.

**OPLib protocol.** Subset rule: all gen1–3 instances with Euclidean (EUC_2D) edge weights, 210 of 258; the excluded 48 use other TSPLIB metrics. Instances are normalized for model input (bounding-box translation, longest-side scaling, prizes divided by the maximum score, budget scaled by the coordinate factor);

Table 8: All decoding schemes on OP50 (gap % to Compass, 10,000 instances; mean ± std over 3 seeds where applicable).

| decoding scheme (rollouts) | enumerated (POMO) | uniform | SAS (ours) |
|---|---|---|---|
| one-shot via start distribution (1) | — | $11.532 \pm 0.080$ | $3.866 \pm 0.021$ |
| multistart 10 | 2.875 | $2.858 \pm 0.028$ | $1.293 \pm 0.024$ |
| multistart 25 | 1.669 | $1.652 \pm 0.026$ | $1.094 \pm 0.019$ |
| multistart 50 ($= n$) | 1.096 | $1.081 \pm 0.017$ | $1.078 \pm 0.022$ |
| multistart 50 × aug 16 (800) | 0.009 | $0.025 \pm 0.020$ | $0.014 \pm 0.016$ |
| one-shot, plain greedy (1) | 12.438 | $12.862 \pm 1.499$ | $13.244 \pm 2.176$ |
| sampling 1280 | 0.149 | $0.145 \pm 0.012$ | $0.134 \pm 0.014$ |
| aug 1280 | 3.806 | $4.840 \pm 1.081$ | $4.752 \pm 1.048$ |
| sampling 1 | 7.473 | — | — |
| sampling 10 | 3.041 | — | — |
| sampling 25 | 1.753 | — | — |
| beam search 10 | 2.939 | — | — |
| beam search 25 | 1.674 | — | — |

Table 9: OP100 zero-shot, all schemes (gap % to Compass 33.19; mean ± std over 3 seeds where applicable).

| decoding scheme (rollouts) | enumerated (POMO) | uniform | SAS |
|---|---|---|---|
| one-shot via start distribution (1) | — | $17.502 \pm 0.416$ | $13.960 \pm 1.032$ |
| multistart 10 | 9.767 | $9.382 \pm 0.296$ | $8.776 \pm 0.609$ |
| multistart 25 | 8.544 | $8.153 \pm 0.235$ | $7.865 \pm 0.497$ |
| multistart 50 | 7.941 | $7.544 \pm 0.222$ | $7.382 \pm 0.380$ |
| multistart 100 ($= n$) | 7.504 | $7.105 \pm 0.203$ | $7.174 \pm 0.353$ |
| multistart 100 × aug 16 (1600) | 4.488 | $4.199 \pm 0.101$ | $4.241 \pm 0.249$ |
| one-shot, plain greedy (1) | 19.693 | $19.169 \pm 0.899$ | $19.475 \pm 0.982$ |

every decoded tour is then mapped back to the original coordinates and re-validated under the TSPLIB integer metric, and rollouts exceeding the integer budget are discarded. An instance whose every rollout is discarded scores zero collected prize; no instance is excluded from any average. Discard rates are close across configurations (one-shot: 11.0% for POMO, $9.4 \pm 2.7$% for uniform, $9.5 \pm 1.9$% for SAS; at fifty rollouts: 12.9%, $10.2 \pm 0.9$%, $10.3 \pm 0.9$%), so the configuration ordering of Table 10 is not an artifact of differential validity. One further caveat: the enumerated configuration beats the uniform control at every OPLib budget (49.17 versus 51.22 at fifty rollouts), which suggests OPLib's file-order node numbering is geographically structured; on this benchmark the enumeration is therefore not fully blind, and we note this rather than correct it. Collectable score excludes the depot. Gaps are relative to the per-instance EA4OP solution values vendored in the OPLib repository (Kobeaga et al., 2018), a heuristic best-known reference rather than certified optima.

