Accelerating Vehicle Routing via AI-Initialized Genetic Algorithms

Ido Greenberg NVIDIA
Piotr Sielski NVIDIA
Hugo Linsenmaier NVIDIA
Rajesh Gandham NVIDIA
Shie Mannor NVIDIA

Alex Fender NVIDIA
Gal Chechik NVIDIA
Eli Meirom NVIDIA

Abstract

This work introduces a novel hybrid method combining Reinforcement Learning and Genetic Algorithms to solve Vehicle Routing Problems (VRP). While Machine Learning approaches have been extensively applied to VRP, they have struggled to surpass state-of-the-art optimization methods. Our approach bridges this gap by leveraging the strengths of both ML and traditional optimization techniques. We also design a novel regularization method for learning to minimize the number of vehicles (in addition to travel distance), as required in many applications. Our method presents state-of-the-art solution costs given limited optimization time budgets, and scales to hundreds of locations within seconds.

1 Introduction

The Traveling Salesperson Problem [Robinson, 1949] and its extension, the Vehicle Routing Problem (VRP, Dantzig and Ramser [1959]), are a cornerstone of combinatorial optimization, with profound implications across industries such as logistics [Jin, 2020] and urban planning [Dowds et al., 2013]. The theoretical and practical importance of VRP led to significant effort to develop fast and scalable algorithms for approximate solutions, such as Adaptive Large Neighborhood Search (ALNS) [Voigt, 2025] and iterated local-search solvers [Helsgaun, 2017]. In particular, Genetic Algorithms (GA) achieve many current state-of-the-art (SOTA) results [Akif Çördük et al., 2024, Vidal, 2022, Santini and V., 2023]. Yet, the NP-hard problem of VRP remains an enduring challenge for nearly 200 years [Anonymous, 1832].

VRP often presents a trade-off between computing time and solution quality. Solving VRP for hundreds of locations within seconds has become increasingly important for real-world applications [Jin, 2020, James et al., 2019, Bertsimas et al., 2019, Crane et al., 2024], motivating early optimization stopping with an approximate solution rather than optimizing longer. Real-time applications include last-mile delivery routing with on-the-fly updates; routing emergency vehicles under changing road conditions where every second may matter; interactive "what-if" logistics planning with sub-second response times; and ride-sharing services. In real-time warehouse order picking, for example, optimization budget is bounded by the countdown to the next assignment – hence the time budget may vary greatly between different instances. Iterative methods like GA naturally support early optimization stopping with any given time budget.

Importantly, VRP instances are often solved not in isolation, but rather in a "repeated-VRP" way, where multiple instances from the same distribution have to be solved. For example, delivery plans within a given region may involve hundreds of related instances daily, all sharing the same roads and with customer requests from the same distribution. Popular VRP solvers like GA and ALNS apply

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Differentiable Learning of Combinatorial Algorithms.

