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

Yangzhe Kong* *yangzhe.kong@hotmail.com*
Independent Researcher

Yanyuan Mo* *yanyuan.mo@outlook.com*
Independent Researcher

Reviewed on OpenReview: *https://openreview.net/forum?id=KfzwYcSe7I*

Abstract

Multistart decoding in POMO (Policy Optimization with Multiple Optima), as implemented in RL4CO, fixes the start node outside the policy with zero log-probability. On selective routing problems such as orienteering (OP) and prize-collecting TSP (PCTSP), where the solver decides which nodes to visit, this hurts low-budget decoding. Our method, selection-aware starts (SAS), restores the first-step learning signal with a single 128-parameter linear head. With backbone and shared baseline unchanged, the head can also be fitted to frozen POMO checkpoints, with results within seed noise of joint training. On OP50, SAS cuts the one-shot gap (one rollout per instance) from 11.9% for POMO's greedy decode to 3.8% with full-budget performance unchanged, and its advantage over random starts carries over to PCTSP50. A hand-crafted start rule reaches 4.7% on the same POMO checkpoints, recovering most of the gain on OP50, while a reimplemented POMO+ with a start agent about 1,700 times larger reaches a one-shot gap of 2.6% against SAS's 3.9% on the same three seeds. On TSP and CVRP, where starts are interchangeable or nearly so, SAS brings no benefit. To gauge whether the start matters, we introduce a model-free *start-stakes* measure, the share of achievable value lost to a random feasible start, computed before training by classical local search. Its level depends on the solvers used to compute it, and as defined the measure does not track the start value (the gain of the best feasible start over a random one) across instance sizes. Across five OP50 length budgets the head captures 0.72 to 0.75 of the start value, and 0.51 when trained at n=500. Under distribution shift the start value remains while capture falls, and the one-shot advantage halves on OP100 and vanishes on OPLib, where classical construction leads by 15 to 21 points on instances with up to 300 nodes.

1 Introduction

Constructive neural solvers built on the multistart recipe of POMO (Policy Optimization with Multiple Optima) (Kwon et al., 2020) come within a fraction of a percent of optimal on the traveling salesman problem (TSP), where every node must be visited. They are much weaker on selective routing, where the solver must also decide which nodes are worth visiting. On the orienteering problem (OP) and the prize-collecting TSP (PCTSP), the reported gaps of POMO-style models are several times larger than on TSP (Berto et al., 2025). The shortfall is concentrated at low decoding budgets, that is, at few rollouts per instance. Given hundreds of rollouts, a POMO model trained with the official recipe comes close to the classical reference, but routing models that run inside planning loops work at the low end (Section 2.1). The shortfall has been documented repeatedly but not explained at the level of mechanism.

*Equal contribution; the order of the authors is arbitrary.

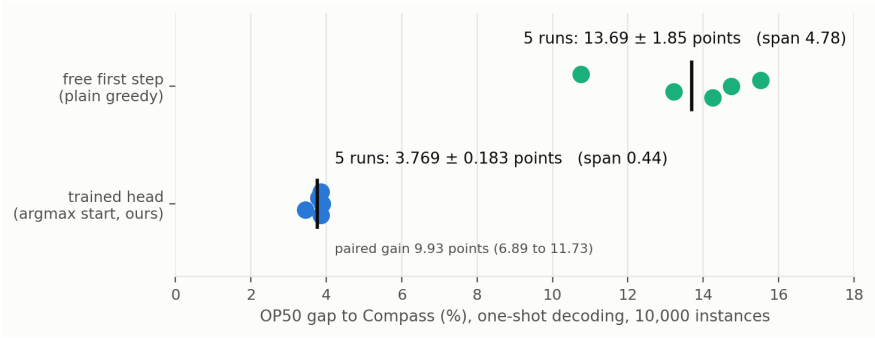


Figure 1: SAS (selection-aware starts) is POMO with our learned start head (Section 2.3); the figure shows the cross-seed spread of one-shot decoding for the same five SAS models. With the head bypassed (a free first step, top row), the one-shot gap spans 4.78 points across seeds; with the trained head (bottom row), it spans 0.44.

We trace the low-budget gap to how multistart decoding handles the first step. POMO starts one rollout from each node, which assumes that every node is an equally good first commitment. When every node must be visited, the assumption is harmless or nearly so. On the TSP every node opens an optimal tour, and an optimal solution of the capacitated vehicle routing problem (CVRP) has several routes, any of whose ends can come first. On selective routing the assumption fails, because choosing the first node already commits the tour to a node that may not be worth visiting. In the POMO recipe as implemented in RL4CO (Berto et al., 2025), the start node is set by a fixed rule and its log-probability is hardcoded to zero, so this decision never enters the policy gradient. On the five trained models of Figure 1, the gap is larger and far more variable across seeds when the first step is the decoder's own untrained argmax (a *free* first step, the only one-shot decode plain POMO has) than when the learned start head of Section 2.3 chooses it.

Our method, selection-aware starts (SAS), changes only this step. A linear head with 128 parameters scores the nodes and picks the starts, and the log-probability of each chosen start enters the policy gradient, which restores the term that POMO's objective drops at step zero (Section 2.3). The backbone and its shared baseline are unchanged, so the head can also be fitted to an already trained POMO checkpoint, with results within seed noise of joint training. On OP50, SAS reduces the one-shot gap from 11.9% for POMO's free first step to 3.8%. The gain shrinks as the decoding budget grows and is zero once every start is rolled out. Most of the gain comes from choosing a sensible start. A hand-crafted start rule already reaches 4.7% on the same POMO checkpoints, and the learned head improves on it by 0.9 points at one rollout and 0.3 at ten. The same mechanism carries over to PCTSP50, although, as on OP50, the one-shot gap remains above that of the Attention Model (AM) (Kool et al., 2019a).

How much SAS helps depends on the problem and the distribution. On the TSP and the CVRP, where starts are interchangeable or nearly so, SAS changes decoding quality by less than a third of a point. To gauge whether the start matters, we use a model-free *start-stakes* measure, the share of achievable value lost when a tour is forced to begin at a random feasible node, computed before any training by classical local search in seconds per instance. It is zero on TSP and between 0.09 and 0.26 on OP. Across five OP50 length budgets it is ordered with the benefit of SAS. Within one distribution the length budget orders the benefit at least as well, and the level of the measure depends on the solvers used to compute it (Sections 4.2 and 4.3). Its link to the start value (the gain of the best feasible start over a random one) holds within one problem and instance size. Across sizes, however, the measure as defined rises while the start value falls (Section 4.4). The share of the start value that a trained head captures stays near its n=50 value at n=100 and falls at n=200 and 500 when the same recipe is trained there, and it falls further under distribution shift, where the start value itself persists (Sections 4.4 and 4.5).

These findings matter for selective-routing solvers built on the POMO recipe, in which the hardcoded first step is part of the recipe itself, for benchmarking with the RL4CO library (Berto et al., 2025), where forced starts are the default in multistart evaluation, and for deployments that must decide whether start

selection is worth adding on their own distribution. The main comparisons use the standard n=50 synthetic benchmarks, on which the baselines and classical references are established, and Section 4.4 trains the same recipe at n=100, 200 and 500. Transfer from synthetic to classical instances remains open for this model family, with zero-shot gaps of 13% to 72% on the OPLib collection (Kobeaga et al., 2018) reported by Yao et al. (2025), and Section 4.5 measures how much of the advantage of SAS remains under such shifts.

We make four contributions. First, we show that the multistart recipe of POMO, as implemented in RL4CO, hardcodes the first step as a symmetry prior that never enters learning. This removes a real decision on selective routing. Second, we return that decision to the policy with SAS, one linear head of 128 parameters that can also be fitted afterwards to a finished POMO checkpoint. In distribution, it substantially improves low-budget decoding and leaves performance at the full decoding budget unchanged. A hand-crafted start rule recovers most of the one-shot gain, and the head improves on it at each measured decoding budget below full coverage. A reimplementation of POMO+ (Jakubicz et al., 2025), whose auxiliary start agent is about 1,700 times larger, reaches lower mean gaps at every decoding budget. It differs from SAS in both selector capacity and training loop, so its margin cannot be attributed to either alone (Appendix I). Third, we characterize when start selection pays off. A model-free start-stakes measure, computed before any training, shows whether the start matters on a problem and orders length budgets by the benefit of SAS within one problem and instance size, although the length budget alone orders that benefit at least as well. We also measure how much of the start value a trained head captures, in distribution, across scale and under distribution shift. Fourth, we locate the limits of SAS. Under zero-shot distribution shift its one-shot advantage halves on OP100, which changes size and length budget, and vanishes on OPLib, which also changes geometry and prizes. On the OPLib instances with up to 300 nodes, one classical greedy construction is both more accurate and faster on one CPU core than the neural one-shot decode, and the lead of SAS over classical construction was established on the training distribution only (Section 4.5).

2 Method

2.1 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 tour that starts and ends at the depot, has total length at most T, and maximizes the total prize of the nodes it visits. The solver makes two coupled decisions, which nodes to visit and in what order. We call such problems *selective routing*, in contrast to must-visit problems such as the TSP, where the set of nodes is fixed. The Prize-Collecting TSP (PCTSP) is selective as well. Its tour must collect at least a minimum total prize, each unvisited node incurs a penalty, and the objective is the tour length plus the penalties (Kool et al., 2019a).

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 one node at a time with a learned policy π_θ. At inference the practical constraint is the *decoding budget*, the number of rollouts that can be afforded per instance; decoding with a single rollout is *one-shot* decoding. Benchmark numbers are usually reported with hundreds of rollouts per instance, whereas a routing model inside a dispatch or re-planning loop, which scores many candidate assignments per decision, often affords one. One-shot decoding of a single OP50 instance takes about 23 ms on one CPU core (Appendix A). Section 3.4 compares the quality of one-shot decoding with that of classical construction of similar or lower cost, timed on one core of the same machine, a comparison that SAS wins on the training distribution and loses on OPLib (Section 4.5).

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 (Williams, 1992),

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

where R_s is the return of trajectory τ_s. On the TSP this design is sound, since by cyclic symmetry every node begins some optimal tour, and forcing N distinct first nodes therefore explores without loss of optimality.

2.2 The first-step bias of multistart decoding

Berto et al. (2025) report the shortfall of this recipe on selective routing as a performance number. We trace it to three places in the reference implementation in RL4CO.[1] We use *bias* for a property of this construction rule. The POMO gradient is unbiased for the policy it trains, but that policy does not include the first step.

Starts outside the policy. Under multistart decoding the first action of rollout s is not sampled from π_θ. It is computed outside the policy, written into the environment, and assigned log-probability zero. The trajectory likelihood therefore starts at step one, and step zero never receives a gradient.

Instance-independent starts. Rollout s starts at node $\big((s-1) \bmod n\big) + 1$, regardless of prizes, length budget or geometry. For OP, when any instance in a batch has fewer feasible starts than rollouts, a guard redraws the starts of the whole batch uniformly *with* replacement from each instance's feasible nodes, so the same start can be drawn repeatedly and rollouts are silently duplicated. For PCTSP there is no guard.

Effect on the shared baseline. Under Eq. (1) a forced start in a region the length budget cannot profitably reach has two effects. The rollout itself is wasted, and its low return lowers the shared baseline b, which inflates the advantage of every sibling rollout, since the baseline is the mean of all N rollouts. SAS (Section 2.3) addresses the first two of the three problems above. It does not remove the third, because in the reported runs every feasible start is still rolled out in training and every rollout still enters the baseline.