The library's solution-length assertion uses a $10^{-5}$ float tolerance that these fractional budgets can exceed by rounding; we relaxed it to $10^{-4}$ for this evaluation only. Of roughly 126,000 rollouts, 520 fell in the relaxed band, the integer re-validation discarded 276 of them, and none exceeded the relaxed bound. The instance budgets fix the budget-to-tour-length ratio at $\alpha = 0.5$, the hardest selection regime in the OP literature and the same regime as the synthetic training budgets.

**Axis separation on OPLib.** On the prize scheme closest to training (distance-based, gen3) the head's hit rate is 31%, so real geometry alone roughly halves it; on the two shifted prize schemes it is 4% and 0%. Cross-seed agreement of the heads' top-10 starts collapses from Jaccard 0.87 in-distribution to 0.47.

Table 10: OPLib zero-shot, mean gap % to the EA4OP reference over 210 instances (mean ± std over 3 seeds). One-shot = greedy-head for configurations with a start distribution, plain greedy for POMO.

| scheme | enumerated (POMO) | uniform | SAS |
|---|---|---|---|
| one-shot (1) | 61.09 | $63.42 \pm 1.06$ | $64.26 \pm 3.02$ |
| multistart 10 | 52.30 | $54.06 \pm 0.31$ | $56.63 \pm 3.67$ |
| multistart 25 | 49.91 | $52.18 \pm 0.60$ | $54.56 \pm 3.96$ |
| multistart 50 | 49.17 | $51.22 \pm 0.57$ | $53.52 \pm 4.06$ |

Table 11: OPLib multistart-50 mean gap % by prize generation and by instance size (mean ± std over 3 seeds). gen3 prizes are distance-based like the training distribution; gen1 are constant, gen2 pseudo-random.

| bucket | instances | enumerated (POMO) | uniform | SAS |
|---|---|---|---|---|
| gen1 | 70 | 55.84 | $58.27 \pm 0.57$ | $61.96 \pm 6.95$ |
| gen2 | 70 | 52.94 | $55.56 \pm 1.45$ | $55.70 \pm 3.50$ |
| gen3 | 70 | 38.74 | $39.82 \pm 1.22$ | $42.90 \pm 3.61$ |
| $n \leq 100$ | 36 | 15.24 | $18.40 \pm 0.82$ | $20.28 \pm 2.00$ |
| $101 \leq n \leq 300$ | 66 | 33.37 | $36.35 \pm 0.13$ | $38.87 \pm 2.81$ |
| $301 \leq n \leq 1000$ | 36 | 55.33 | $58.46 \pm 1.40$ | $62.23 \pm 4.42$ |
| $n > 1000$ | 72 | 77.56 | $77.63 \pm 2.56$ | $79.21 \pm 6.44$ |

## G The budget-sweep calibration: protocol

**Budgets and training.** The five budgets were chosen by a calibration pass that measured only the near-optimal visit fraction as a function of $T$ (64 instances, grid $T \in [1.5, 6.0]$) and selected $\{1.5, 2.0, 4.0, 4.5\}$ around the canonical $T$=3 to span visit fractions 24% to 81%. Each non-canonical budget trains SAS with the identical official recipe and three seeds (42/43/44); the canonical budget reuses the main-paper checkpoints. The comparison thresholds, the two predictors, and the interpretation criteria are stated in full in this appendix.

**Outcome conventions.** All outcome quantities are measured on the first 1,000 canonical test instances with the budget field substituted (base file checksum-pinned as in Appendix A). One-shot decodes: learned start (greedy-head), uniform start (the same checkpoint decoding from a uniformly random feasible start; mean over eight fixed draws), and oracle (best one-shot over all fifty starts). The per-budget reference is the mean best prize of the Appendix E solver (16 restarts) on the same instances; realized benefit and oracle value are expressed in percentage points of it. This baseline convention differs from the three-configuration table, which compares against a separately trained uniform configuration: at the canonical budget the two conventions give $7.78 \pm 0.06$ and $7.67 \pm 0.09$ points respectively, distinct estimands whose closeness is consistent with the near-equivalence of the two backbones at full budget (Table 1).