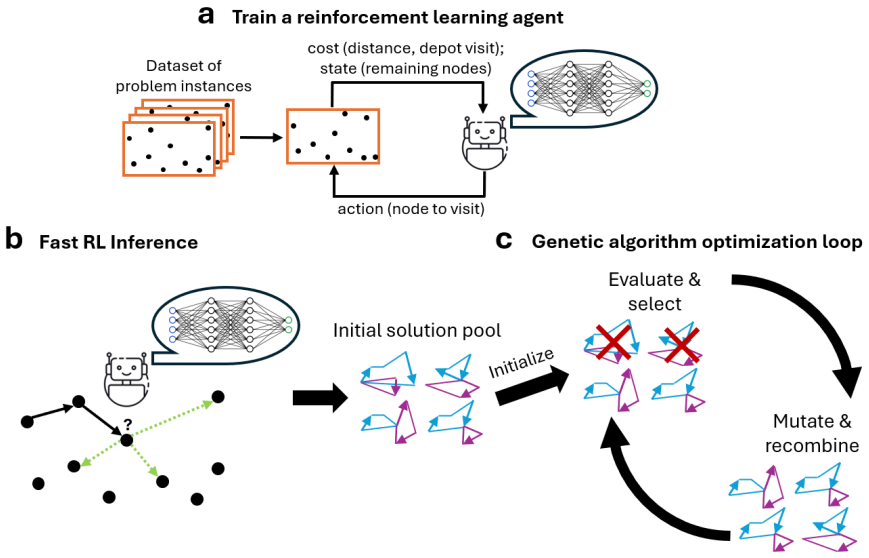

Figure 1: EARLI: Evolutionary Algorithm with Reinforcement Learning Initialization. a, During offline training, an RL agent interacts with a dataset of problem instances and learns to generate high-quality solutions. b, In production, the trained RL agent faces a new problem instance and generates K solutions with quick decision making. c, The K solutions are used as the initial population of a genetic algorithm (GA), initiating its optimization loop.

iterative refinement of approximate solutions, but do not benefit from the repeated-VRP settings: they solve each instance in isolation without learning from other instances. In contrast, recent approaches based on Machine Learning (ML) and Reinforcement Learning (RL) leverage the repeated-VRP settings and train a solver using an offline dataset of VRP instances [Nazari et al., 2018, Kwon et al., 2020, Zhou et al., 2023, Ma et al., 2024, Bengio et al., 2021]. The trained solver can be applied to new instances and produce solutions very quickly, but current ML solvers still yield lower solution quality compared to GAs, in particular in large-scale [Berto et al., 2023, Ma et al., 2024]. The challenge still remains how to achieve superior solutions faster by combining (a) leveraging offline data for fast inference, and (b) enable the user to produce solutions for any point on the trade-off curve of solution quality vs. optimization time.

To that end, we present the framework ***Evolutionary Algorithm with Reinforcement Learning Initialization*** (***EARLI***) for solving the NP-hard combinatorial optimization problem of repeated-VRP. To address the problem in real time and large scale, EARLI is designed to leverage the key strengths of both its components: the RL agent benefits from training data, and the GA provides control over the trade-off between inference time and solution quality. The workflow is illustrated in Figure 1. After training an RL agent over a set of problem instances (Figure 1a), EARLI uses the pre-trained RL agent to quickly generate high-quality (but sub-optimal) solutions (Figure 1b); these are then used as the seed population for a GA process (Figure 1c), which improves the solutions.

We found two main challenges for making EARLI generate initial solutions that genuinely help the GA search. First, the number of vehicles is often the dominant objective in practical VRP, yet this is a challenging metric to optimize: it is sparse, non-continuous, and often ignored in RL methods that focus solely on distance optimization. When initial solutions use too many vehicles, the GA may waste time or even converge to suboptimal fleet sizes. We developed a capacity-aware regularization scheme that guides the RL policy toward using fewer vehicles, outperforming a naïve penalty over the number of vehicles. Second, scaling common RL approaches is hard. They typically target up to 100 customers, as direct training on larger problem instances is prohibitively slow. We scaled model fine-tuning to significantly larger problems using a curriculum learning protocol.

Many methods have been proposed for initializing GAs [Alkafaween et al., 2024, Kazimipour et al., 2013, Rahnamayan et al., 2007, Yang, 1997], including ML approaches [Li et al., 2024]. Most initialization methods substantially differ from EARLI, as they aim to maximize the quality of the

end solution rather than solver speed, and do not use the repeated-VRP settings. For example, some popular ML initialization methods apply test-time training without learning from offline instances [Ruan et al., 2024, Cai et al., 2019, Mundhenk et al., 2021, Radaideh and S., 2021]. As another example, common metaheuristics for initialization focus on covering the solution landscape to avoid local-optima that would limit the end solution [Kazimipour et al., 2013, Rahnamayan et al., 2007]. Some learning approaches, like the Deep Ant Colony Optimization solver (DeepACO) [Ye et al., 2023], do exploit offline data in a repeated-VRP manner and generate an edge-heatmap using RL guidance. Our experiments below show that GA solvers strongly outperform this approach.

EARLI substantially improves SOTA performance on VRP with limited time-budget and scales to instances with hundreds of delivery points, including instances derived from real delivery data. This improvement is robust across several GA solvers, data sources, problem sizes and optimization time budgets – from sub-second up to minutes into the optimization process. In some settings, EARLI achieves in 1s the same solution quality that takes the GA over 10s to reach. Such 10x speedup can significantly enhance existing applications and even enable new use-cases of few-second optimization, for both interactive scientific research and practical applications as discussed above.

