# GEPA-TSP: Adapting Lin–Kernighan Heuristics with LLM-Guided Program Search

TODO: Author list

TODO: Submission

**Abstract**

Placeholder abstract summarizing GEPA, sandboxed Concorde integration, and headline gains on structured TSP distributions; include note on reward/prompt sensitivity and VRP pilot.

## 1 Introduction

- Position GEPA as program-level optimization for combinatorial solvers.
- Motivation: adapting Concorde to target distributions (city-like graphs) and OR relevance (VRP).
- Contributions bullets (data, method, results, prompt sensitivity, VRP pilot).

## 2 Related Work

- Classical LK improvements and Concorde heuristics.
- Learning to optimize solvers (learned heuristics, autotuning, RL for OR).
- LLM-based program synthesis / editing for algorithms; reward shaping for LLM agents.

## 3 Method: GEPA for Lin–Kernighan

- Sandbox architecture: copy Concorde, sentinel block injection, build/eval loop.
- Metric: wall time primary, optional multi-objective with BB nodes/timeouts.
- Prompts: student and reflector roles; ablation knobs (wall-time-only vs multi-objective).
- C89 constraints and safety (no globals, no I/O, bounded buffers).

## 4 Benchmarks and Data

- Existing splits: toy20, toy200, tsplib_random (explicit matrices).
- New structured TSP distributions: grid/Manhattan, clustered depots, blocked-edge maps; generation details and seeds.
- VRP pilot set (e.g., CVRPLIB subset or synthetic capacitated instances) and chosen solver hook.

## 5 Experimental Setup

- Models: student/reflector choices (gpt-5-mini/nano, kimi-k2-thinking), token budgets, temperature.
- GEPA budgets: steps, reflection batch, repeats per instance, timeouts.
- Evaluation protocol: repeats, splits, run_root layout; hardware summary.
- Baselines: Concorde default LK, solver defaults for VRP; prompt variants.

## 6 Results: TSP Adaptation

- Core table: baseline vs GEPA on each distribution (wall time, BB nodes, timeout rate).
- Rollout plots (wall time + BB nodes) with smoothing; discuss trends.
- Structured distributions: adaptation gains and generalization across seeds.
- Ablation: steps vs improvement; effect of batch size, reward shaping on LK edits.

## 7 Results: Prompt and LM Sensitivity

- Compare wall-time-only vs multi-objective reflector prompts; trade-offs.
- LM swap study: gpt-5-mini/nano vs kimi-k2-thinking; stability and regressions.
- Case studies of learned heuristic blocks (qualitative differences).

## 8 Results: VRP Pilot

- Solver integration description (injection point) and limitations.
- Metrics: wall time, solution quality gaps vs baseline heuristic.
- Early findings; note if gains are modest or distribution-dependent.

## 9 Discussion

- When GEPA helps most: structured vs unstructured instances.
- Reward design pitfalls (BB-node focus increasing wall time).
- Safety/robustness: build failures, sandboxing, deterministic seeds.
- Practical deployment: adapting to a city map with few GEPA steps.

## 10 Conclusion

- Summary of gains and insights.
- Future work: richer OR tasks, automatic reward balancing, caching/dedup.
- Release plan: code, datasets, candidate blocks.