**The start-stakes measure.** For each instance, eight forced start nodes are drawn uniformly without replacement from the feasible set $\{s : 2\,d(0, s) \leq T\}$. A lightweight iterated local search (greedy insertion, 2-opt, drop and reinsertion rounds; the forced variant keeps the start node pinned through every move) estimates the best prize achievable from each forced start and, with the same procedure and restart budget, the unconstrained best. The stakes of an instance are $1 - \overline{P}_{\text{forced}}/P_{\text{ref}}$ with the denominator taken from the stronger Appendix E solver as a common anchor for both sides of the ratio (values clipped to $[0, 1]$; zero clipping events occurred), and a budget's stakes are the mean over 200 instances. Using one solver for both sides cancels its quality bias inside the ratio; the residual drift of the light solver against the anchor spans 3.5 points across budgets, an order of magnitude smaller than the 17.5-point stakes range it would need to explain.

**Uncertainty.** The five-point OLS of realized benefit on stakes gives slope 71.8 (95% interval $[43, 100]$, $t$-based with three degrees of freedom) and intercept $-2.6$ ($[-7.9, 2.7]$). Bootstrap resampling (10,000 draws) of the three seeds within each budget gives an $R^2$ interval of $[0.950, 0.962]$; resampling the 200 stakes instances per budget gives $[0.933, 0.972]$; resampling both gives $[0.932, 0.972]$. Leave-one-budget-out refits span $R^2 \in [0.945, 0.983]$ and slopes $[62, 79]$. Per-budget stakes standard errors are 0.0017 to 0.0050. The conversion ratio pooled over the fifteen budget-seed cells is $0.734 \pm 0.016$ (95% CI $[0.725, 0.744]$), all cells in $[0.706, 0.762]$. The companion regression of the oracle start value on the stakes gives $R^2 = 0.958$, slope 94.4, intercept $-3.1$; the ratio of the realized to the oracle slope is 0.761, inside the per-cell conversion range, which is the slope-level statement of the factorization in the main text.

**Robustness to the shared-reference form.** Both the headline outcome and the stakes divide by a local-search quantity. Two sensitivity fits, added in revision, address the resulting ratio-correlation concern. With the benefit normalized by the checkpoint's own uniform-start prize (no local-search quantity in the outcome), the regression gives $R^2 = 0.94$, slope direction and rank correlation unchanged. With no normalization (raw prize units), $R^2 = 0.04$: absolute prize scales are incommensurable across budgets, so the calibration is a statement about relative value.

**Cross-problem and shifted-set instrument runs.** The three overlay points of Figure 2 use the identical instrument code and parameters with fresh, recorded seed bases. PCTSP50 uses the minimization mirror: stakes $= \overline{C}_{\text{forced}}/C_{\text{ref}} - 1$ (clipped to $[0, 1]$; zero clips), with the same forced-first-node invariant, all nodes feasible as starts, and the per-instance reference from the same 16-restart solver that built PCTSP's near-optimal start sets. OP100 uses the evaluation budget $T$=4 and the first 200 instances; OPLib uses the $n \leq 300$ subset (102 instances) on the normalized geometry the models see. The light solver's quality against its anchor drifts off-distribution (0.96 to 0.99 in-distribution, 0.92 on OP100, 0.91 on OPLib, 1.00 on PCTSP50), which inflates shifted-set stakes by at most a few points and does not affect the qualitative reading, since the shifted points would need stakes near 0.09, not 0.19 to 0.24, to sit on the line. Outcome calibers differ by necessity and are stated: the PCTSP50 realized benefit ($7.17 \pm 0.24$ points) is same-checkpoint, first 200 instances, in points of the 16-restart reference, the sweep's convention; the three-configuration convention on the full test set gives 7.2 points, the same closeness as OP's 7.78/7.67 pair. The OP100 and OPLib realized values are the zero-shot numbers of Table 4 (independently trained uniform control). The available-value check decodes the same sas checkpoints from every start and subtracts the one-shot decode, an oracle-versus-head estimand (its uniform-anchored counterpart is not measurable on those sets without retraining, which this revision does not do).