In summary, our work introduces a framework for accelerating combinatorial optimization by integrating iterative solvers with learning from past experience. It provides high-quality routing solutions at speed and scale previously considered impractical. This capability can cut costs in classic industries and enable the emergence of applications like interactive logistics planning. With the release of both code and data, our framework opens new avenues for future work in both ML and optimization communities, encouraging their synergy in NP-hard optimization in general and routing problems in particular.

2 Problem setup

In the *Capacitated Vehicle Routing Problem* (VRP or CVRP), a problem instance is defined over a graph $G = (V, E)$, where the vertices V represent a depot and delivery points, and the edges E represent travel costs (e.g., distances). Each delivery i is associated with a known demand d_i. A fleet of K vehicles with capacity Q is stationed at the depot (vertex $i = 0$). A solution is defined by a partition that assigns each vertex to a vehicle route, and by the order of deliveries within each route. In a feasible solution, all routes start and end at the depot, each customer is visited exactly once, and the total demand in each route does not exceed Q. The goal is to find a feasible solution that minimizes first the number of vehicles, and subject to that, the total travel cost. Solvers are typically evaluated under a time budget t given per instance $I = (G, \{d_i\}, K, Q)$.

We define the *repeated-VRP* problem, where instances are drawn from a fixed distribution $I \sim \mathcal{D}$. The distribution $\mathcal{D}$ may express recurring structure, such as distances derived from the same road network, or consistent demand scales and vehicle capacities. $\mathcal{D}$ is unknown but can be sampled offline for training. During inference on new instances, the solver may leverage pre-learned knowledge without reducing its time budget.

3 Method

EARLI leverages offline data of instances $\{I\} \sim \mathcal{D}$ and trains an RL agent to generate high-quality solutions (Figure 1a). In inference time, the agent quickly generates solutions and feeds them as the initial population of the GA, which refines them iteratively (Figure 1b-1c).

In the RL environment, the state at step τ is defined by the instance $I = (G, \{d_i\}, K, Q)$, the set of already-visited vertices, and the remaining capacity in the current route. The action is the next vertex to visit. The reward is the travel cost to the next vertex, possibly with additional regularization as discussed below. We train a parameterized policy $\pi_\theta(a \mid s)$ based on an Attention Model similar to Kwon et al. [2020]. The policy outputs a distribution over feasible actions. A stochastic solution is generated by iteratively sampling this distribution, and a deterministic solution by choosing the action with highest probability. We optimize π_θ using PPO [Schulman et al., 2017], relying on a learned value function to reduce variance and accelerate training.

In inference time, the RL agent generates 1 deterministic solution and 7 stochastic ones per problem. The local-search operator of Vidal et al. [2012] is applied to each solution. The resulting solutions are

then fed to the solver as its initial population. If the solver only permits fewer than 8 solutions (LKH3), we choose the lowest-cost solutions. If the solver's initial population is larger than 8 solutions (HGS and PyVRP), we let the solver fill in random solutions using its internal implementation, up to its standard initial population size. Below we discuss two main challenges in our approach.

Scalable RL training via curriculum learning: Scaling RL to large VRPs is often unstable and prohibitively slow in practice: direct training on 500-customer instances failed to learn in our setup. Standard RL techniques for variance reduction in long horizons, like Generalized Advantage Estimation (GAE), failed to produce reliable learning. We therefore adopt a variant of curriculum learning [Soviany et al., 2022]: train on 50-node instances and gradually increase size, fine-tuning at each stage on new larger instances. This enables stable learning at tractable training times. In addition, unlike alternative curriculums for VRP [Ma et al., 2021], our method naturally produces a family of size-specific models that can be used for different problems, as demonstrated in our experiments.

Vehicle minimization via capacity regularization: Most RL works for VRP focus on minimizing cost or distance, ignoring the number of vehicles [Nazari et al., 2018, Kwon et al., 2020]. By contrast, many practical problems prioritize vehicle minimization, as supported by common solvers [Akif Çördük et al., 2024]. We found empirically that initial solutions with sub-optimal vehicles often slow down or even prevent the solver from optimizing the vehicles. As detailed below, we address this by (a) encourage the RL agent to minimize the number of vehicles; (b) only feed the solver with initial solutions whose number of vehicles is guaranteed to be optimal.

