State of the Art in Deep Hierarchical Reinforcement Learning and its Application to a Worker-Manager Model for the Periodic Capacitated Vehicle Routing Problem (PCVRP)

1. Executive Summary and Introduction

The optimization of logistics and supply chain networks is undergoing a fundamental transformation, driven by the geometric expansion of global commerce and the increasing complexity of last-mile delivery. The global market for Reinforcement Learning (RL) technologies, which underpins modern autonomous decision-making, is projected to grow from over \$52 billion in 2024 to an estimated \$32 trillion by 2037.[1] Within this expansive domain, the Periodic Capacitated Vehicle Routing Problem (PCVRP) represents a critical "grand challenge"—a problem class that combines the combinatorial hardness of the Traveling Salesman Problem (TSP) with the temporal scheduling constraints of multi-period planning.

Traditional Operations Research (OR) methods, such as exact solvers and hand-crafted metaheuristics (e.g., Lin-Kernighan-Helsgaun or LKH-3), have long served as the gold standard for accuracy. However, they face a scalability crisis; their computational time grows exponentially or effectively polynomially with problem size, rendering them impractical for real-time decision-making in large-scale logistics networks involving thousands of customers.[2] Conversely, early "monolithic" Deep Reinforcement Learning (DRL) approaches, which attempt to solve routing problems end-to-end with a single neural network, struggle with generalization. A model trained on 100 nodes typically fails catastrophically when optimizing a 1000-node instance due to the quadratic complexity of attention mechanisms and the "vanishing gradient" of reward signals over long decision horizons.[3]

To bridge this gap, the field has converged toward **Deep Hierarchical Reinforcement Learning (DHRL)**, specifically the **Manager-Worker** architecture (also known as Feudal Networks). This paradigm mimics human cognition and industrial management by decomposing the intractable global problem into manageable sub-problems. The "Manager" operates at a high level of abstraction, partitioning the customer base spatially and temporally (assigning visits to days), while the "Worker" operates at a low level, executing the precise routing tactics (sequencing stops).[4]

This report provides an exhaustive analysis of the state-of-the-art in DHRL for PCVRP as of 2024-2025. It dissects the theoretical underpinnings of hierarchical decomposition, evaluates the latest neural architectures for both Managers (e.g., Graph Neural Networks, Partitioning Policies) and Workers (e.g., POMO, Sym-NCO), and synthesizes performance data against established benchmarks. The analysis reveals that while monolithic models have plateaued, hierarchical systems are achieving near-optimal solutions with linear inference scalability, effectively solving the "Curse of Dimensionality" that has plagued combinatorial optimization for decades.

2. The Periodic Capacitated Vehicle Routing Problem (PCVRP): Theoretical Framework

2.1 Formal Problem Definition

The PCVRP is a generalization of the standard Capacitated Vehicle Routing Problem (CVRP) extended over a planning horizon $T$. It is defined on a complete undirected graph $G = (V, E)$, where $V = \{0, 1, \dots, N\}$ is the set of vertices (with node 0 representing the depot) and $E$ is the set of edges with associated costs $c_{ij}$ (typically Euclidean distance or travel time).[6]

Unlike the static CVRP, the PCVRP introduces a temporal dimension. Each customer $i \in V \setminus \{0\}$ is characterized by:

1. **Demand ($q_i$):** The quantity of goods to be delivered.
2. **Service Frequency ($f_i$):** The number of times the customer must be visited within the planning horizon $T$.
3. **Allowable Patterns ($R_i$):** A set of valid visit combinations. For example, if $T=5$ (Monday to Friday) and a customer requires $f_i=2$ visits, $R_i$ might be $\{\{\text{Mon, Thu}\}, \{\text{Tue, Fri}\}\}$.

The objective is to select a pattern $r_i \in R_i$ for each customer and construct a set of routes for each period $t \in \{1, \dots, T\}$ such that:

- **Capacity Constraint:** The total demand on any route $k$ in period $t$ does not exceed the vehicle capacity $Q$.[6]
- **Route Continuity:** Every route starts and ends at the depot ($0$).
- **Frequency Satisfaction:** Each customer is visited exactly $f_i$ times according to the selected pattern.
- Objective Function: Minimize the total travel cost over the entire horizon $T$:

  $$\min \sum_{t=1}^{T} \sum_{k \in K_t} \sum_{(i,j) \in A_k} c_{ij}$$

  where $K_t$ is the set of vehicles used in period $t$ and $A_k$ is the set of arcs in route $k$.