**The classical constructor baseline.** Best-of-one is the reference solver's construction stage, deterministic prize-per-detour greedy insertion, no descent. Best-of-five is its deterministic five-shape criterion portfolio; best-of-sixteen samples shapes with multiplicative noise 0.15 (seeds recorded). OP50: gaps against the same per-instance ILS reference as the sweep, first 1,000 instances: 30.1%, 14.7%, 11.3%. OPLib $n \leq 300$: constructions run on the normalized instance, tours re-validated under the original integer metric and scored against the EA4OP reference exactly as the neural configurations are, over-budget constructions voided: 27.4%, 19.5%, 17.2% (neural configurations on the same subset: one-shot 42.5% to 46.4%, fifty rollouts 27.0% to 32.3%).

**The hit-rate bound, quantified at its loosest point.** At $T$=2.0 the head's measured hit@1 is $17.8\% \pm 7.1$ against 53% to 63% elsewhere, while its benefit and conversion ratio are on the calibration curve. Metadata, indexing, and checkpoint assignment were audited and ruled out (index shifts drop the hit rate to the chance level, so no off-by-one exists; every evaluation records its checkpoint, budget, and dataset checksum). Forcing the Appendix E construction to begin at the head's argmax start on 200 instances (a post-hoc probe, designed only after all sweep outcomes were recorded) shows that 25% of the picks counted as misses reach 99.5% of the unconstrained optimum (corrected bound $\geq$ 33%) and the median counted miss reaches 97%; the same probe at the canonical budget serves as control. The under-coverage and the pinned $|S|$ have one cause, the construction bias of the instrument, and the hit rates are therefore reported everywhere as lower bounds rather than corrected.

## H The auxiliary-agent alternative: a POMO+ reimplementation

POMO+ (Jakubicz et al., 2025) is the mechanism closest to ours: it also makes POMO's starting nodes a learned decision, but through an auxiliary agent with its own parameters and training loop rather than through a term of POMO's own objective. It is evaluated in its paper on the must-visit CVRP only, and its code is not released. To compare the two mechanisms where start selection has real consequences, we reimplemented POMO+ from its published description and ran it head-to-head with SAS on OP50 and PCTSP50 under the protocol of this paper. Table 12 summarizes both the conceptual axes and the outcome; Table 13 gives the full budget ladder.

**Fidelity and interpretation decisions.** As specified in the POMO+ paper: the agent is a Multi-Head Attention layer and a feedforward network with residual connections and normalization after each block, followed by a scoring head of linear layers (214,913 parameters at our backbone width); it produces a distribution through Gumbel-Softmax during training and Softmax at inference; encoder embeddings enter it through a stop-gradient; it trains by its own REINFORCE objective against POMO's shared mean baseline, with gradients accumulated over $n$=100 steps per update; it samples $K$=0.2$N$ starts per instance (here $K$=10), each repeated $N/K$ times to fill the $N$ trajectories. Where the paper is silent we fixed the following: dimensions from POMO conventions (8 heads, $d$=128, feedforward width 512), Gumbel-Softmax temperature 1, the main recipe's optimizer and schedule for the agent, the $K$ starts sampled without replacement, feasibility masking identical to SAS (on selective routing an unmasked agent would waste rollouts on infeasible starts), the agent path computed in float32 inside the mixed-precision forward (the same underflow fix as SAS), and start distributions consumed at evaluation by the same budget-ladder rules as every other configuration. Everything else is identical: the official recipe, three seeds, the checksum-pinned test sets, and the budget accounting of Appendix A.