(a) To encourage the RL agent to minimize vehicles, we first attempted straight-forward vehicle penalty on training: every time the agent adds a new vehicle, a fixed cost is added. We found this approach limited, especially on large problem instances. We hypothesize the feedback is too delayed to be learned. For example, consider the last order of the first vehicle in a solution: if its demand does not utilize the remaining vehicle capacity, the immediate vehicle penalty remains the same, yet the capacity waste may lead to an additional vehicle in the end. This extra penalty is only observed hundreds of steps later.

Instead, we propose a novel capacity regularization penalty. At the end of each vehicle route, we add a penalty proportional to the remaining capacity in the vehicle:

$$penalty(vehicle\ route) = \lambda \left(Q - \sum_{i \in route} d_i \right),$$

where Q is the vehicle capacity, d_i are the route demands and λ is the regularization scale coefficient. This provides immediate feedback on capacity waste, allowing the RL agent to strategize vehicle minimization without propagating penalties throughout hundreds of steps. As demonstrated below (Figure 5), capacity regularization significantly improves vehicle minimization compared to vehicle regularization.

(b) To verify that the solver is fed only solutions with an optimal number of vehicles, we recall that the minimal number of vehicles for a VRP instance is lower bounded by $\lceil \frac{total\ demand)}{capacity} \rceil$. In the experiments, our RL solutions have met this lower bound in 87-96% of the problem instances (depending on the benchmark). Hence, in all these instances, at least one RL solution was guaranteed to obtain the optimal number of vehicles. We fed EARLI's initial solutions to the GA solver only in these cases. In the remaining instances, we used instead the nearest-neighbor solution for initialization; and if it did not meet the lower bound either, we simply executed the solver without initialization, with the remaining time-budget.

4 Results

To evaluate EARLI in a realistic challenging scenario, we introduce a new VRP benchmark derived from e-commerce data, in addition to the standard synthetic benchmark in the literature of ML for VRP [Nazari et al., 2018, Kwon et al., 2020]. The standard synthetic benchmark consists of up to 100 customers, with uniformly distributed locations and demands, and Euclidean traveling distances (as illustrated in Figure 2a). However, most real-world instances are fundamentally different: customers

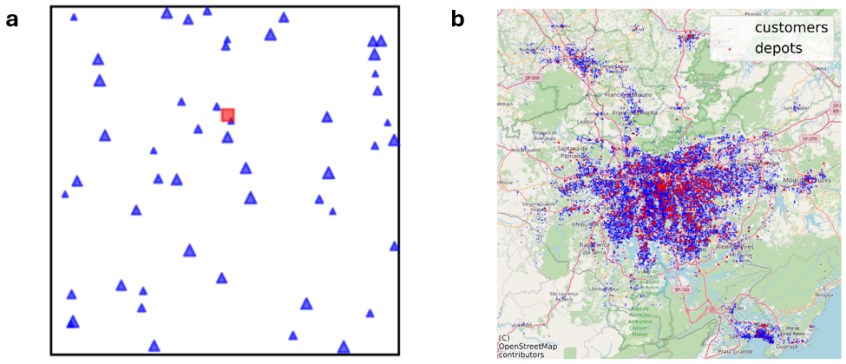

Figure 2: Synthetic and real datasets used in this study. a, Synthetic data: An illustration of a problem instance with 49 customers and 1 depot. Locations are uniformly distributed and travel distances are Euclidean. Random customer demands are represented by the triangle sizes. b, Real data: Olist orders, 100km^2 around Sao Paulo center, include locations of 23K customers and 1K sellers. For every problem instance, multiple customers and one depot are sampled from these locations, respectively. Travel costs correspond to driving time computed with OSRM. Demands correspond to actual product volumes.