2.2 Complexity and the Need for Hierarchy

The PCVRP is NP-Hard, but its complexity is distinct from the CVRP. It effectively nests two NP-Hard problems:

1. **The Tactical Level (Period Assignment):** Assigning customers to days is equivalent to a constrained Bin Packing Problem or a Generalized Assignment Problem. This decision dictates the density and spatial distribution of customers for each day. Poor assignment leads to sparse, inefficient routes.[5]
2. **The Operational Level (Routing):** Once customers are assigned to a specific day $t$, the problem collapses into a standard CVRP for that day.

Standard "flat" neural solvers fail here because they conflate these two levels. A Transformer model trying to predict a sequence of actions for the entire week encounters an action space that explodes combinatorially with $T$. Furthermore, the "credit assignment" problem in RL becomes severe: if a route on Friday is inefficient, it is difficult for a monolithic agent to determine whether the error was a bad routing decision on Friday (Worker error) or a bad pattern assignment made on Monday (Manager error).[3]

This structural complexity necessitates a hierarchical approach. By decoupling the Period Assignment (Manager) from the Routing (Worker), we align the neural architecture with the problem's mathematical structure, enabling more effective learning and generalization.

3. Foundations of Neural Combinatorial Optimization (NCO)

To understand the hierarchical system, we must first establish the state-of-the-art for the "Worker"—the component responsible for solving the routing sub-problems. The period 2020-2025 has seen a shift from basic Pointer Networks to sophisticated, symmetry-aware constructive heuristics.

3.1 The Constructive Paradigm and Attention Models

The foundation of modern NCO is the **Attention Model (AM)** proposed by Kool et al. (2019), which treats routing as a sequence-to-sequence (Seq2Seq) problem. The model constructs a solution node-by-node, using an Encoder-Decoder architecture.

- Encoder: A Transformer network maps the input set of customers $X$ to a set of embeddings $H$.

  $$h_i = \text{Encoder}(x_i)$$

  These embeddings capture the interaction between nodes (e.g., relative distances) using Multi-Head Self-Attention (MHA).8

- Decoder: An autoregressive network that outputs a probability distribution over the unvisited nodes at each step t, conditioned on the current partial tour.

  $$p(y_{t+1} | Y_t, X) = \text{Softmax}(C \cdot \tanh(h_{context}^T W h_{nodes}))$$

  where context typically includes the embedding of the last visited node and the remaining vehicle capacity.10

3.2 State-of-the-Art Worker Architectures (2024-2025)

While the vanilla AM was a breakthrough, it suffers from sensitivity to initialization and limited exploration. Recent advancements have refined this into robust "Worker" candidates:

3.2.1 POMO: Policy Optimization with Multiple Optima

POMO (Kwon et al., 2020) addresses the issue that a CVRP cycle (e.g., 0-1-2-3-0) can be represented by multiple sequences depending on the start node. POMO leverages this by forcing the agent to generate N trajectories in parallel (one starting from each customer).

- **Mechanism:** It uses the REINFORCE algorithm but replaces the high-variance baseline with the average reward of the N generated trajectories.
- **Impact:** This dramatically stabilizes training and serves as the backbone for most hierarchical workers in 2024-2025. It forces the model to learn a robust policy that is not overly dependent on the random seed of the first action.[8]

3.2.2 Sym-NCO: Leveraging Geometric Symmetries

Sym-NCO (Kim et al., 2022) extends POMO by exploiting geometric invariants. A VRP solution should be invariant to rotation, reflection, and translation of the coordinate system.

- **Mechanism:** Sym-NCO trains the model on augmented views of the instance (e.g., rotated by 90 degrees) and enforces consistency in the policy.
- **Performance:** It achieves tighter optimality gaps (approx. 1.23% on standard benchmarks) compared to POMO's baseline (approx. 4.46% in greedy settings), making it a superior choice for the low-level Worker where precision is paramount.[13]

3.2.3 MatNet: Handling Asymmetric Costs

For real-world PCVRPs where travel costs are not Euclidean (e.g., road networks with one-way streets), **MatNet** replaces the node-based Transformer encoder with a matrix-based encoder. This allows the Worker to process the full distance matrix directly, capturing asymmetric relationships that standard Graph Neural Networks (GNNs) might miss.[15]

3.3 Limitations of Flat Workers

Despite these advances, "flat" Workers like POMO or Sym-NCO scale poorly. Their computational complexity is $O(N^2)$ due to the self-attention mechanism. For a 2000-node instance, the memory requirement becomes prohibitive, and the ability of the model to "attend" to relevant distant nodes degrades. This is the precise entry point for Hierarchical RL.[2]

4. Deep Hierarchical Reinforcement Learning: The Manager-Worker Architecture

Deep Hierarchical Reinforcement Learning (DHRL) formally separates the control policy into layers. In the context of PCVRP, we define a **Manager** ($\pi_H$) and a **Worker** ($\pi_L$). The Manager operates in a discrete, coarse-grained state space to make strategic decisions, while the Worker operates in a continuous or fine-grained state space to execute routing.