Table 12: Axis comparison with POMO+ (Jakubicz et al., 2025), which also learns start selection for POMO; empirical rows from our reimplementation on selective routing (one-shot gap %, 3 seeds).

| axis | POMO+ | SAS (ours) |
|---|---|---|
| problem class | must-visit CVRP | selective routing (OP, PCTSP) |
| mechanism | auxiliary selection agent | term of POMO's own objective |
| selector parameters | 214,913 | 128 |
| selector training | own loop ($n$=100 accumulation) | one term in the POMO loss |
| one-shot gap, OP50 (port) | $2.60 \pm 0.06$ | $3.87 \pm 0.02$ |
| one-shot gap, PCTSP50 (port) | $3.32 \pm 0.10$ | $4.68 \pm 0.15$ |
| hit@1, OP50 / PCTSP50 | 55% / 70% | 53% / 62% |

Table 13: Head-to-head at matched budgets (gap %; mean ± std over 3 seeds; OP50 to Compass 16.17, PCTSP50 to OR-Tools 4.48).

| | OP50 | | PCTSP50 | |
|---|---|---|---|---|
| scheme (rollouts) | POMO+ (port) | SAS | POMO+ (port) | SAS |
| one-shot (1) | $2.60 \pm 0.06$ | $3.87 \pm 0.02$ | $3.32 \pm 0.10$ | $4.68 \pm 0.15$ |
| multistart 10 | $0.95 \pm 0.02$ | $1.29 \pm 0.02$ | $1.99 \pm 0.21$ | $2.35 \pm 0.06$ |
| multistart 25 | $0.89 \pm 0.02$ | $1.09 \pm 0.02$ | $1.98 \pm 0.21$ | $2.18 \pm 0.03$ |
| multistart 50 ($= n$) | $0.88 \pm 0.02$ | $1.08 \pm 0.02$ | $1.97 \pm 0.21$ | $2.16 \pm 0.03$ |
| multistart 50 × aug 16 (800) | $-0.05 \pm 0.01$ | $0.01 \pm 0.02$ | $0.91 \pm 0.01$ | $1.02 \pm 0.03$ |

**Outcome.** The auxiliary agent reaches lower gaps than the 128-parameter head at every deployment budget on both problems: one-shot 2.60% against 3.87% on OP50 and 3.32% against 4.68% on PCTSP50, with the ten-rollout budget following the same order. Two observations bound the reading. First, the advantage is not purely a selection effect: at the selection-free budgets, where every configuration decodes

the identical start set, the POMO+ backbone is also ahead by about 0.2 points (multistart-$n$ 0.88 against 1.08 on OP50, 1.97 against 2.16 on PCTSP50, and the full augmented budget follows), so part of its one-shot advantage comes from the training-side effect of its rollout distribution, ten distinct starts each sampled five times rather than fifty sampled once, which improves the decoder itself. Second, both selectors aim at the same structure: the agent's argmax start lands in the near-optimal start sets of Appendix E on 55% (OP50) and 70% (PCTSP50) of the first 1,000 instances against chance rates near 4%, alongside SAS's 53% and 62%, while holding a much softer distribution (mean top-1 probability 0.19 to 0.30 against 0.99). The PCTSP family ceiling is unchanged: the port's one-shot 3.32% also does not reach the AM anchor of Section 3.5. Two independent designs profit from repairing the same bias; the one with about 1,700 times the selector parameters, a separate optimizer and accumulation schedule, and its own architectural surface profits more; and both land their selections in the same sparse near-optimal start sets. That is what one expects if the bias is real and the repair works through aimed concentration on those sets. SAS isolates the effect of restoring step-zero credit with one loss term and 128 parameters; the auxiliary agent shows how much additional selector capacity adds on top of it.