2.3 Selection-aware starts

Selection-aware starts (SAS) return the first step to the policy and leave the multistart skeleton, the shared baseline, the encoder, the decoder and the training recipe unchanged. The only change to the loss is that the start head's step-zero log-probability enters the policy gradient.

Start head. A linear layer $w \in \mathbb{R}^d$ scores each node from its encoder embedding through a stop-gradient, $u_i = w^\top \mathrm{sg}(h_i)$, so the head cannot change the encoder (Appendix B records the degradation without the stop-gradient). The depot and the nodes that the length budget makes infeasible are masked to $-\infty$, and the scores define a start distribution $\pi_{\text{head}}(i) = \mathrm{softmax}(u)_i$. The head adds d parameters ($d = 128$ in our models). Below, π_θ is the decoder policy, π_{head} the start distribution (its parameters w are part of θ), and s_s the start of rollout s.

Distinct starts. Whenever N starts are feasible, training draws N distinct starts from π_{head} with Gumbel-top-k (Kool et al., 2019b), which preserves POMO's N distinct trajectories. The head logits are divided by a temperature of 5, then 3, then 1 from the third epoch on. The credit term below uses the same tempered logits, so the schedule scales the head's credit gradients during the first two epochs. With $N{=}n$ and every start feasible, as at the length budgets of the main results, the schedule does not affect which starts are taken.

Restoring step-zero credit. If each start is drawn independently from π_{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,

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

[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 = logprobs.exp() = 1". Start enumeration: `utils/ops.py`, `select_start_nodes`. Shared baseline: `models/rl/reinforce/baselines.py`, `SharedBaseline`. We verified the same structure on the current main branch.

We therefore train with

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

with the same shared baseline b and all N rollouts participating. Under independent draws this reproduces Eq. (3) term by term, and the added term enters at coefficient 1, the value Eq. (3) gives. Gumbel-top-k draws starts without replacement, and for those draws Eq. (4) approximates that gradient.

Training regime of the reported runs. Every start is feasible once $T \geq 2\sqrt{2} \approx 2.83$, the longest depot round trip on the unit square, which covers the canonical T=3. With N=n in training, every start is then rolled out whatever π_{head} is, and the head term of Eq. (4) reduces to an advantage-weighted log-likelihood of every start under π_{head}, which trains the head's *ranking* of starts by return. The ranking is used at decoding time, when the decoding budget is smaller than n. When $T < 2\sqrt{2}$, as at T=1.5 and 2.0 in the sweep of Section 4.2, an instance can have fewer feasible starts m than rollouts N. Training then rolls out each feasible start once and draws the remaining $N - m$ starts with replacement from π_{head}. All N rollouts, repeated starts included, enter the shared baseline and Eq. (4) with weight $1/N$, so a start that occurs r times among the N rollouts contributes r terms, and the loss is not renormalized by m (Appendix B).

In the exhaustive-start regime the change is confined to the head. The three configurations of Section 3.1 roll out the same n starts in training, and because of the stop-gradient the credit term updates only the head's d parameters. The encoder and decoder therefore receive the same gradient as under plain POMO.

2.4 Decoding at inference

SAS decodes at any decoding budget. With N rollouts it takes the head's top-N starts deterministically. Its one-shot decode is one rollout from the head's argmax start, decoded greedily. Plain POMO has no learned counterpart at N=1 and decodes greedily from a free first step, the step it never trained.

3 Results on OP50 and PCTSP50

3.1 Experimental setup

Data and reference. We evaluate on the canonical OP50 test set of Kool et al. (2019a), 10,000 instances with n=50 nodes, distance-based prizes and length budget T=3. All gaps are relative to Compass, the released implementation of the evolutionary algorithm for the orienteering problem (EA4OP) of Kobeaga et al. (2018) and the strongest classical heuristic reported on this set, with an average prize of 16.17.

Configurations. Every model is trained with the official POMO recipe for OP50 as published (Appendix B), and only the start mechanism changes. We train three configurations. *Enumerated* (labeled POMO in Tables 1 and 9 and plain POMO in the text) is unmodified POMO with the forced starts of Section 2.2. *Uniform*, the uniform control, replaces the start head by a uniform distribution over feasible starts and drops the credit term of Eq. (4); its code is otherwise identical to SAS, so it separates learned start selection from merely randomized starts. *SAS* is the full method. Every configuration is trained at five seeds, and means are reported with the standard deviation over those five. Paired comparisons use every seed the compared configurations share. SAS and the uniform control share all five, while the enumerated configuration takes the released recipe seed in place of one of them and so shares four with each. Analyses that cost far more than a decode use three seeds, as noted where they appear; they include the hit-rate analyses built on the near-optimal start-set extraction, our reimplementation of POMO+ (Jakubicz et al., 2025), the length-budget sweep and the scale experiments.

Decoding schemes. We decode every configuration at matched decoding budgets. One rollout is decoded greedily from the configuration's start choice (for SAS the head's argmax start, for the uniform control a uniformly random feasible start) or, for POMO, from a free first step. Multistart uses $N \in \{10, 25, 50\}$ rollouts, and multistart with 16 symmetric augmentations uses $50 \times 16 = 800$, the *full decoding budget*

under which published numbers are produced. Significance is assessed per seed with paired Wilcoxon signed-rank tests over the 10,000 instances, pairing configurations trained with the same seed, and across seeds by whether the two-standard-error interval of the seed-paired mean difference excludes zero.

3.2 Decoding quality across decoding budgets

With a single rollout, plain POMO can only decode from its free first step and reaches an 11.86% gap over five runs (the plain-greedy row of Table 1; 12.4% for the released-recipe run among them). SAS decodes through its head at 3.77%.

The advantage shrinks as the decoding budget grows and disappears once every start is rolled out (Figure 3, Appendix A). Over the uniform control the paired advantage falls from 7.7 points at one rollout to 0.557 ± 0.006 at twenty-five (mean with two standard errors over the shared seeds) and is significant in every seed at one, ten and twenty-five rollouts (paired Wilcoxon $p < 10^{-6}$). At ten and twenty-five rollouts SAS is also significantly ahead of plain POMO in every seed (Table 1). At $N{=}n$ the paired difference SAS minus POMO is -0.030 ± 0.035 at fifty rollouts and $+0.009 \pm 0.024$ at the full decoding budget, both covering zero. Every start is rolled out in training as well, so this result confirms that the change is confined to the head (Section 2.3).

Fitting the head to a trained checkpoint. Because the backbone receives the standard POMO gradient when every start is rolled out in training (Section 2.3), the head can equally be fitted to a finished POMO checkpoint. With each of the five POMO checkpoints frozen and only the 128 head weights trained with the same credit term (Appendix B), the head converges within one epoch of rollouts and decodes within seed noise of joint training (Table 1). Seed-paired against SAS, the differences at one, ten and twenty-five rollouts are $+0.03 \pm 0.17$, $+0.02 \pm 0.07$ and $+0.03 \pm 0.04$ points, all covering zero.

3.3 Sources of the gain

Most of the gain comes from choosing a sensible start. A hand-crafted start rule that starts each instance at the node with the highest prize per unit of distance to the depot reaches $4.73\% \pm 0.15$ on the POMO checkpoints with no training, and 4.70% on the SAS checkpoints. The learned head improves on the rule wherever starts are chosen. Paired on the same SAS checkpoints, the improvement is 0.934 ± 0.034 points at one rollout, 0.296 ± 0.010 at ten and 0.187 ± 0.003 at twenty-five (means with two standard errors over five seeds), with paired Wilcoxon $p < 10^{-90}$ for every seed.

Randomizing the starts or spending the rollouts differently does not close the gap. The uniform control has a start distribution but an untrained one, and its one-shot gap of 11.4% is close to that of plain POMO. Sampling and beam search spend the same rollouts on the POMO checkpoints without choosing starts. In sampling, each rollout starts at a distinct, uniformly random feasible node and samples later steps at temperature 1. Seed-paired, SAS is ahead of both by 0.6 to 1.7 points on average at ten and twenty-five rollouts and by at least 0.54 points in each of the four shared seeds (paired Wilcoxon $p < 10^{-250}$), and both also trail the start rule (Table 1).

Decoded from the free first step (Table 1, "one-shot, plain greedy"), our trained models are no better than a uniformly random feasible start. Across the fifteen trained models the gap ranges from 9.9% to 17.1%.

3.4 Comparison with other methods

Other models with a trained first step are stronger at some decoding budgets (Table 1). AM (Kool et al., 2019a), the sampled-start family that never had the bias this paper repairs, is ahead of SAS at one rollout and behind it with 1,280 samples against SAS's 800 rollouts. MDAM (Xin et al., 2021), which trains five decoders, decodes with beam search at about 250 rollouts and is ahead of SAS at 800 (-0.25% against 0.04%). POMO+, which repairs the same bias with a separately trained start agent of about 215,000 parameters, reaches 2.60% one-shot against 3.87% for SAS on the same three seeds and is ahead at every OP50 decoding budget (Appendix I). Part of that margin comes from a better backbone, since POMO+ also leads by 0.2 points where every start is rolled out.

Table 1: OP50 (gap % to Compass 16.17, 10,000 instances; mean ± std over five seeds per configuration). The configurations and decoding schemes are those of Sections 3.1 to 3.3 and Appendix A. At 50 rollouts every start is taken, so the start rule and the fitted head decode exactly as POMO does. Bold marks the lowest gap in a row and every configuration whose seed-paired difference from it covers zero (mean ± two standard errors), when the marked configurations are significantly better than all others in every seed (Section 3.1). Rows in which every seed-paired difference covers zero (50 and 800 rollouts, plain greedy) and rows with a single entry have no bold entry. AM and multi-decoder AM (MDAM) rows are published numbers on the same test set and reference (conversion in Appendix A). POMO+ is our reimplementation at three seeds, with SAS at the same seeds in brackets.

decoding scheme (rollouts)	POMO	start rule	fitted head	uniform	SAS (ours)
one-shot via start choice (1)	—	4.730 ± 0.147	$\mathbf{3.783 \pm 0.043}$	11.431 ± 0.266	$\mathbf{3.769 \pm 0.183}$
multistart 10	2.825 ± 0.102	1.565 ± 0.082	$\mathbf{1.256 \pm 0.077}$	2.797 ± 0.123	$\mathbf{1.232 \pm 0.127}$
multistart 25	1.629 ± 0.076	1.265 ± 0.076	$\mathbf{1.074 \pm 0.076}$	1.604 ± 0.103	$\mathbf{1.047 \pm 0.105}$
multistart 50 ($= n$)	1.061 ± 0.072	as POMO		1.036 ± 0.101	1.033 ± 0.100
multistart 50 × aug 16 (800)	0.033 ± 0.032	as POMO		0.046 ± 0.046	0.042 ± 0.052
one-shot, plain greedy (1)	11.855 ± 1.198			13.764 ± 2.169	13.695 ± 1.848
sampling 1	11.815 ± 0.140				
sampling 10	2.963 ± 0.100				
sampling 25	1.700 ± 0.074				
beam search 10	2.883 ± 0.107				
beam search 25	1.626 ± 0.084				
POMO+ (reimpl.), three seeds [SAS at the same seeds]: one-shot 2.60 [3.87]; 10 rollouts 0.95 [1.29]; 25 rollouts 0.89 [1.09]					
published, same set (Kool et al., 2019a): AM greedy (1) 3.23; AM sampling (1,280) 0.60					
published, same set (Xin et al., 2021): MDAM greedy (5) 1.55; MDAM beam 50 (≈ 250) −0.25					

Classical construction at low latency. The classical alternative at single-instance latency is the construction stage of the local-search solver of Appendix E. The same prize-per-detour greedy insertion that seeds that solver, run without any descent, reaches a 30.2% gap with one construction, 15.0% with its full five-criterion portfolio and 11.3% at best-of-sixteen with noisy criterion shapes, on the same 10,000 instances and reference as SAS's one-shot 3.77%. Timed per instance without batching on one core of one machine, one construction costs 1.8 ms and sixteen cost 31 ms, while the SAS one-shot decode costs 22.6 ms (Table 6). SAS therefore beats even the best of sixteen constructions, which takes longer and has three times the gap of SAS, and the classical side needs its full local search, at 0.32 seconds per restart on that core, to surpass the neural configuration. On OPLib the ordering of SAS and classical construction reverses (Section 4.5). Compass defines the reference, so its own gap is zero, and its published run time converts to about one second per instance on one thread (Appendix A).