4.1 The Feudal Network Paradigm

The governing philosophy is derived from **Feudal Reinforcement Learning**.[5] In a Feudal hierarchy:

1. The **Manager** sets a goal $g_t$ or sub-task for the Worker.
2. The **Worker** attempts to satisfy this goal for a duration $k$, receiving an intrinsic reward $r_{int}$ based on goal achievement.
3. The **Manager** receives the extrinsic reward $R_{ext}$ from the environment (e.g., the negative total cost of the solution) only after the Worker completes the task.

For PCVRP, the "Goal" is typically a **Partition** or a **Period Assignment**.

4.2 The "Manager": Partitioning and Period Assignment

The Manager's primary role is **Decomposition**. This transforms a large-scale, multi-period problem into a series of smaller, single-period CVRPs.

4.2.1 Spatial Partitioning Policies

For large-scale instances (e.g., >1000 customers), the Manager must first spatially cluster nodes to fit within the Worker's effective range (e.g., 100 nodes).

- **Algorithm: Hierarchical Learning-based Graph Partition (HLGP)**.[3] The Manager uses a GNN to process the global graph and outputs a cluster assignment for each node.
- **Innovation:** Unlike K-Means, which clusters solely based on distance, a Learned Manager considers **Routability** and **Capacity**. It learns to avoid creating clusters that are spatially compact but demand-heavy (which would require inefficient extra trips to the depot).
- **Mechanism:** The Manager typically outputs a "Center" for each cluster, and a "Regret-based" allocation policy assigns nodes to centers. The regret metric $Regret_{ij} = d(i, c_j) - d(i, c_{best})$ measures the cost of assigning node $i$ to a suboptimal cluster $j$ to balance capacity.[6]

4.2.2 Temporal Partitioning (Period Assignment)

The unique challenge of PCVRP is temporal assignment. The Manager must decide *when* to visit.

- **Action Space:** For each customer i, select a pattern $r \in R_i$.
- **State Representation:** The Manager observes the spatial distribution of customers and their frequency requirements.
- **Objective:** Assign patterns such that the daily workload is balanced and customers on the same day are spatially clustered. If Customer A (North) and Customer B (North) both need Monday visits, they should be assigned to Monday to allow a single vehicle to serve both.[8]

4.3 Differentiable Subset Selection: The Training "Glue"

A critical technical challenge in DHRL is backpropagating the error signal from the Worker (which produces a route length) through the Manager (which makes a discrete assignment). Standard backpropagation cannot flow through the discrete argmax operation used to assign a customer to a day or cluster.

4.3.1 The Gumbel-Softmax Trick

State-of-the-art implementations in 2024-2025 utilize the **Gumbel-Softmax** relaxation (or Concrete distribution) to make this process differentiable.[17]

- Mechanism: Instead of outputting a hard assignment (e.g., "Monday"), the Manager outputs a probability vector π_i over the days. We add Gumbel noise g and apply a softmax with temperature τ:

  $$y_i = \text{Softmax}\left(\frac{\log(\pi_i) + g}{\tau}\right)$$
- **Forward Pass:** As $\tau \to 0$, y_i approaches a one-hot vector (discrete assignment), allowing the Worker to construct routes.
- **Backward Pass:** Gradients flow through the continuous softmax approximation, allowing the Manager to learn which assignments resulted in lower routing costs.

4.3.2 REINFORCE with Baseline

Alternatively, some architectures use the REINFORCE algorithm (Policy Gradient) for the Manager.

$$\nabla J(\theta) = \mathbb{E} \left$$

Here, R is the final total distance returned by the Worker. The baseline b is crucial for variance reduction and is often computed using a greedy rollout of the current policy (similar to POMO).20

4.4 The "Worker": Execution and Feedback

Once the Manager has partitioned the problem (e.g., "Solve Cluster A on Monday"), the Worker executes.

