Learning to Search, Searching to Learn

A Closed-Loop Framework for Large-Scale Routing Problems

Yongji Fu[1] Yi Zhou[2] Gaojie Jin[3] Guanqun Cao[4,†]

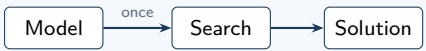
[1]University of Bristol [2]Cardiff University [3]University of Macau
[4]Bristol Robotics Laboratory, University of the West of England, Bristol [†]Corresponding author

NeurIPS 2026

Neural routing at scale: two gaps, one idea

- At $N = 100$K nodes: LKH3 needs **25 h**, and neural solvers cut corners

Typical neural pipeline



• **Gap 1**: scale by partitioning or local candidates, so cross-region structure is lost
• **Gap 2**: search is a one-shot fix, with no feedback to the model

LSSL: a closed loop



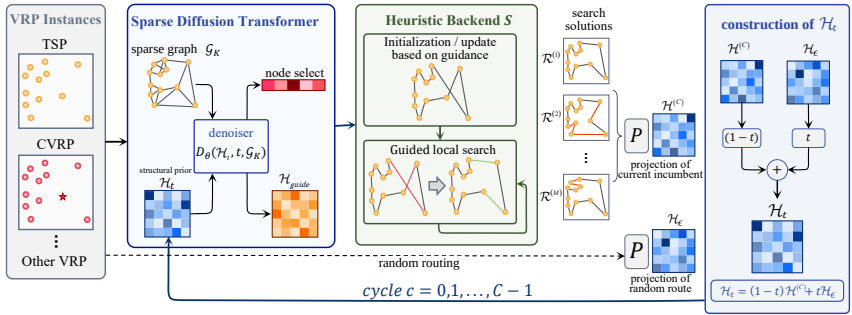
• **Learn where to search**: a heatmap on a sparse graph steers the solver
• **Learn from what search finds**: its results recondition the model and supervise training

100K
nodes, one RTX 4090

0.000%
gap on TSP100K, 6.86 h vs 25 h

-1.630%
gap on CVRP10K, 8.5 min vs 60 min

LSSL: a sparse transformer in a loop with a classical solver



- Each cycle: **project** incumbent → **predict** guidance → **search**; repeat C times
- Backend **S**: Guided-LKH (TSP), Guided-HGS (CVRP); it enforces feasibility
- Sparse $O(NK)$ candidate graph + streamed attention: 100K nodes on one RTX 4090

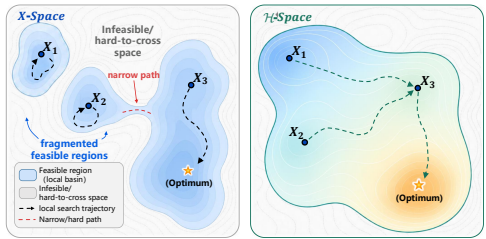
Training on search results, in heatmap space

- Backend returns M good solutions per instance
- Target $\mathcal{H}^\star$: row-normalized edge frequencies (preferred edges *and* alternatives)

$$\mathcal{H}_t = (1-t)\,\mathcal{H}^\star + t\,\mathcal{H}_\epsilon$$

$$\mathcal{L} = -\frac{1}{N}\sum_i \sum_{j\in\mathcal{N}_K(i)} \mathcal{H}^\star_{ij}\log\left[D_\theta(\mathcal{G}_K,\mathcal{H}_t,t)\right]_{ij}$$

- $\mathcal{H}_\epsilon$: random-route projection; CVRP adds a node loss



x-space: early commitment, fragmented basins
$\mathcal{H}$-space: smooth, connected landscape
Conceptual illustration

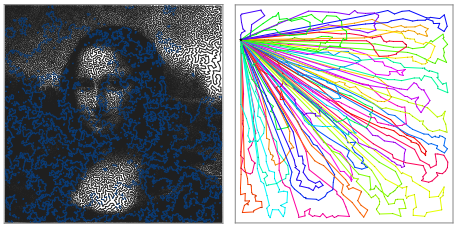
Synthetic benchmarks: as good as the reference, much faster