3.5 Starts selected by the head

We test 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 nodes adjacent to the depot in any solution reaching 99.5% of the best found. Each near-optimal tour contributes both of its depot-adjacent nodes, since it can be traversed in either direction. The resulting near-optimal start set S has 2.2 nodes per instance on average and never more than 5, and on 83% of instances S is exactly the two ends of the best tour. The head's argmax start falls inside S (hit@1) on $52.8\% \pm 5.6\%$ 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, this hit rate is likely an underestimate (Figure 4, Appendix E).

The head's softmax is nearly a point mass, with top-1 probability 0.99 on average, which is a property of the training loss. With $N{=}n$ the gradient of the credit term with respect to each start's logit is that start's negative advantage divided by N, whatever the current distribution, so it does not shrink as the head sharpens. A learned temperature floor could keep the head's entropy up if an application wanted calibrated start sampling; we leave this to future work.

Table 2: The three configurations on PCTSP50 (gap % to OR-Tools cost 4.48, 10,000 instances; mean ± std over five seeds per configuration). Same design as Table 1; bold as there. AM reference values: as retrained 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)	—	11.849 ± 0.057	$\mathbf{4.732 \pm 0.148}$
multistart 10	4.201 ± 0.032	4.203 ± 0.023	$\mathbf{2.377 \pm 0.056}$
multistart 25	2.856 ± 0.037	2.846 ± 0.022	$\mathbf{2.182 \pm 0.027}$
multistart 50 ($= n$)	2.151 ± 0.030	2.160 ± 0.015	2.159 ± 0.024
multistart 50 × aug 16 (800)	1.023 ± 0.016	1.024 ± 0.006	1.021 ± 0.020
one-shot, plain greedy (1)	9.745 ± 0.625	10.660 ± 0.856	10.151 ± 1.105
AM greedy (1): 2.46 retrained (Berto et al., 2025); 2.66 original (Kool et al., 2019a)			

3.6 Results on PCTSP50

We apply the three-configuration design of OP50 to PCTSP50 without retuning, with the same official training recipe, the same three configurations at five seeds each, the same decoding budgets, the same comparisons translated to PCTSP's scale, and the same near-optimal start-set extraction, rerun with PCTSP's objective (Appendix C). The start rule, sampling, beam search and the fitted head of Table 1 were run on OP50 only. Gaps are relative to the reference cost obtained with OR-Tools (Perron & Furnon, 2023) on the canonical test set (Berto et al., 2025; Kool et al., 2019a). A sweep over the penalty factor (Appendix C) shows that good solutions at this benchmark's parameters leave nodes unvisited, so the first step is a real decision on PCTSP50 as well.

Agreement with OP50. The three-configuration comparison reproduces (Table 2). The paired advantage over the uniform control decays along a curve like OP's, from $+7.1 \pm 0.1$ points at one rollout to $+0.7$ at twenty-five and zero once every start is rolled out. Up to twenty-five rollouts it is positive in every seed (paired Wilcoxon $p < 10^{-6}$). The near-optimal start-set results reproduce as well. PCTSP's near-optimal start sets are as sparse as OP's (2.1 nodes per instance), the head concentrates as sharply (top-1 probability 0.99), and its argmax falls in the near-optimal set on $62.1\% \pm 6.5\%$ of 1,000 instances (three seeds), about fourteen times the 4.3% chance rate.

Differences from OP50. SAS decodes one-shot at 4.73%, farther above AM's published one-shot greedy gaps than on OP50 (Table 2; Berto et al., 2025; Kool et al., 2019a), and all five seeds lie between 4.50% and 4.91%. POMO-style models lie farther from the classical reference on PCTSP than on OP at every decoding budget in Tables 1 and 2 except the plain-greedy one-shot decode. At the full 800-rollout budget our retrained POMO remains 1.02% from OR-Tools, where its OP counterpart is 0.03% from Compass (Table 1). SAS closes part of the one-shot gap, and the remaining distance is consistent with this family-level shortfall.

4 Dependence on problem, scale and distribution

How much a good start is worth depends on the problem, the length budget, the instance size and the distribution.

4.1 Must-visit control

If the diagnosis is right, SAS should matter only where the first step is a decision. We apply SAS unchanged to TSP50 and CVRP50 (official recipes; two seeds each against one plain-POMO comparator, so the control is qualitative, without an interval estimate). The head still concentrates as sharply as on selective routing (top-1 probability 0.97 to 0.99, against 0.02 for uniform; Appendix D), as the training loss implies (Section 3.5). On must-visit routing the concentration changes decoding quality very little, with SAS within 0.083 points

of plain POMO at every multistart decoding budget on both problems and one-shot differences between -0.195 and $+0.314$ (Table 8, Appendix D). This agrees with the diagnosis, since the bias is present on every problem but costs little where starts are interchangeable.

4.2 Start value and capture across length budgets

Must-visit routing and the canonical OP benchmark lie at two ends of a range. To make the range continuous, 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 of Section 3. Each length budget trains SAS with the unchanged recipe at three seeds. On each checkpoint we decode one-shot from the head's start, from a uniformly random feasible start and from every feasible start, keeping the best result per instance. The *realized benefit* is the gain of the head's start over the random start, so it isolates start selection from all training-side effects, and the *oracle start value*, the start value available to the model, is the gain of the best feasible start over the same random baseline. Both are normalized by a per-budget local-search reference on the same 1,000 test instances (Appendix G). We call their ratio the *capture* κ, the share of the available start value that the trained head captures. In mean one-shot prizes, $\kappa = (P_{\text{SAS}} - P_{\text{unif}})/(P_{\text{oracle}} - P_{\text{unif}})$. All three terms come from decodes of the same checkpoint, so the reference cancels and κ, a ratio of means over a test set, does not depend on the local-search solver. In this sweep the candidate pool is every feasible start, so the head's pick is its one-shot argmax and the oracle is the best feasible start.

The tighter the length budget, the more the start matters. The realized benefit rises from 4.5 points at T=4.5 to 17.4 points at T=1.5, and the oracle start value from 6.3 to 23.1 (Table 3). Capture changes much less. Across the five length budgets κ runs from 0.72 to 0.75 over a nearly fourfold range of benefit, declining monotonically as the budget loosens, by about 0.011 per unit of T.

Loose length budgets. We expected the loosest budget to approach the must-visit regime, with the benefit falling to the small differences the must-visit control exhibits. At an 81% visit fraction, however, the benefit is still 4.5 points, against must-visit differences that stay under a third of a point (Section 4.1). Loosening a selective problem's length budget does not create the rotation symmetry of the TSP, because the tour still starts and ends at the depot and the first hop remains a geometric decision.

Hit rate across the sweep. The hit rates against the near-optimal start sets of Section 3.5 likely underestimate the true rates throughout the paper, and the sweep shows how far below they can fall. The sets are built from the restarts of one local-search procedure whose greedy prize-per-detour construction favors one route structure, and where several routes of nearly equal value exist, the solver reports only its own. The underestimate is largest at T=2.0, where the head's measured hit@1 drops to 18% while its benefit and capture stay in line with the other budgets (Table 3). Forcing a lighter local-search solver to start at the head's chosen node resolves the discrepancy (a separate measurement, Appendix G). On the 200 instances of that measurement, which used one seed with a measured hit@1 of 9%, a quarter of the picks counted as misses reach 99.5% of that solver's unconstrained result, so the corrected rate there is at least 32%, and the median counted miss still reaches 97% of that result. The low hit rate is consistent with near-equivalent routes being most common at T=2.0.

4.3 A model-free start-stakes measure

The start-stakes measure $\sigma = 1 - \overline{P}_{\text{forced}}/P_{\text{ref}}$ is the share of achievable value lost when a tour must begin at a random feasible node (mirrored for minimization problems, Appendix G). Here $\overline{P}_{\text{forced}}$ averages over eight random feasible starts the best prize that a *light solver* (greedy insertion, 2-opt and drop-and-reinsertion rounds, two restarts) reaches with the first node pinned. The reference P_{ref} is the unconstrained best prize of the stronger solver of Appendix E at sixteen restarts. Needing no trained model, σ can be computed on a distribution before training. It is zero on TSP, where the forced and unconstrained optima coincide, and should be near zero on the CVRP.

Ordering within one distribution. Across the five OP50 length budgets the stakes (Table 3) are ordered with the realized benefit (Figure 2). Their rank correlation is 1.0, and a linear fit on the five points reaches

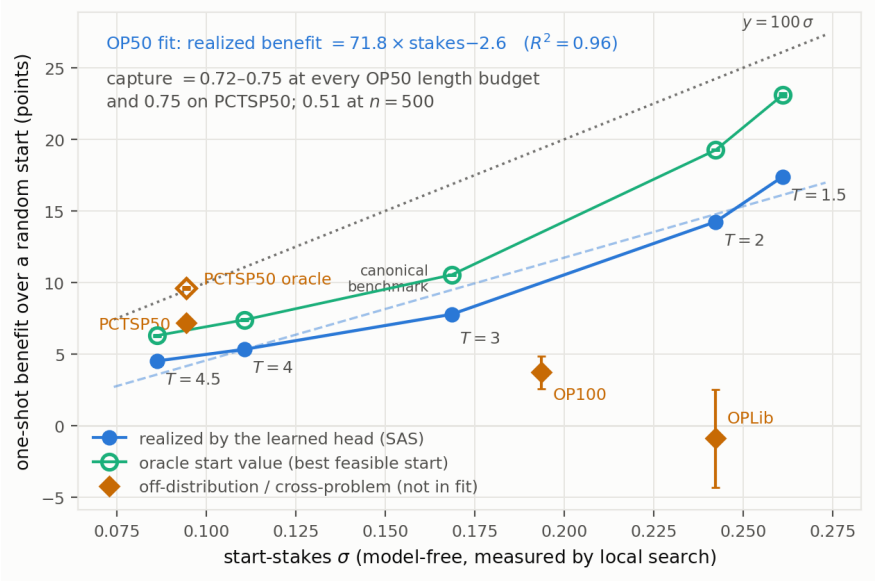


Figure 2: Start-stakes and the realized benefit across five OP50 length budgets. Start-stakes σ (Section 4.3) lie above the oracle start value (rings) at every length budget and are ordered with the realized one-shot benefit of the learned head (blue), with a linear fit over the five budgets. Stakes and both series are shares of the same per-budget reference, so a ring below the dotted $y = 100\,\sigma$ line is a setting where the measure exceeds the oracle start value. Error bars are three-seed standard deviations for the OP50 series, mostly smaller than the markers, and the five-seed standard deviations of Table 5 for OP100 and OPLib. Diamonds are points outside the fit. PCTSP50 lies above the OP50 fit, and its oracle start value (hollow diamond) lies on the dotted line. OP100 and OPLib fall below the fit, but their instance sets contain larger instances than the sweep's, and across sizes the stakes do not track the start value (Section 4.4). The OP100 and OPLib values are also one-shot advantages against an independently trained uniform control, a different estimand (Section 4.5, Appendix G).