- **Zero-Shot Generalization:** A key advantage of DHRL is that the Worker can be trained once on generic small instances (e.g., random 50-node CVRPs). The Manager effectively "translates" complex PCVRP instances into this standard format. This allows the system to solve 10,000-node problems without ever training on 10,000 nodes directly, simply by decomposing them into 200 chunks of 50 nodes.[22]
- **Dense Rewards:** To aid training, the Worker can provide *intermediate* feedback. For example, if a cluster assignment is impossible to route (e.g., total demand > total fleet capacity), the Worker can return a large penalty immediately, without attempting the expensive routing step.[6]

5. Comparative Analysis of Manager-Worker Architectures

The effectiveness of a DHRL system depends heavily on how the Manager and Worker interact. We compare the leading architectural paradigms found in the recent literature.

Feature	**Decoupled Hierarchy**	**Jointly Trained Hierarchy (End-to-End)**	**Iterative Hierarchy (Improvement)**
Training Strategy	Train Worker first, freeze it, then train Manager.	Train Manager and Worker simultaneously.	Cycle between Manager assignment and Worker routing.
Stability	High. Worker provides a stable reward signal.	Low. Non-stationary target problem.	Medium. Requires careful tuning.
Optimality	Sub-optimal. Manager learns to exploit Worker biases.	**Near-Optimal.** Finds synergies (e.g., specific shapes).	High. Can escape local optima via re-partitioning.
Sample Efficiency	High.	Low (Needs	Medium.

		massive data).	
Representative Papers	5	3	23

Analysis:
The trend in 2025 is shifting towards Joint Training with stabilized gradients (using POMO baselines) or Iterative Hierarchies. The Iterative approach mimics Large Neighborhood Search (LNS): the Manager creates an initial partition, the Worker solves it, and then the Manager "destroys" part of the solution (e.g., swaps nodes between Monday and Tuesday) and asks the Worker to re-solve, keeping changes that improve the cost.23

6. Detailed Benchmarking and Performance Evaluation

To validate the efficacy of DHRL for PCVRP, we examine performance on standard datasets. The primary metrics are **Optimality Gap** (percentage difference from the best-known solution) and **Inference Time**.

6.1 Standard Benchmark Datasets

Rigorous evaluation relies on three main libraries:

1. **CVRPLib (Set X, Set XXL):** The standard for Capacitated VRP. Set XXL contains instances with up to 30,000 nodes, which are the primary testing ground for Hierarchical scalability.[26]
2. **Cordeau Instances (MD-PVRP):** Specific benchmarks for Periodic and Multi-Depot VRPs. These include constraints on frequency and patterns, making them the litmus test for the Manager's period assignment logic.[28]
3. **Real-World Datasets (ORTEC/Loggi):** Industry-derived datasets from the DIMACS challenge, featuring clustered customer distributions and realistic road constraints.[31]

6.2 Performance Comparison: DHRL vs. Baselines

Table 1 summarizes the performance of SOTA methods on large-scale instances (e.g., 2000+ nodes).

Algorithm Class	Method	Optimality Gap (%)	Inference Time (2k Nodes)	Scalability
OR Heuristics	LKH-3 (Helsgaun)	**0.00% (Ref)**	~45 minutes	Poor ($O(N^{2.5})$)

OR Heuristics	Google OR-Tools	~3-5%	~5 minutes	Medium
Flat NCO	POMO (Single)	>15%	< 1 second	Fails (OOD)
Flat NCO	Sym-NCO	>12%	< 2 seconds	Fails (OOD)
Hierarchical NCO	**UDC (Unified Divide-Conquer)**	**0.5 - 1.5%**	**~10 seconds**	**Excellent (Linear)**
Hierarchical NCO	**HDRL (Partition + POMO)**	1.0 - 2.0%	~8 seconds	Excellent

Data synthesized from.[2]

Key Insights:

1. **The Scalability Wall:** Flat models like POMO fail completely on 2000 nodes (Gap > 15%) because they were trained on 100 nodes. They cannot generalize to the larger graph structure.
2. **The HRL Advantage:** Hierarchical models (UDC, HDRL) maintain a gap of roughly 1% relative to LKH-3 but are orders of magnitude faster (seconds vs. 45 minutes). This speed is transformative for dynamic environments where re-planning must occur in real-time.[33]
3. **Manager Contribution:** Ablation studies show that replacing a Learned Manager with a random or K-Means partitioner increases the gap to >5%, proving that the RL-based partitioning policy captures non-trivial structural insights (e.g., balancing capacity vs. distance).[3]

6.3 PCVRP Specific Results (Cordeau Instances)

On Periodic instances, the Manager's ability to balance workload across days is critical.