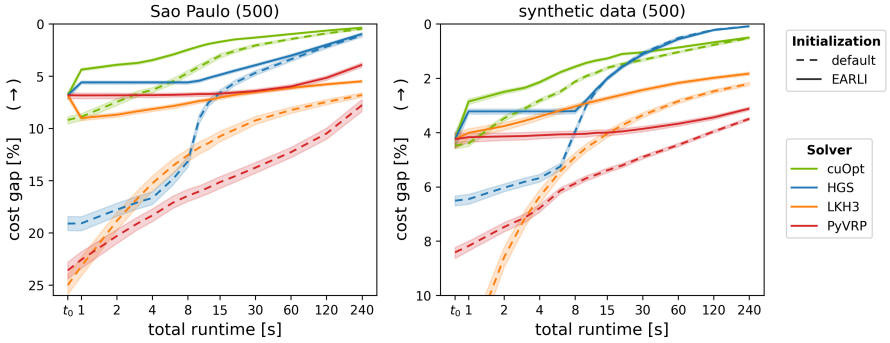

Figure 3: EARLI improves solution quality given a fixed time-budget. EARLI (solid lines) improves the mean cost across a variety of optimization times from seconds up to minutes. Cost gaps are averaged over 256 test instances of 500 customers. Shading corresponds to 95% confidence intervals. t_0 corresponds to the runtime of the RL initialization on its own, before applying the GA.

are often located in clusters of varying sizes, and driving times vary according to the roads, not necessarily even being symmetric.

In our real-data benchmark, the locations of customers and of the depot are sampled from a real-world dataset [Olist and André Sionek, 2018], as visualized in Figure 2b. Driving durations between locations are computed based on real roads, using Project OSRM [Luxen and Vetter, 2011]. Demands are derived from real order volumes.

We evaluate EARLI when applied to 4 popular VRP solvers: HGS [Vidal, 2022], cuOpt [Akif Çördük et al., 2024], PyVRP [Wouda et al., 2024] and LKH3 [Helsgaun, 2017]. The first three are based on GAs, and LKH3 relies on an iterative local-search operator. HGS and cuOpt report SOTA results on various routing problems [Vidal, 2022, Akif Çördük et al., 2024]. By default, all 4 solvers initialize their population with random solutions. We test each solver with its own random initialization; with our proposed RL initialization; and with various alternative initialization methods as specified below. For each of the 4 solvers, the initialization schemes are compared over 256 test instances across a range of time budgets. For every time budget, we show the gap between the obtained cost and the best-known solution cost, defined as the lowest cost amongst all solvers, initialization schemes and time budgets.

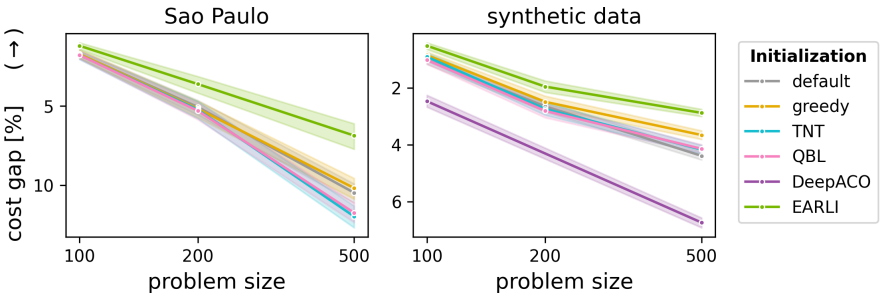

Figure 4: EARLI outperforms alternative initializations, and its benefit grows with problem size. The solver (cuOpt or DeepACO) is given a time budget of 1s per problem. Shading corresponds to 95% confidence intervals over 256 test instances.

Overall, as presented in Figure 3, EARLI obtains state-of-the-art solution costs across a wide range of time budgets in both real data and synthetic data benchmarks, improving all 4 solvers up to minutes into the optimization process.

As EARLI combines an RL agent with a GA solver, it strongly outperforms each of the two on their own. Consider for example the real-data problem instances of 500 customers in Figure 3. When compared to the RL agent alone, EARLI with HGS or cuOpt solvers improves solution quality almost immediately, and the improvement grows over time. When compared to the GA-based HGS solver after 6s, for example, EARLI improves the solution quality by 7.7%. In fact, it achieves within 1s the same average solution quality that takes HGS over 10s to reach with its default initialization, obtaining 10x optimization speedup in this scenario.