Table 3: Length-budget sweep on OP50 (three seeds each; benefit and oracle in percentage points of the per-budget reference; stakes as defined in Section 4.3 and hit@1 in Section 3.5). Every column measured on a model reports the standard deviation over the seeds; the stakes are model-free and measured once per budget; margins in the text use the unrounded stakes.

budget T	visit frac	$\lvert S\rvert$	stakes	realized benefit	oracle	capture κ	hit@1
1.5	0.24	2.05	0.261	17.38 ± 0.38	23.09 ± 0.11	0.753 ± 0.014	0.57 ± 0.04
2.0	0.34	2.10	0.242	14.24 ± 0.27	19.25 ± 0.01	0.740 ± 0.014	0.18 ± 0.07
3.0 (canonical)	0.55	2.21	0.169	7.78 ± 0.06	10.53 ± 0.02	0.738 ± 0.004	0.53 ± 0.06
4.0	0.73	2.39	0.111	5.34 ± 0.03	7.39 ± 0.04	0.722 ± 0.007	0.63 ± 0.04
4.5	0.81	2.48	0.086	4.52 ± 0.08	6.29 ± 0.02	0.718 ± 0.013	0.63 ± 0.06

$R^2 = 0.96$, within $[0.93, 0.97]$ under a bootstrap over seeds and stakes instances. Normalizing the benefit instead by the model's own one-shot prize from a uniformly random feasible start gives $R^2 = 0.94$. The length budget alone orders the benefit at least as well ($R^2 = 0.98$ on $1/T$ and 0.99 on $\log T$, Appendix G) and has the smaller held-out error. In a leave-one-budget-out test, each budget is predicted by a linear fit on the other four. The root-mean-square error is then 1.80 points for the stakes against 1.38 for $1/T$ and 0.85 for $\log T$, although on single budgets the stakes are sometimes closer (Table 4). The fit rests on five budgets, and in this test the two ends of the sweep are extrapolations.

Start-stakes and the oracle start value. The stakes and the oracle start value both measure the share of the reference that depends on the start, so they can be compared directly. In points of that reference the

Table 4: Leave-one-budget-out prediction error for the realized benefit (three-seed means), in percentage points of the per-budget reference, for the start-stakes measure and for two functions of the budget alone (error is realized minus predicted).

		start-stakes		error, budget only	
held-out budget	realized benefit	predicted	error	$1/T$	$\log T$
1.5 (extrapolation)	17.38	14.75	+2.64	−2.55	+0.21
2.0	14.24	15.12	−0.88	+1.67	+0.65
3.0 (canonical)	7.78	9.91	−2.13	−0.30	−1.37
4.0	5.34	5.32	+0.01	−0.24	−0.04
4.5 (extrapolation)	4.52	2.54	+1.98	−0.23	+1.13
root-mean-square error			1.80	1.38	0.85

stakes exceed the oracle start value by 2.3 to 6.3 at every OP50 length budget and agree with it on PCTSP50 (−0.16 points against a standard error of 0.20), where the light solver matches the reference. This ordering is empirical, and on OP50 the margin depends on the measurement. Taking the light solver's own best result on each instance, forced or unconstrained, as the denominator removes 0.8 to 2.7 points of the margin but leaves 1.2 to 4.1 (Appendix G). With sixteen restarts of one solver on both sides, forced and unconstrained alike, the stakes fall below the oracle start value at the canonical length budget (Appendix H), so their level depends on restarts as well as on solvers.

Problems and distributions. At the sweep's instance size the stakes, zero on TSP and well above zero on OP50, separate problems where the start matters from those where it does not, which a length budget cannot do. Their link to the benefit does not carry across problems, however (Figure 2, Appendix G). From the PCTSP50 stakes of 0.094 the OP50 line predicts 4.2 points against 7.2 from the same-checkpoint measurement on PCTSP50 (standard deviation 0.2 over three seeds), while capture transfers, at 0.75 (0.72 to 0.78 across seeds). The PCTSP50 stakes even lie below those of OP50 at T=4.0 while the PCTSP50 benefit is higher, so the stakes do not rank problems by benefit. OP100 and OPLib have stakes at or above the canonical benchmark's, 0.19 and 0.24 against 0.17, yet one-shot advantages far below the line (Table 5). These advantages are a different estimand, measured against the independently trained uniform control, on sets that contain larger instances than the sweep's, where σ is not comparable to the line (Section 4.4). Section 4.5 compares the best start with the head's pick on OP100 and OPLib instead.

Ordering length budgets, not individual instances. Both the stakes and the oracle start value fall as the budget loosens and measure how unequal the starts are, so their relation across budgets is expected. The line of realized benefit against stakes is that relation scaled by capture, its slope 0.76 of the slope of the oracle start value on stakes. Within a length budget even the size of an instance's near-optimal start set does not predict its realized benefit. The median rank correlations over three seeds lie between −0.08 and +0.02, and per-seed tests reach $p < 0.01$ only at T=4.5, in two of the three seeds and with negative correlations.

4.4 Training at larger scales

To test how far the capture measured at n=50 extends, we train SAS with the same OP recipe at n=100, 200 and 500, with three seeds each. At every scale the length budget is chosen so that good tours visit the same share of nodes, holding the near-optimal visit fraction near 0.55 with T=4.0, 5.5 and 9.0 respectively (Appendix H). At n=100 this is the standard OP100 budget, so these models solve the OP100 distribution of Section 4.5. Capture is computed as in Section 4.2, with the oracle over every feasible start.

Realized benefit across scales. The head beats a uniformly random feasible start on the same checkpoint at every trained scale, and its raw gain grows from +1.81 prize units (absolute collected prize, not gap points) at n=100 to +3.22 at n=500 (Appendix H). Normalizing by the uniform-start prize, the mean one-shot prize

Table 5: The one-shot advantage and the head's hit rate decay together under distribution shift; entries are mean ± std. Advantage = paired one-shot gap difference, uniform control minus SAS, over every seed the two configurations share (k in the table). hit@1 as in Section 3.5, at three seeds throughout. 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	k	one-shot advantage (points)	head hit@1	chance
OP50 (in-distribution)	5	$+7.66 \pm 0.11$	$52.8\% \pm 5.6\%$	4.4%
OP100 (size + budget)	5	$+3.72 \pm 1.15$	$47.2\% \pm 9.4\%$	2.5%
OPLib (compound)	5	-0.89 ± 3.42	$11.8\% \pm 4.3\%$	2.0%

from such a start, removes the growth of absolute prize with n. Over the four sizes the normalized benefit roughly halves for every threefold increase in n, from 0.088 at n=50 to 0.020 at n=500.

Capture across scales. Capture stays near the n=50 range at n=100 (0.712) and falls below it at n=200 and 500 (0.574 and 0.505), with the nine cells of scale and seed spanning 0.358 to 0.745. The narrow range across length budgets (Section 4.2) thus does not extend across scale, although most of the decline in normalized benefit comes from the fall in the oracle start value relative to the uniform-start prize. From n=50 to n=500 the normalized benefit falls 4.5-fold, a 3.0-fold fall in that start value times a 1.5-fold fall in capture, so the start value carries about three quarters of the decline on a log scale. Section 4.5 computes capture within a fifty-draw pool, fifty distinct random feasible starts decoded by the checkpoint, with the best of them as the oracle and the head's highest-ranked one as its pick. Such a pool covers all feasible starts at n=50 but a tenth of them at n=500, where it misses the best start more often and overstates capture by 0.08.

Start-stakes across scales. As defined, the start-stakes do not track the start value across scale, rising from 0.17 at n=50 to 0.22 at n=500 while the start value relative to the uniform-start prize falls threefold. The rise comes from the two solvers in the measure, since over that range the light solver's unconstrained eight-restart result falls from 0.967 to 0.835 of the reference. In a single-solver form, the same local search at the same number of restarts computes both the forced and the unconstrained value. With the light solver at sixteen restarts on both sides, the stakes on the same instances fall from 0.090 at n=50 to 0.034 at n=500, as the start value does (Appendix H). The stakes as defined are therefore usable at one instance size only. The single-solver form, which does track the start value, rests on four sizes.

4.5 Distribution shift

The advantage of SAS decays with distribution shift, and under compound shift SAS is no better overall than the uniform control (Table 5). We evaluate the OP50-trained models zero-shot at two levels of shift. OP100 doubles the nodes of the same generator, with its canonical length budget of T=4. The OPLib benchmark of 210 Euclidean instances shifts four axes at once. It swaps uniform squares for real city geometries and the 50 training nodes for 50 to 5,933. Only one of its three prize schemes resembles the training prizes, and its length budgets range from 2.0 to 22.1 in the unit-square coordinates the models see, against 3 in training. Gaps on OPLib are relative to the best-known EA4OP solutions shipped with the benchmark.

Moderate shift (OP100). On OP100 the one-shot advantage over the uniform control halves, and the advantage narrows further as the decoding budget grows. At one rollout it is +3.7 points and positive in all five seeds (Tables 5 and 10). The paired difference is significant in every seed at one and ten rollouts. From twenty-five rollouts one of the five seeds reverses, while the seed-paired interval still excludes zero at twenty-five and fifty rollouts and covers it from N=n=100 upward (Appendix F). The head still hits the near-optimal start sets far above chance, but SAS's one-shot gap loses the cross-seed stability it has in distribution (Figure 1). Its cross-seed standard deviation is 0.18 points on OP50 (Table 1) and 2.4 points on OP100 (Table 10).

Compound shift (OPLib). On OPLib the whole model family is weak, and learned start selection does not change the overall gap significantly. At fifty rollouts POMO, the uniform control and SAS reach gaps of 54.6%, 53.1% and 55.1% over all instances (Table 11), with every two-standard-error interval of the overall seed-paired differences covering zero (Appendix F). Instance size degrades all three configurations alike, and start selection changes only the first step of each tour. At fifty rollouts POMO's own gap grows from 21% at $n \leq 100$ to 80% beyond 1,000 nodes, and the other two configurations stay within a few points of that gap at every size bracket (Table 12).

Capture by prize scheme on OPLib. On the 102 OPLib instances with $n \leq 300$, capture against a fifty-draw pool is indistinguishable from zero on the constant and pseudo-random prize schemes, the two farthest from training (-0.059 ± 0.082 and $+0.033 \pm 0.055$, two standard errors over three seeds), against $+0.340 \pm 0.072$ on the distance-based scheme used in training. Against the independently trained uniform control, SAS is nominally worse on OPLib and significantly worse on the 70 distance-based instances of all sizes, where the control leads by 2.8 to 3.2 points on average at ten to fifty rollouts and by a positive margin in every seed. Capture compares the head's pick with a random one on the same checkpoint, whereas the comparison with the control involves two independently trained configurations, and the control has the smaller mean gap in every row of Tables 11 and 12 (Appendix F).

The head concentrates on other nodes. On OPLib the head still concentrates its mass as sharply as in distribution, with top-1 probability 0.98 to 0.99. On the instances with $n \leq 300$ the near-optimal start sets remain sparse at 2.3 nodes per instance, but on most instances the mass lands outside them. Both the prize scheme and the other shifts lower the hit rate. On distance-based prizes the hit rate falls from 53% in distribution to 31% under the shifts in geometry, size and length budget, and the two shifted prize schemes lower it to 4% and 0%. A controlled instance set that varies only the prize scheme shows that the scheme alone changes how much the choice of start matters (Appendix G).