- **Result:** DHRL models outperform constructive heuristics that assign patterns greedily. By visualizing the "global horizon," the RL Manager creates balanced schedules that reduce the fleet size required, a metric often more valuable than pure distance minimization.[34]
- **Comparison:** Against "Flat" Multi-Task Learning (MTL) approaches, DHRL achieves superior convergence. The hierarchical separation allows the Manager to operate at a lower frequency (solving the assignment once) while the Worker optimizes the route repeatedly, efficient use of computational resources.[3]

7. Emerging Architectures and Future Directions

As of 2025, several frontier technologies are being integrated into the Manager-Worker framework.

7.1 Generative AI and Diffusion Models

Diffusion Models (e.g., Diffusco) are entering the arena as potential "Workers." They formulate routing as a conditional generation task, "denoising" random noise into a coherent route.

- **Benefit:** They naturally handle multi-modal distributions (finding diverse good solutions).
- **Drawback:** Inference is slow due to the iterative denoising steps.
- **HRL Application:** A High-Level Manager could partition the problem, and a Diffusion Worker could generate ultra-high-quality routes for the sub-problems where precision is critical, trading off speed for quality.[35]

7.2 Foundation Models for Routing

The concept of a "Foundation Model" for VRP is gaining traction. Models like **RouteFinder** or **PolyNet** are trained on massive datasets of varying sizes and constraints.

- **Mechanism:** These models use "Adapter Layers" to handle specific constraints (like Time Windows or Periodicity) without retraining the core transformer.
- **Implication:** Future HRL systems might not train a Worker from scratch. Instead, a light-weight RL Manager will simply prompt a pre-trained, frozen Foundation Model to solve the sub-problems it defines.[38]

7.3 Robustness and Stochasticity

Real-world operations are stochastic. A Manager trained purely on static data may create brittle schedules that break if a driver is delayed.

- **Robust HRL:** New research incorporates **Distributional RL** into the Manager. Instead of predicting a single cost for an assignment, the Manager predicts a *distribution* of costs, allowing it to select assignments that minimize "Value-at-Risk" (VaR)—effectively building buffer time into the schedule to handle uncertainty.[40]

8. Conclusion

The application of Deep Hierarchical Reinforcement Learning to the Periodic Capacitated Vehicle Routing Problem represents the cutting edge of Neural Combinatorial Optimization. By abandoning the "one-size-fits-all" approach of monolithic networks in favor of a bio-inspired **Manager-Worker architecture**, researchers have successfully cracked the code on scalability.

The **Manager**, leveraging Graph Neural Networks and differentiable partitioning (Gumbel-Softmax), solves the strategic problem of temporal and spatial decomposition. The **Worker**, utilizing robust constructive heuristics like **POMO** and **Sym-NCO**, solves the operational routing with extreme efficiency.

The synergy of these components allows DHRL systems to solve 10,000-node PCVRP instances with near-optimal quality in seconds—a feat previously impossible with classic OR or flat RL methods. As the field integrates Generative AI and Foundation Models, the gap between "Learning to Route" and "Exact Optimization" continues to vanish, heralding a new era of autonomous, real-time logistics management.

Detailed Technical Appendix: Mechanism of Action

A. The Manager's Policy Gradient (REINFORCE with Baseline)

The Manager maximizes the expected reward $J(\theta_M)$. Since the action A_M (partitioning) is discrete, we approximate the gradient.
Let $R(A_M)$ be the reward (total distance calculated by the Worker) for a given partition A_M.
The gradient update is:

$$\theta_M \leftarrow \theta_M + \alpha \nabla_{\theta_M} \log \pi_{\theta_M}(A_M | S_{global}) \cdot (R(A_M) - b(S_{global}))$$

Here, $b(S_{global})$ is a baseline, typically the moving average of rewards from recent episodes, to reduce variance.

B. The Worker's Attention Mechanism (POMO)

The Worker computes compatibility between the query node (current position) and key nodes (unvisited customers).

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

In POMO, this is augmented by calculating context embeddings $h_{(c)}$ derived from the graph embedding $\bar{h}$:

$$h_{(c)} = [\bar{h}; h_{current\_node}; h_{remaining\_capacity}]$$

This context vector allows the Worker to make decisions that respect the constraints imposed by the Manager (e.g., "you only have 50% capacity left").