Our experiments focus on problem sizes of 100-500 customers, a regime that poses a significant challenge for existing solvers given limited time budget. As presented in Figure 4, the advantage of EARLI holds for different problem sizes and increases with the size, as larger instances pose a harder challenge for the GA solver. We further compared EARLI to several alternative initialization methods: (a) the default random initialization; (b) a nearest-neighbor initialization procedure, shown effective for routing problems [Shanmugam et al., 2013]; (c) TNT and (d) QBL, found to perform best among the candidates evaluated in [Kazimipour et al., 2013]. We further consider (e) DeepACO [Ye et al., 2023], which also uses RL to learn from data and guide an Ant Colony Optimization solver. In contrast to the other methods, DeepACO is a complete pipeline that includes its own ACO solver rather than a GA initialization scheme. We test DeepACO following the settings of [Ye et al., 2023], namely, synthetic problem instances with 100 or 500 customers. As shown in Figure 4, EARLI provides significant value over the compared methods.

Finally, we evaluate our capacity regularization method for vehicle minimization, as a key component of the RL agent. To that end, we take an RL agent pre-trained for 200 customers in Sao Paulo, and fine-tune for 500 customers. For each regularization method, fine-tuning has a trade-off between vehicle minimization and net cost minimization, depending on the regularization coefficient λ. As shown in Figure 5, capacity regularization produces a superior trade-off curve compared to direct vehicle regularization. In addition, vehicle regularization provides only limited control over the trade-off, as increasing λ does not necessarily improve vehicle minimization.

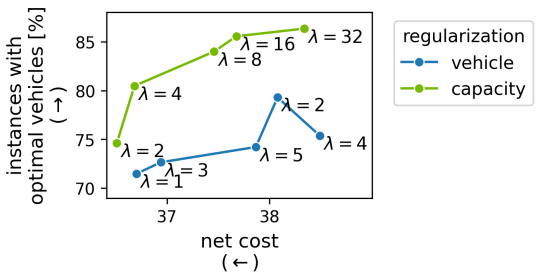

Figure 5: Capacity regularization improves the trade-off between vehicle minimization and net cost minimization. Every point represents fine-tuning of the RL agent with a different regularization scale λ, tested over 256 instances with 500 customers.

References

Akif Çördük, Piotr Sielski, and Moon Chung. Record-breaking nvidia cuopt algorithms deliver route optimization solutions 100x faster, 2024. URL https://developer.nvidia.com/blog/record-breaking-nvidia-cuopt-algorithms-deliver-route-optimization-solutions-100x-faster/. NVIDIA.

E. Alkafaween, A. Hassanat, E. Essa, and S. Elmougy. An efficiency boost for genetic algorithms: Initializing the ga with the iterative approximate method for optimizing the traveling salesman problem. *Applied Sciences*, 2024.

Anonymous. *The traveling salesman – how he should be and what he has to do to receive orders and to be sure of a happy success in his business*. 1832.

Y. Bengio, A. Lodi, and A. Prouvost. Machine learning for combinatorial optimization: a methodological tour d'horizon. *European Journal of Operational Research*, 2021.

F. Berto et al. Rl4co: an extensive reinforcement learning for combinatorial optimization benchmark. In *NeurIPS 2023 GL Frontiers Workshop*, 2023.

D. Bertsimas, P. Jaillet, and S. Martin. Online vehicle routing: The edge of optimization in large-scale applications. *Operations Research*, 2019.

Q. H. Cai, W. M., A. T., G. W., J. W., and W. W. Reinforcement learning driven heuristic optimization. In *Workshop on Deep Reinforcement Learning for Knowledge Discovery*, 2019.

T. Crane, B. K. Hosseini, P. Singh, and R. Sitter. Down to the last mile: Revolutionizing route optimization with nvidia cuopt, 2024. Slalom.

G. B. Dantzig and J. H. Ramser. The truck dispatching problem. *Management Science*, 1959.

J. Dowds, J. Sullivan, D. Scott, D. Novak, et al. Optimization of snow removal in vermont, 2013.

K. Helsgaun. An extension of the lin–kernighan–helsgaun tsp solver for constrained traveling salesman and vehicle routing problems, 2017.