Decomposing the decline. On both shifted sets value remains in the choice of start while capture falls. On the same checkpoints, decoding from every start beats the head's one-shot choice by 7.1 points on OP100, and decoding from fifty starts, the evaluation's decoding budget, beats it by 14.2 points on the 102 OPLib instances with $n \leq 300$. Against a fifty-draw pool, capture falls from about three quarters in distribution to about half of that for the n=50 model on n=100 instances (the paragraph on zero-shot transfer below).

Classical construction on OPLib. On the 102 OPLib instances with $n \leq 300$, classical construction beats every neural configuration in quality, the reverse of OP50 (Section 3.4), and one construction costs less than one neural decode. At equal numbers of constructions and rollouts, the construction stage of the local-search solver, prize-per-detour greedy insertion without local search, reaches a 27% gap with one construction against 48% one-shot for the neural configurations. Its best of sixteen, at 17%, beats even fifty rollouts at 32% to 34%. On one CPU core of the same machine, one construction costs a mean of 10.2 ms per instance and the SAS one-shot decode 37.7 ms (Table 6). One construction is cheaper in every size bracket, though the ratio narrows toward the largest instances in this subset, while best of sixteen costs more than one decode. At the cost of one decode, construction is therefore the stronger choice here, and the neural decode's lead over construction was measured on OP50 only.

Zero-shot transfer and training on the target distribution. Training on the target distribution roughly doubles capture, and the start value available to a model depends on the model as well as on the instances. Checkpoints trained at n=50 and at n=100 decode one fixed set of 1,000 instances at n=100 from the same fifty-draw pool. A model trained at n=100 captures 0.729 of the start value in the pool, against 0.369 for the n=50 model applied zero-shot, a paired difference of $+0.360 \pm 0.012$ (two standard errors over three seeds). The start value in the pool is 2.46 prize units for the trained model against 3.43 for the zero-shot model, whose uniform-start prize is lower.

4.6 Summary of the limits

Three cases limit the benefit of SAS. When the decoding budget covers every start, the advantage vanishes (Section 3.2). When the OP length budget loosens, the start value shrinks, but across the sweep the realized benefit stays more than an order of magnitude above the differences SAS makes on must-visit problems (Sections 4.1 and 4.2). Under distribution shift, value remains in the choice of start while capture falls, halving on OP100 and not differing significantly from zero on two of the three OPLib prize schemes for instances with up to 300 nodes (Section 4.5). Larger instances combine the second and third cases, since both the start value and capture fall from n=50 to n=500, the start value accounting for about three quarters of the decline on a log scale. At one instance size the model-free stakes and the model's capture together show which of the last two cases applies, whereas across sizes the stakes do not track the start value (Section 4.4).

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. On the TSP the forced starts are justified by cyclic symmetry, and later work developed the same idea further; Sym-NCO (Kim et al., 2022) regularizes with solution symmetries. This line of work treats the first step only as a source of diversity. On selective routing, however, the forced first step takes the subset decision at that step away from the learned policy.

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 references, 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.[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. None of these works identifies a mechanism behind the selective-routing gap.

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 first steps chosen by the policy then work better than forced ones. SAS differs in three ways. Its diversity comes from the start set itself (one policy, no latent vectors), its loss is POMO's original full-N shared-baseline objective with one deleted term restored and no best-of-K selection, and its first step remains a designated multistart step, chosen by a trained head whose log-probability enters the gradient. The two approaches are complementary in principle, since a start head could be combined with a latent-conditioned decoder. Closest to SAS, 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; Table 13 compares the two designs. Since its code is not released, we reimplemented it and ran it head-to-head on OP50 and PCTSP50 (Appendix I): an independent design repairs the same bias, reaches lower mean gaps at every decoding budget, and places its selections in the same sparse near-optimal start sets as ours. Its selector is about 1,700 times larger, and our reimplementation trains it with a separate optimizer and 100-step gradient accumulation (Appendix I). 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).[3] These are complementary consumers of the same budget.

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

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

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.

6 Conclusion

The multistart recipe behind POMO-style constructive neural solvers, as implemented in RL4CO, hardcodes the first step as a symmetry prior that never enters learning. On selective routing this removes a real decision. A single linear head of 128 parameters returns the decision to the policy and cuts the one-shot gap on OP50 from 11.9% to 3.8% over five seeds, with performance at the full decoding budget unchanged, and the same head fitted afterwards to a frozen POMO checkpoint decodes within seed noise of joint training. The mechanism carries over to PCTSP50, although, as on OP50, the one-shot gap remains above that of AM. On OP50 a hand-crafted start rule recovers most of the gain, and the learned head improves on the rule at each measured decoding budget below full coverage. POMO+, with a much larger start agent, reaches lower gaps on both problems.

How much the first step is worth depends on the problem, the length budget and the distribution. Across five OP50 length budgets a trained head captures about three quarters of the available start value. It captures about as much when the same recipe is trained at n=100 and less at n=200 and 500. Under distribution shift the start value persists, while capture halves on OP100 and does not differ significantly from zero on two of OPLib's three prize schemes for instances with up to 300 nodes, against +0.34 on the third. The start-stakes measure, computed before any training, shows whether the start matters on a problem and orders length budgets by the benefit of SAS within one problem and instance size, although the length budget alone orders that benefit at least as well. Its level depends on the solvers used to compute it, and as defined the measure does not track the start value across instance sizes. Making capture robust to distribution shift, through test-time input normalization, training-time distribution augmentation, or head features invariant to prize and geometry rescaling, is the natural next step.

Broader Impact Statement

This work analyzes and repairs a first-step bias of the POMO multistart recipe and characterizes when start selection 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 start-stakes measurement may help practitioners avoid spending compute on start selection where an inexpensive check shows that the start matters little.

Code Availability

Code and per-run outputs are available at
`https://github.com/Yanko96/Returning-the-First-Step-to-the-Policy`.

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)*, pp. 5278–5289, 2025. doi: 10.1145/3711896.3737433.

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.

Laurent Perron and Vincent Furnon. OR-Tools. `https://developers.google.com/optimization/`, 2023.

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.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3–4):229–256, 1992.

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*, volume 35, pp. 12042–12049, 2021. doi: 10.1609/aaai.v35i13.17430.

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):1–24, 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) with test seed 1234 and pinned by checksum (`md5 47a828139a9fcf1e642a060f40837d9e`).

Gap definition. Gap $= (P^\star - P_{\text{model}})/P^\star$, where P is the mean collected prize over the test set and $P^\star = 16.17$ (Compass). For PCTSP, a minimization problem, gap $= (C_{\text{model}} - C^\star)/C^\star$ with $C^\star = 4.48$ (OR-Tools). This is the aggregate-mean convention used by the benchmark literature on this set.

Decoding-budget accounting. Each decoding scheme is charged its rollout count per instance. Greedy decoding, from a free first step or from a configuration's start choice, costs 1. Multistart with N starts costs N, whether the starts come from the head, the start rule (its N highest-scoring nodes) or the enumeration. Sampling with N samples costs N; Table 9 uses 1,280 library samples. Beam search of width N keeps the best final beam and costs N. Instance augmentation costs the number of augmentations (1,280), and multistart combined with 16 symmetric augmentations costs $16N$.

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 multistart evaluation utilities call the policy without passing on their environment, so the policy silently instantiates a default one sized for 20 nodes and takes start indices modulo 20. On 50-node instances multistart with 25 and with 50 starts then evaluate the same 20 starts, which makes the results of the two schemes bit-identical. Evaluating the unmodified baseline through that path would understate its multistart performance and favor SAS.

Start selection of the uniform control. At every decoding budget the uniform configuration draws N distinct feasible starts uniformly at random. Concretely, the head's logits are replaced by uniform random scores over the feasible mask before the top-k, so the selection is an unordered random subset and no tie-break needs to be specified. At $N < n$ it shares no start set with the enumerated configuration, and at $N = n$ all three configurations decode the same n starts.

The sixteen augmentations. The augmented scheme uses the library's `symmetric` augmentation with 16 copies, which is not a fixed enumeration of the dihedral group. The first copy is the instance itself; each of the remaining fifteen draws an angle $\phi \sim \mathcal{U}[0, 4\pi)$, rotates the coordinates about $(0.5, 0.5)$ by ϕ, and additionally swaps the x and y axes when $\phi > 2\pi$, so about half the copies are reflected as well as rotated. The transforms are therefore random draws, which the evaluation does not seed.

Other evaluations reached by the same path. The condition that triggers the 20-start truncation is a policy call without an environment argument, as inside the library's multistart evaluation utilities, on instances larger than 20 nodes, at a start count above 20. Any result produced that way reports the same 20 starts for every larger decoding budget, so its multistart rows are understated and stop improving above 20 starts. The reported weakness of POMO-style models on selective routing is the starting point of this paper, and our own retrained baseline reaches substantially stronger multistart performance than the corresponding published row. Whether a given published number is affected depends on how it was evaluated, which the condition above lets anyone check against their own configuration.

Published reference values. 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 decoding runs once per decoder, five in all, so its cheapest decode costs five rollouts.

Wall-clock costs. Table 6 reports two kinds of measured cost. *Batched throughput* (GPU rows, single RTX 4090, full 10,000-instance set at the fixed evaluation batch sizes) answers how much a large offline evaluation costs; it amortizes fixed per-forward overheads. *Single-instance latency* answers what one decision costs in

a planning loop. It is measured at batch size one on one core of one machine (an Intel Xeon Gold 6240R), for the neural decode and for classical construction alike, on the first 100 OP50 test instances (the first 32 for the iterated local search rows) and on the 102 OPLib instances with $n \leq 300$ of Section 4.5. Entries are means over instances, with input preparation outside the timer, and for best of sixteen only its sixteen noised constructions are timed. On the OPLib subset one construction stays cheaper than the neural decode in every size bracket, although the ratio narrows from about seven at $n \leq 100$ (3.7 against 27.0 ms) to about two at $201 \leq n \leq 300$ (24.2 against 52.7 ms), while best of sixteen costs more than the neural decode in every bracket. Compass is cited as published. Kool et al. (2019a) report five minutes for the same 10,000 instances, solved 32 at a time on a 32-thread CPU system (two Xeon E5-2630), which is about 30 ms per instance amortized over the 32 threads and about one second per instance on one thread; comparisons with it are accurate to an order of magnitude. Batched on a GPU, the SAS one-shot decode costs 0.053 ms per instance, about 570 times less than that amortized figure. The start head itself adds no measurable inference cost (the POMO and SAS throughput rows agree within 0.002 ms).

Table 6: Measured decoding cost per instance. Batched throughput on OP50 on one RTX 4090. Single-instance latency at batch size one on one core of an Intel Xeon Gold 6240R, mean over the first 100 OP50 test instances (iterated local search rows over the first 32) and over the 102 OPLib instances with $n \leq 300$. ILS = the iterated local search solver of Appendix E; constructor = its construction stage only.

decode	rollouts	OP50	OPLib, $n \leq 300$
batched throughput (GPU, 12 to 2,000 instances per batch)			
POMO, plain greedy	1	0.051 ms	—
SAS one-shot	1	0.053 ms	—
SAS multistart 50	50	0.41 ms	—
SAS multistart × aug	800	4.2 ms	—
single-instance latency (batch 1, one CPU core)			
SAS one-shot	1	22.6 ms	37.7 ms
constructor, best of 1	—	1.8 ms	10.2 ms
constructor, best of 16	—	31.0 ms	183 ms
ILS, 1 restart	—	0.32 s	—
ILS, 16 restarts	—	5.2 s	—
Compass (published, one thread, other hardware)	—	∼1 s	—

Determinism. Evaluation uses fixed per-scheme batch shapes, which matters because batch-shape dependent kernel numerics can shift greedy decoding results; greedy decoding without random starts is deterministic given the checkpoint, and among the stochastic schemes of Table 1 only the sampling rows fix and record their seeds.