	TSP1K	TSP10K	TSP100K	CVRP1K	CVRP10K
Mean gap vs. reference (%)					
GLOP	3.119	4.545	5.139	11.019	22.140
L2C-Insert (I=1000)	0.438	2.052	4.971	4.261	8.738
LEHD RRC1000	0.731	12.709	–	2.902	9.916
LSSL (C=4)	**0.000**	−0.006	**0.000**	−0.063	−1.630
Total time on the test set					
Reference (LKH3 / HGS)	30.8 m	33.2 m	25 h	3.5 h	1 h
LSSL (C=4)	**3.6 m**	**3.3 m**	**6.86 h**	**53.3 m**	**8.5 m**

- Same or better than the reference, about 4–10× faster
- GenSCO, Fast-T2T, UDC run out of memory at ≥10K

B_{trials}=512/1024 (TSP), B_t=128 s (CVRP). Negative gap = shorter than reference; zero gap does not certify optimality.

Solutions at scale, and real-world transfer



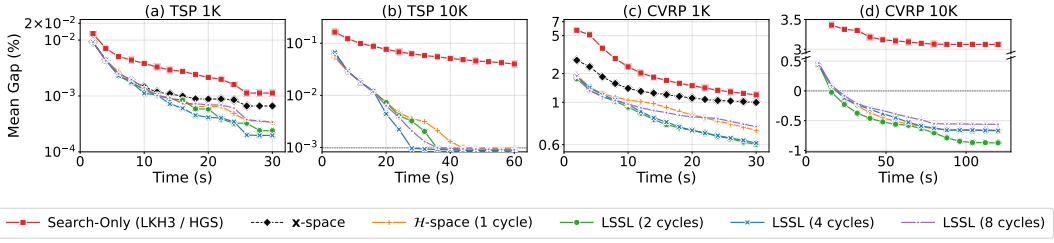
TSP, 100K nodes · CVRP, 1,001 nodes, 43 routes

Trained on uniform synthetic instances only, then applied to TSPLIB, VLSI, Art-TSP and CVRPLIB without fine-tuning

Mean gap	TSPLIB 50–1K	VLSI 2K–200K	Art-TSP 100K+	CVRP-X 100–1K	CVRP-XL 1K–10K
GLOP	1.153%	8.051%	3.238%	–	–
L2C-Insert	1.499%	4.142%	4.240%	3.223%	17.494%
NeuroLKH	0.020%	–	–	2.496%	–
BQ bs16	1.557%	–	–	6.160%	–
LSSL	**0.000%**	-0.299%	**0.026%**	**0.171%**	**1.838%**

TSP: $C{=}2$, $B_{\text{trials}}{=}256$; CVRP: $C{=}4$, $B_t{=}128$ s. Gaps vs. published reference tours; – = not reported.

Does the feedback matter? Yes, at matched cycles



Anytime mean gap (%) vs. time; Search-Only, $\mathbf{x}$-space (1K panels), $\mathcal{H}$-space ($C{=}1$), LSSL $C \in \{2, 4, 8\}$

32 s, $C{=}2$	TSP1K	CVRP1K
Frozen guidance	0.00037%	0.6877%
Refreshed	0.00024%	0.5942%

- Same restarts, seeds, budget splits
- $C{=}2$ is best on CVRP1K/10K: too many refreshes fragment the budget

Takeaways

- **Guidance + feedback**: a sparse neural prior in a loop with a strong discrete solver
- Matches or beats LKH3 / HGS on 10K–100K instances at a fraction of the time, on one GPU
- Refreshing guidance from the incumbent helps at matched cycle structure
- **Limits**: instantiated for TSP and CVRP; other constraints need new features, data, and a feasibility-preserving backend

Thank you!

Code: github.com/yongjifu7/LSSL