J. James, W. Yu, and J. Gu. Online vehicle routing with neural combinatorial optimization and deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, 2019.

H. Jin. Next-generation optimization for dasher dispatch at doordash, 2020. URL https://careers.doordash.com/blog/next-generation-optimization-for-dasher-dispatch-at-doordash/. DoorDash Blog.

B. Kazimipour, X. Li, and A. K. Qin. Initialization methods for large scale global optimization. In *IEEE Congress on Evolutionary Computation*, pages 2750–2757, 2013.

Y.-D. Kwon, B. Kim, I. Yoon, Y. Gwon, and S. Min. Pomo: Policy optimization with multiple optima for reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 21188–21198, 2020.

P. Li et al. Bridging evolutionary algorithms and reinforcement learning: A comprehensive survey on hybrid algorithms. *IEEE Transactions on Evolutionary Computation*, 2024.

D. Luxen and C. Vetter. Real-time routing with openstreetmap data. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 513–516, 2011.

Y. Ma, Z. Cao, and Y. M. Chee. Learning to search feasible and infeasible regions of routing problems with flexible neural k-opt. In *Advances in Neural Information Processing Systems*, 2024.

Y. Ma et al. Learning to iteratively solve routing problems with dual-aspect collaborative transformer. In *Advances in Neural Information Processing Systems*, 2021.

K. Mundhenk, T. N., L. M., G. R., S. C., P. F., D. M., P. B., and K. Symbolic regression via neural-guided genetic programming population seeding. In *NeurIPS*, 2021.

M. Nazari, A. Oroojlooy, L. Snyder, and M. Takac. Reinforcement learning for solving the vehicle routing problem, 2018.

Olist and André Sionek. Brazilian e-commerce public dataset by olist, 2018. URL `https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce`.

Majdi I. Radaideh and K. S. Rule-based reinforcement learning methodology to inform evolutionary algorithms for constrained optimization of engineering applications. *Knowledge-Based Systems*, 2021.

S. Rahnamayan, H. R. Tizhoosh, and M. M. Salama. A novel population initialization method for accelerating evolutionary algorithms. *Computers & Mathematics with Applications*, 2007.

J. Robinson. On the hamiltonian game (a traveling salesman problem), 1949.

Yaqi Ruan, C. W., and J. W. Combining reinforcement learning algorithm and genetic algorithm to solve the traveling salesman problem. *The Journal of Engineering*, 2024.

Alberto Santini and M. S. T. V. D. V. Decomposition strategies for vehicle routing heuristics. *INFORMS Journal on Computing*, 2023.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

M. Shanmugam, M. S. Basha, P. V. Paul, P. Dhavachelvan, and R. Baskaran. Performance assessment over heuristic population seeding techniques of genetic algorithm: benchmark analyses on traveling salesman problems. *International Journal of Applied Engineering Research (IJAER)*, pages 1171–1184, 2013. Research India Publications.

P. Soviany, R. T. Ionescu, P. Rota, and N. Sebe. Curriculum learning: A survey. *International Journal of Computer Vision*, 130:1526–1565, 2022.

T. Vidal. Hybrid genetic search for the cvrp: Open-source implementation and swap* neighborhood. *Computers & Operations Research*, 2022.

T. Vidal, T. G. Crainic, M. Gendreau, N. Lahrichi, and W. Rei. A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. *INFORMS Journal on Operations Research*, 2012.

S. Voigt. A review and ranking of operators in adaptive large neighborhood search for vehicle routing problems. *European Journal of Operational Research*, 2025.

N. A. Wouda, L. Lan, and W. Kool. Pyvrp: a high-performance vrp solver package. *INFORMS Journal on Computing*, 2024.

R. Yang. Solving large travelling salesman problems with small populations. In *Second International Conference on Genetic Algorithms in Engineering Systems*, 1997.

H. Ye, W. J., C. Z., L. H., and L. Y. Deepaco: Neural-enhanced ant systems for combinatorial optimization. In *Advances in Neural Information Processing Systems*, 2023.

J. Zhou, Y. Wu, W. Song, Z. Cao, and J. Zhang. Towards omni-generalizable neural methods for vehicle routing problems. In *International Conference on Machine Learning*, 2023.