Statistics of the decoding-budget curve. At one rollout the median paired improvement of SAS over the uniform control is +6.4 to +6.7 points per seed (interquartile range roughly 1.9 to 12.5), with about 85% of instances improved against 14% worsened; the paired Wilcoxon signed-rank test gives $p < 10^{-6}$ for every seed at every decoding budget up to twenty-five rollouts. Per-seed medians are +1.24 to +1.30 at ten rollouts and +0.25 to +0.31 at twenty-five. The seed-paired mean advantage over the uniform control is 1.565 ± 0.026 points at ten rollouts, and over plain POMO it is 1.591 ± 0.036 at ten and 0.583 ± 0.039 at twenty-five (two standard errors over the shared seeds). Figure 3 plots the full curve.

Precision from pairing. Pairing supplies most of the statistical precision on this benchmark. Across decoding budgets the configurations' gaps correlate across seeds at $r = 0.95$ to 1.00 (SAS and the uniform control at one rollout, all three from ten rollouts), so seed quality is shared across configurations and the paired difference isolates the start mechanism. At one rollout the cross-seed standard deviations of SAS and the uniform control are 0.18 and 0.27 points, while their paired difference has a standard deviation of 0.11, a third of what independent configurations would give.

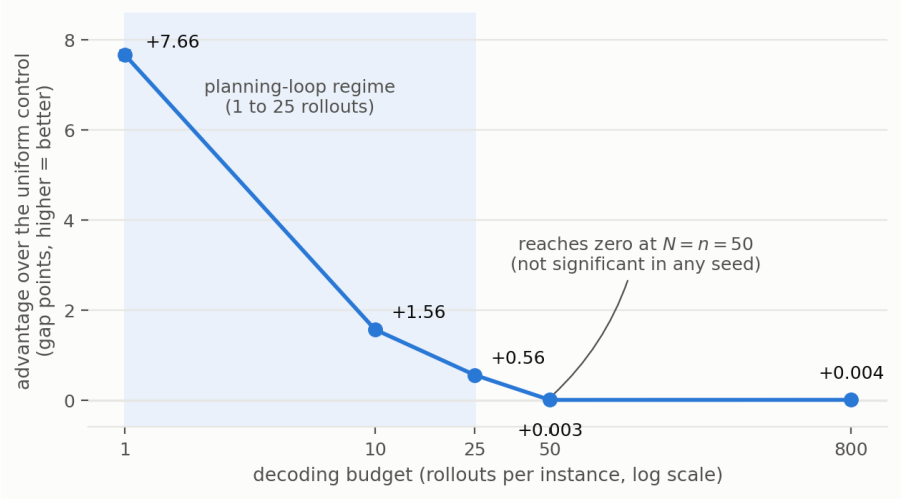


Figure 3: Advantage of SAS over the uniform control (gap percentage points, mean ± std over the five seed pairs the two configurations share) versus decoding budget. The advantage decays from $+7.662 \pm 0.106$ at one rollout to zero at $N{=}n{=}50$ (paired Wilcoxon not significant in any seed), and its seed-paired interval covers zero through the full 800-rollout scheme. Error bars are smaller than the markers beyond $N{=}1$.

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. 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 mixed-precision forward, because the Gumbel sampler's noise clamp underflows to zero in 16-bit precision and the logarithm of the clamped uniform draw then produces infinities.

The exhaustive-start regime of the reported runs. Every start is feasible under the length budget once $T \geq 2\sqrt{2} \approx 2.83$, the longest depot round trip on the unit square, which covers $T = 3$, 4 and 4.5. Training uses $N{=}n$, so Gumbel-top-k selects all n starts in every batch and the start set is exhaustive. In this regime the realized returns do not depend on the head, and Eq. (4) reduces to an advantage-weighted likelihood of every start under π_{head}, in which each of the n starts pushes its own log-probability up or down by its advantage in every batch. This exhaustive-start update 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 3.5).

Training below $2\sqrt{2}$. Two budgets of the sweep in Section 4.2, $T{=}1.5$ and $T{=}2.0$, lie below $2\sqrt{2}$, and only SAS is trained there. On the first 1,000 canonical test instances a mean of 39.4 of the 50 nodes are feasible starts at $T{=}1.5$, and fewer than 50 are feasible on 90.7% of instances; at $T{=}2.0$ the figures are 48.6 and 45.3%, while at $T = 3.0$, 4.0 and 4.5 every start is feasible on every instance. When an instance has $m < N$ feasible starts, Gumbel-top-k returns each of them once, and the remaining $N - m$ rollouts take starts drawn with replacement from the head's softmax over the feasible nodes, at the temperature of the schedule in Section 2.3. Later steps are sampled during training, so rollouts that share a start are sampled independently after the first step. All N rollouts, repeated starts included, enter the shared baseline b, which remains their mean, and each contributes $-(R_s - b) \log \pi_{\text{head}}(s_s)$ to Eq. (4) with weight $1/N$, where $\log \pi_{\text{head}}$ is the log-softmax over the feasible nodes. A start that occurs r times among the N rollouts therefore contributes r terms, and the loss is not renormalized by m or by the number of distinct starts. On the test instances above, $N - m$ averages 10.6 at $T{=}1.5$ and 1.4 at $T{=}2.0$. At these two budgets the head's update is thus the exhaustive form over the feasible starts plus $N - m$ samples from the head, which add weight to

the starts the head already prefers. This padding replaces the library's guard of Section 2.2, which would redraw every start of the batch uniformly.

Head-local credit. In a seed-42 run, backpropagating the credit term through the encoder embeddings degraded the whole policy, which reached a gap of 3.21% at the full decoding budget, where the released-recipe POMO run evaluated the same way reaches 0.009%. The start-credit signal is orders of magnitude sparser than the per-step objective, which is consistent with its being a poor training signal for the encoder. With the stop-gradient of Section 2.3, training curves are indistinguishable from plain POMO.

Fitting the head to a frozen checkpoint. Only the head weights train, with the credit term of Eq. (4) over all $N{=}n$ starts at temperature 1 from the first step (Adam, learning rate 10^{-4}, weight decay 10^{-6}, batch 64, ten epochs of 2,500 steps). Validation instances are disjoint from the test set, and the head after the last epoch is kept.

C PCTSP penalty sensitivity, protocol and results

Data. 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).

Penalty sensitivity. How many nodes good PCTSP solutions leave unvisited rests on one published design choice, the penalty factor of 3 (Kool et al., 2019a). Over the factor grid $[0.5, 8]$ we measure two properties, the design intent of visiting about half the nodes and the source paper's statement that the prize constraint is "sometimes binding and sometimes not"; two extreme penalty factors outside the grid serve as checks of the solver. Both hold at the standard setting with margin, with visit rate 0.54, quota binding on 33% of instances, and visit-all saturation only beyond the grid.

Results under the OP protocol. Table 7 gives all schemes. The near-optimal start sets are extracted with the same solver as the sensitivity study above, under the conventions of Appendix E with the near-optimal rule in minimization form (objective within 0.5% of the best restart).

Table 7: PCTSP50, all schemes (gap % to OR-Tools cost 4.48; bold as in Table 1). Rows down to plain greedy are mean ± std over five seeds per configuration. In the sampling and augmentation rows below, the uniform and SAS entries are means over three seeds and the enumerated entry is a single run, so those two rows are not paired across configurations.

decoding scheme (rollouts)	enumerated (POMO)	uniform	SAS (ours)
one-shot via start distribution (1)	—	11.849 ± 0.057	$\mathbf{4.732 \pm 0.148}$
multistart 10	4.201 ± 0.032	4.203 ± 0.023	$\mathbf{2.377 \pm 0.056}$
multistart 25	2.856 ± 0.037	2.846 ± 0.022	$\mathbf{2.182 \pm 0.027}$
multistart 50 ($= n$)	2.151 ± 0.030	2.160 ± 0.015	2.159 ± 0.024
multistart 50 × aug 16 (800)	1.023 ± 0.016	1.024 ± 0.006	1.021 ± 0.020
one-shot, plain greedy (1)	9.745 ± 0.625	10.660 ± 0.856	10.151 ± 1.105
sampling 1,280	0.823	0.817 ± 0.019	0.823 ± 0.020
aug 1,280	2.507	3.227 ± 0.527	2.858 ± 0.051

D Setup and results of the must-visit control

SAS was applied unchanged to TSP50 and CVRP50, with the official POMO recipe per problem and the head configuration of Section 2.3 (on TSP, where there is no depot, node 0 is an ordinary city and is not masked from the start distribution). Uniformity is measured on 1,000 freshly sampled instances (head softmax over feasible first actions) as mean entropy against $\ln n$, the Kullback–Leibler (KL) divergence 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.

Table 8: 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 in points against one plain-POMO run per problem, the released recipe; each problem contributes two SAS runs against that single comparator, so the parity columns have no dispersion. Column heads: one-shot = one rollout from the head's argmax start, decoded greedily; ms10/ms50 = multistart with 10/50 rollouts; msaug = multistart with 16 augmentations.

problem	run	$H/\ln n$	KL	top-1	one-shot	ms10	ms50	msaug
TSP50	1	0.017	3.85	0.972	−0.195	+0.083	+0.006	+0.001
TSP50	2	0.015	3.85	0.976	−0.178	+0.012	−0.015	0.000
CVRP50	1	0.004	3.90	0.994	+0.263	−0.059	+0.006	−0.004
CVRP50	2	0.005	3.89	0.991	+0.314	−0.024	+0.022	+0.001

The one-shot differences 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. The persistence of concentration under exact symmetry is itself consistent with the exhaustive credit signal of Section 2.3, since nothing in the objective pushes the head toward uniform, and accumulated advantage noise is enough for softmax dynamics to settle on an arbitrary winner.

E Near-optimal start-set extraction

Every hit-rate number in the paper is measured against near-optimal start sets extracted by an independent local-search solver. Construction is greedy insertion under the score $\rho_i^{\gamma_p}/\text{detour}_i^{\gamma_d}$, run once without noise and followed by descent 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 from a sampled criterion shape with noise 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. Figure 4 shows the head's start distribution against these sets on the first three test instances.

F Additional results

Table 9 extends Table 1 with the sampling and augmentation-only schemes. Tables 10, 11 and 12 give the complete zero-shot numbers behind Section 4.5. Per-seed rows and raw per-instance outputs for every cell are included in the supplementary material.

Table 9: Decoding schemes on OP50 not shown in Table 1 (gap % to Compass 16.17, 10,000 instances). The uniform and SAS entries are means over three seeds and the POMO entry is a single run, so the rows are not paired across configurations.

decoding scheme (rollouts)	POMO	uniform	SAS (ours)
sampling 1,280	0.149	0.145 ± 0.012	0.134 ± 0.014
aug 1,280	3.806	4.840 ± 1.081	4.752 ± 1.048

Sampling at one rollout in Table 1 injects one uniformly random feasible start and samples the remaining steps. The library's sampling evaluator cannot be used there, because when asked for one sample it turns multistart off, finds no sample count, and falls back to n rollouts whose first step is drawn from the untrained step-zero distribution.

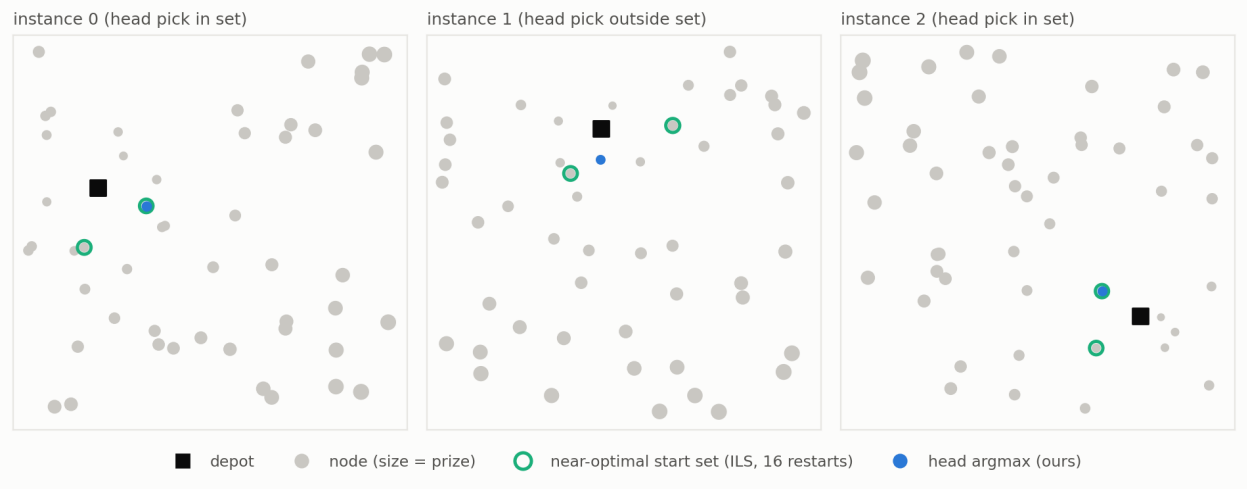


Figure 4: Head start distribution versus the near-optimal start set (first three test instances, one trained model). Rings mark the near-optimal start set extracted from ILS solutions; the filled marker is the head's argmax.

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 on this set, from the same source table as the OP50 reference. Because $n \neq 50$, both multistart at 50 starts and at $N{=}n{=}100$ are reported. Seed-paired, SAS leads the uniform control by $+0.64 \pm 0.50$ points at twenty-five rollouts and $+0.46 \pm 0.41$ at fifty (two standard errors), intervals that exclude zero although one of the five seeds reverses at both. From $N{=}n{=}100$ upward the interval covers zero.

Table 10: OP100 zero-shot, all schemes (gap % to Compass 33.19; mean ± std over five seeds per configuration). Bold as in Table 1. SAS differs significantly from both controls in every seed at one and ten rollouts only; from twenty-five upward the per-seed differences change sign against both.

decoding scheme (rollouts)	enumerated (POMO)	uniform	SAS
one-shot via start distribution (1)	—	17.765 ± 1.368	$\mathbf{14.046 \pm 2.379}$
multistart 10	9.194 ± 0.631	9.525 ± 0.521	$\mathbf{8.491 \pm 0.819}$
multistart 25	7.996 ± 0.594	8.229 ± 0.362	7.589 ± 0.706
multistart 50	7.387 ± 0.579	7.594 ± 0.312	7.136 ± 0.592
multistart 100 ($= n$)	6.967 ± 0.557	7.148 ± 0.271	6.949 ± 0.525
multistart 100 × aug 16 (1,600)	4.176 ± 0.248	4.301 ± 0.210	4.186 ± 0.184
one-shot, plain greedy (1)	18.048 ± 2.064	19.673 ± 1.861	19.404 ± 0.903

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); every decoded tour is then mapped back to the original coordinates and re-validated under the TSPLIB integer metric, and rollouts exceeding the integer length budget are discarded. An instance whose every rollout is discarded scores zero collected prize. Rollout discard rates are close across configurations (one-shot: $10.6 \pm 2.2\%$ for POMO, $9.5 \pm 2.5\%$ for uniform, $10.4 \pm 1.9\%$ for SAS; at fifty rollouts: $10.7 \pm 1.4\%$, $9.6 \pm 1.6\%$, $10.7 \pm 0.9\%$), so the spread of Table 11 is not an artifact of differential validity. Collectable score excludes the depot. Gaps are relative to the per-instance EA4OP solution values included in the OPLib repository (Kobeaga et al., 2018), heuristic best-known values.

The library's solution-length assertion uses a 10^{-5} float tolerance that these fractional length budgets can exceed by rounding, so evaluation relaxes it to 10^{-4}. Of roughly 271,000 rollouts, 1,440 fell in the relaxed band and the integer re-validation discarded 714 of them. The instance length budgets fix the budget-to-tour-length ratio at $\alpha = 0.5$, the same regime as the synthetic training length budgets.

Axis separation on OPLib. On the prize scheme closest to training (distance-based, gen3) the head's hit rate is 31%, so the remaining shifts in geometry, size and budget lower it from 53% by about two fifths; on the two shifted prize schemes it is 4% and 0%. Cross-seed agreement on the heads' argmax start (seed 42 against seeds 43 and 44) falls from 72% and 79% of the first 200 test instances in distribution to 18% and 30% of the 210 OPLib instances (33% and 39% of the 102 with $n \leq 300$). The near-optimal start sets on OPLib remain as sparse as on synthetic data (2.3 nodes per instance), and on 80% to 89% of instances the head places over 0.9 probability on a start outside the set. On the $n \leq 300$ subset the fifty-draw pool covers between 17% and 100% of the nodes.

Seed-paired differences on OPLib. At fifty rollouts the enumerated configuration alone ranges from 49.1% to 63.3% across its five runs. SAS minus the uniform control is +1.97 points against two standard errors of 3.74, and SAS minus POMO is −0.81 against 1.81 over the four seeds those two configurations share; the one-shot pair is +0.89 against 3.06 and −0.77 against 1.63 (Table 5 reports the same one-shot difference with the opposite sign, as uniform minus SAS). No interval excludes zero, and the two differences have opposite signs. In every row of Tables 11 and 12 the uniform control has a smaller mean gap than SAS, and in all but one SAS has a larger mean gap than the enumerated baseline, while against the enumerated baseline the paired difference favors SAS at all four decoding budgets. Those two readings rest on different seed sets and disagree. The five-run means of Table 11 put the enumerated configuration ahead at all four decoding budgets, and the pairing drops its released-recipe run, which has no SAS counterpart. That run is the second best of that configuration's five here, so dropping it raises the enumerated mean by 1.35 points at fifty rollouts, and the −0.81 favoring SAS is in part an artifact of which seeds can be paired. Neither reading is significant. By prize scheme and instance size, the only difference whose interval excludes zero at more than one decoding budget is SAS minus the uniform control on the distance-based prizes at ten to fifty rollouts (+3.2, +3.1 and +2.8 points, a higher gap for SAS, with two standard errors of 1.7 to 1.8, positive in every seed). Three isolated cells at one or ten rollouts also exclude zero, with mixed signs, and at fifty rollouts every other scheme and size covers zero for all three pairs of configurations.

Table 11: OPLib zero-shot, mean gap % to the EA4OP reference over 210 instances (mean ± std over five seeds per configuration). One-shot decodes greedily from the configuration's start choice, and for POMO from a free first step. The cross-seed spread exceeds every between-configuration difference at every decoding budget.

scheme	enumerated (POMO)	uniform	SAS
one-shot (1)	65.32 ± 4.88	64.93 ± 2.50	65.82 ± 3.18
multistart 10	57.25 ± 5.70	55.79 ± 2.79	58.20 ± 3.74
multistart 25	55.57 ± 5.92	54.03 ± 2.97	56.23 ± 4.07
multistart 50	54.56 ± 5.90	53.09 ± 3.00	55.06 ± 3.97

G Protocol of the length-budget sweep

Length budgets and training. The four non-canonical length budgets, $\{1.5, 2.0, 4.0, 4.5\}$, are selected around the canonical T=3 by the near-optimal visit fraction alone, measured as a function of T on 64 instances over the grid $T \in [1.5, 6.0]$, to span visit fractions 24% to 81%. The canonical length budget reuses the main-paper checkpoints.

Outcome conventions. All outcome quantities are measured on the first 1,000 canonical test instances under each swept length budget. One-shot decodes: learned start (the head's argmax start), uniform start (the same checkpoint decoding from a uniformly random feasible start; mean over eight fixed draws), and

Table 12: OPLib multistart-50 mean gap % by prize generation and by instance size (mean ± std over five seeds per configuration). gen3 prizes are distance-based like the training distribution; gen1 are constant, gen2 pseudo-random.

bucket	instances	enumerated (POMO)	uniform	SAS
gen1	70	62.51 ± 7.17	61.33 ± 5.05	63.01 ± 5.27
gen2	70	58.82 ± 7.74	58.10 ± 3.83	59.59 ± 8.34
gen3	70	42.36 ± 4.62	39.84 ± 3.04	42.59 ± 3.21
$n \leq 100$	39	21.48 ± 5.48	21.19 ± 3.88	22.79 ± 4.86
$101 \leq n \leq 300$	63	41.19 ± 7.23	39.42 ± 3.21	41.20 ± 3.77
$301 \leq n \leq 1000$	36	62.05 ± 6.83	60.81 ± 5.35	63.86 ± 3.88
$n > 1000$	72	80.44 ± 4.79	78.47 ± 3.69	80.28 ± 4.88

oracle (best one-shot over every budget-feasible start). The per-budget reference is the mean best prize of the Appendix E solver (16 restarts) on the same instances; realized benefit and oracle start 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 length budget the two conventions give 7.78 ± 0.06 points over the sweep's three seeds and 7.66 ± 0.11 over the table's five, distinct estimands whose closeness is consistent with the near-equivalence of the two backbones at the full decoding budget (Table 1).

The start-stakes measure. The measure is $\sigma = 1 - \overline{P}_{\text{forced}}/P_{\text{ref}}$. The two quantities come from different solvers:

- $\overline{P}_{\text{forced}}$: eight start nodes drawn uniformly without replacement from the feasible set $\{s : 2\,\text{dist}(0, s) \leq T\}$; for each, a lightweight iterated local search (greedy insertion, 2-opt, drop and reinsertion rounds, with the start node pinned through every move) run with **two restarts**; the mean of the eight best prizes.
- P_{ref}: the unconstrained best from the stronger solver of Appendix E at **sixteen restarts**.

A budget's stakes are the mean over 200 instances. The light solver is also run unconstrained, at eight restarts, but that estimate does not enter σ and is reported only as a drift diagnostic below.

Because the numerator and the denominator use different solvers, no quality bias cancels inside the ratio. The light solver's unconstrained estimate reaches 0.955 to 0.990 of the reference across the five budgets, so part of every reading is the light solver's own shortfall. Recomputing the stakes with the light solver on both sides, its best result on each instance (the eight-restart unconstrained run or the best forced run) as the denominator, lowers them by 1.9, 2.7, 2.2, 1.4 and 0.8 points at $T = 1.5$ to 4.5, to 24.2, 21.5, 14.7, 9.6 and 7.8, which still lie above the oracle start value at every length budget, by 1.2, 2.2, 4.1, 2.2 and 1.5 points; this variant fits the realized benefit with $R^2 = 0.98$. On PCTSP50 the light solver's unconstrained run matches the reference (1.000), so the two definitions agree there.

As defined, the stakes exceed the oracle start value of Table 3 by 3.0, 5.0, 6.3, 3.7 and 2.3 points at $T = 1.5$ to 4.5. With every mean prize written as a share p of the reference, the margin is, up to the instance-set difference noted below, $\sigma - (p_{\text{oracle}} - p_{\text{unif}}) = (1 - p_{\text{oracle}}) + (p_{\text{unif}} - \bar{p}_{\text{forced}})$, where p_{oracle} and p_{unif} come from the model decoding from its best feasible start and from a uniformly random one, and $\bar{p}_{\text{forced}}$ from the light solver forced to a random start. The first term, the reference solver's lead over the model's best start, is 0.9 to 1.3 points. The second, by which the network from a random start exceeds the local search from a forced one, is 1.5 to 6.0 points. The stakes are a per-instance average over 200 of the 1,000 test instances, so the two terms match the margins to within 0.8 points.

Uncertainty. The five-point ordinary least-squares (OLS) fit 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 $[0.932, 0.972]$. In sample, the leave-one-budget-out refits span $R^2 \in [0.945, 0.983]$ and slopes $[62, 79]$, and their held-out errors are in Table 4 of Section 4.3. Per-budget stakes standard errors are 0.0017 to 0.0050. Capture pooled over the fifteen budget-seed cells is 0.734, with the cells running from 0.706 to 0.762 and the five per-budget means from 0.718 to 0.753. The cells are five budgets crossed with three seeds on one test set, neither fifteen independent draws nor a sample from a population of budgets. 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 capture range, which is the slope-level statement of the factorization in the main text.

On the same five budgets the length budget alone fits the realized benefit with $R^2 = 0.98$ on $1/T$ and 0.99 on $\log T$, and the held-out error of the stakes, 1.80 points, is 14.0% of the range the outcome spans. The size of the near-optimal start sets stays between 2.0 and 2.5 starts per instance across the sweep, a 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 its held-out error is 2.88 points, 22.4% of the range. With the realized benefit in absolute prize, without any per-budget normalization, the stakes fit falls to $R^2 = 0.04$, because absolute prize scales differ across budgets by construction.

Stakes on the cross-problem and shifted sets. The diamond points of Figure 2 (PCTSP50, OP100 and OPLib) use the sweep's procedure and parameters. PCTSP50 uses the minimization form, stakes $= \overline{C}_{\text{forced}}/C_{\text{ref}} - 1$, 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 its canonical length 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 reference drifts off distribution (0.955 to 0.990 in distribution, 0.92 on OP100, 0.91 on OPLib, 1.00 on PCTSP50). With the light solver on both sides and its unconstrained run alone as the denominator, the shifted-set stakes fall by 7.5 and 7.6 points, to about 0.12 and 0.17, against 1.0 to 3.7 points in distribution with the same denominator. From the stakes as defined, 0.19 and 0.24, the OP50 line predicts one-shot advantages of 11.3 and 14.8 points, and the shifted points would need stakes near 0.09 (OP100) and 0.02 (OPLib) to lie on it. Both sets contain larger instances than the sweep, and across sizes the stakes as defined rise while the available start value falls (Appendix H). On the instances of the scale experiments, at n=100 and the OP100 length budget, they measure 0.190 against 0.170 at n=50, so the OP100 and OPLib points do not stand on the sweep's footing. Section 4.5 separates start value from capture on them with the model's own oracle instead. The outcome measures differ among the three points. The PCTSP50 realized benefit (7.17 ± 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 7.1 points, the same closeness as OP's 7.78/7.66 pair. The OP100 and OPLib realized values are the zero-shot numbers of Table 5 (independently trained uniform control). The check of the available start value decodes the same SAS checkpoints from every start on OP100 and from fifty starts on OPLib, and subtracts the one-shot decode, an oracle-versus-head estimand; Section 4.5 reports the corresponding capture against a uniform start on the same checkpoints.

The classical constructor baseline. Best-of-one is the construction stage of the local-search solver of Appendix E, deterministic prize-per-detour greedy insertion, no descent. Best-of-five is its deterministic five-criterion portfolio; best-of-sixteen samples shapes with multiplicative noise 0.15. OP50: gaps against the Compass reference of Table 1 on all 10,000 test instances: 30.2%, 15.0%, 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, five seeds each: one-shot 48.1% to 48.3%, fifty rollouts 32.4% to 34.2%).

Prize structure on a controlled pool. Prize structure changes how much the choice of start matters without removing the choice, and a controlled pool isolates that effect. It consists of synthetic instances from one geometry generator, one size and one budget rule, and varies only the prize scheme across the

three of Kool et al. (2019a), with 150 instances per scheme. A random feasible start reaches 99.5% of the unconstrained optimum on 17 of the 150 constant-prize instances, 1 of the 150 pseudo-random ones and 18 of the 150 distance-based ones. The two schemes whose prizes are deterministic functions of the node's location, constant and distance to the depot, give similar counts, 17 and 18; the pseudo-random prize is drawn independently of location and gives 1. The same measurement on the TSP is 100% by identity (Section 4.3). The three OPLib prize layers of Section 4.5 also share their geometries, sizes and budgets, so they isolate the prize scheme on real geometry in the same way.

The hit rate at T=2.0. At T=2.0 the head's measured hit@1 is 17.8% ± 7.1 against 53% to 63% at the other length budgets. Forcing the light solver, at four restarts against eight unconstrained ones, to begin at the head's argmax start on 200 instances (a separate measurement) gives the corrected rate of Section 4.2, and the same measurement at the canonical length budget is the control. The under-coverage and the near-constant $|S|$ are consistent with one cause, the construction bias of the Appendix E solver, and the hit rates are reported everywhere uncorrected.

H Experiments at larger scales

Training. We train SAS at three scales beyond the main benchmark, n = 100, 200 and 500, three seeds each, with the OP50 recipe of Appendix B unchanged except for the instance size and the length budget, and with N=n rollouts.

Length-budget choice. The length budget is not carried over from n=50, because a fixed T changes the share of nodes that good tours visit as n grows. At each scale we choose T so that the near-optimal visit fraction matches the canonical benchmark's 0.55, measured by the local-search solver of Appendix E at four restarts, on 48 instances at n=100 and 200 and on 64 at n=500. The grid search gives T = 4.0 at n=100 (visit fraction 0.549), T = 5.5 at n=200 (0.535), and T = 9.0 at n=500 (0.554).

Evaluation. Evaluation uses held-out sets of 1,000 instances per scale, generated by the same official script, pinned by checksum and disjoint from training. The quantities are the ones defined in Section 4.2, computed identically at every scale: the uniform baseline is the mean of eight fixed-seed random feasible starts, the oracle is the best over every feasible start (all n at these budgets), and κ is the head's one-shot gain over that baseline as a share of the oracle's. The n=50 entry is measured on all 10,000 test instances.

The normalized benefit and its fit. The normalized benefit is 0.0876, 0.0600, 0.0375 and 0.0196 at $n = 50, 100, 200, 500$. A log-log fit over the three trained scales gives an exponent of -0.695 with a multiplier of 1.48; the n=50 value is the main experiment's reading, listed for comparison. Three points do not support a goodness-of-fit statistic. Section 4.4 decomposes the decline into available start value and capture. In prize units the head's gain over a uniformly random feasible start is +1.81, +2.32 and +3.22 at n = 100, 200 and 500 (standard errors 0.06, 0.10 and 0.42 over three seeds). The available start value, normalized by the uniform-start prize like the realized benefit, is 0.119, 0.084, 0.065 and 0.039 at n = 50, 100, 200 and 500. Capture pooled over the nine scale-seed cells is 0.597, and with a fifty-draw oracle pool, in which both the head's pick and the oracle come from the pool, it is 0.729, 0.618 and 0.584 at n = 100, 200 and 500.

Start-stakes across scales. The start-stakes of Section 4.3 are measured at the budgets of the scale experiments, on the first 200 evaluation instances of each scale, with the definition used there and with two variants that use one solver on both sides. As defined, with the light solver forced at two restarts against the sixteen-restart reference, the stakes rise from 0.170 at n=50 (Table 3: 0.169, from separate solver runs) to 0.190, 0.210 and 0.217 at n = 100, 200 and 500, while the light solver's unconstrained result, at eight restarts, falls from 0.967 to 0.835 of the reference. With the light solver on both sides, sixteen restarts each, the stakes fall from 0.090 to 0.034, and with the reference solver on both sides, its first move locked to the forced start, from 0.091 to 0.024. At n=50 both lie below the oracle start value of Table 3 (0.105), whereas the light-solver variant of Appendix G, with two forced restarts against eight free ones, stays above it. Over the same range the available start value falls by a factor of 3.0 (Section 4.4). Across scales the measure therefore depends on the solvers it uses. As defined it rises with n while the light solver falls behind the

reference, so it does not track the available start value across scale; with one solver on both sides it falls with n, as the available start value does.

I Reimplementation of POMO+

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, where SAS uses 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 13 summarizes both the conceptual axes and the outcome; Table 14 gives all decoding budgets.

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 100 steps per update; it samples $K{=}0.2N$ 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 precaution as SAS), and start distributions used at evaluation by the same decoding-budget rules as every other configuration. The official recipe, three seeds, the test sets, and the decoding-budget accounting of Appendix A are shared with SAS.

Table 13: 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 % and hit@1 in % of instances, three 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 (100-step accumulation)	one term in the POMO loss
one-shot gap, OP50 (reimpl.)	2.60 ± 0.06	3.87 ± 0.02
one-shot gap, PCTSP50 (reimpl.)	3.32 ± 0.10	4.68 ± 0.15
hit@1, OP50 / PCTSP50	55% / 70%	53% / 62%

Outcome. The auxiliary agent reaches lower mean gaps than the 128-parameter head at every decoding budget on both problems, one-shot 2.60% against 3.87% on OP50 and 3.32% against 4.68% on PCTSP50, and it is ahead for every seed except at twenty-five and fifty rollouts on PCTSP50. Both methods use seeds 42, 43 and 44. Two observations bear on this margin. First, at the selection-free decoding 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 decoding budget follows), so part of its one-shot advantage comes from a better decoder, plausibly a training-side effect of its rollout distribution, ten distinct starts each sampled five times instead of fifty starts sampled once. Second, both selectors target the same structure. The agent's argmax start falls 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 0.18 to 0.30 against 0.99). The shortfall of POMO-style models on PCTSP remains, since the reimplementation's

Table 14: Head-to-head at matched decoding budgets (gap %; mean ± std over three seeds; OP50 to Compass 16.17, PCTSP50 to OR-Tools 4.48). Bold follows the rule of Table 1 and here marks the auxiliary agent, which is ahead for every seed at every OP50 decoding budget and at three of the five budgets on PCTSP50; at PCTSP50's twenty-five and fifty rollouts the per-seed differences change sign.

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

one-shot 3.32% also stays above AM's published gap (Section 3.6). Two independent designs gain from repairing the same bias. The auxiliary agent has about 1,700 times the selector parameters, a separate optimizer and accumulation schedule, and its own architecture, and it gains more. Both rank starts from the same sparse near-optimal start sets first, as expected if the bias is real and start selection works by putting those starts first. SAS isolates the effect of restoring step-zero credit with one loss term and 128 parameters; the auxiliary agent changes selector capacity and training loop together, the loop including its rollout distribution, so its further margin cannot be attributed to either.