C. Hyperparameter Tuning for HRL

- **Learning Rates:** Typically, the Manager requires a lower learning rate ($\approx 1e-5$) than the Worker ($\approx 1e-4$) to prevent non-stationarity (the "moving target" problem where the Manager overfits to a changing Worker).
- **Entropy Regularization:** High entropy regularization is crucial for the Manager to prevent it from collapsing into a single partitioning strategy early in training (e.g., always splitting the map down the middle), ensuring exploration of diverse spatial decompositions.

Citations Integrated: [1]

Works cited

1. The State of Reinforcement Learning in 2025 - DataRoot Labs, accessed December 22, 2025, https://datarootlabs.com/blog/state-of-reinforcement-learning-2025
2. Hierarchical Deep Reinforcement Learning for Vehicle Routing Problem - OpenReview, accessed December 22, 2025, https://openreview.net/forum?id=6G7cF9RNzP
3. [2502.08340] Hierarchical Learning-based Graph Partition for Large-scale Vehicle Routing Problems - arXiv, accessed December 22, 2025, https://arxiv.org/abs/2502.08340
4. Reinforcement Learning Foundations for Deep Research Systems: A Survey - arXiv, accessed December 22, 2025, https://arxiv.org/html/2509.06733v1
5. Hierarchical reinforcement learning in network routing optimization - DiVA portal, accessed December 22, 2025, http://www.diva-portal.org/smash/get/diva2:1955666/FULLTEXT01.pdf
6. A Reinforcement Learning Framework for Scalable Partitioning and Optimization of Large-Scale Capacitated Vehicle Routing Problems - MDPI, accessed December 22, 2025, https://www.mdpi.com/2079-9292/14/19/3879
7. FeUdal Networks for Hierarchical Reinforcement Learning, accessed December 22, 2025, https://proceedings.mlr.press/v70/vezhnevets17a/vezhnevets17a.pdf
8. A Deep Reinforcement Learning Model to Solve the Stochastic Capacitated Vehicle Routing Problem with Service Times and Deadlines - MDPI, accessed December 22, 2025, https://www.mdpi.com/2227-7390/13/18/3050
9. BQ-models with greedy rollouts (G) or Beam Search (bs) versus classic and neural baselines. Underlined methods are rebu - OpenReview, accessed December 22, 2025, https://openreview.net/attachment?id=ZrfcnN1qYV&name=pdf
10. Reinforcement Learning for Solving the Vehicle Routing Problem - Lehigh University, accessed December 22, 2025, https://engineering.lehigh.edu/sites/engineering.lehigh.edu/files/_DEPARTMENTS/ise/pdf/tech-papers/19/19T_002.pdf
11. A Deep Reinforcement Learning-Based Decision-Making Approach for Routing

Problems, accessed December 22, 2025, https://www.mdpi.com/2076-3417/15/9/4951

12. Preference Optimization for Combinatorial Optimization Problems - OpenReview, accessed December 22, 2025, https://openreview.net/forum?id=8QkpCRio53
13. alstn12088/Sym-NCO - GitHub, accessed December 22, 2025, https://github.com/alstn12088/Sym-NCO
14. Sym-NCO: Leveraging Symmetricity for Neural Combinatorial Optimization - arXiv, accessed December 22, 2025, https://arxiv.org/pdf/2205.13209
15. Neural Combinatorial Optimization for Real-World Routing - arXiv, accessed December 22, 2025, https://arxiv.org/html/2503.16159v1
16. Multi-Agent Reinforcement Learning for Connected and Automated Vehicles Control: Recent Advancements and Future Prospects - IEEE Xplore, accessed December 22, 2025, https://ieeexplore.ieee.org/iel8/8856/10839176/11016811.pdf
17. Gumbel-Softmax: Differentiable Discrete Sampling - Emergent Mind, accessed December 22, 2025, https://www.emergentmind.com/topics/gumbel-softmax
18. Reparameterizable Subset Sampling via Continuous Relaxations - IJCAI, accessed December 22, 2025, https://www.ijcai.org/proceedings/2019/0544.pdf
19. Leveraging Recursive Gumbel-Max Trick for Approximate Inference in Combinatorial Spaces - NIPS papers, accessed December 22, 2025, https://papers.nips.cc/paper/2021/file/5b658d2a925565f0755e035597f8d22f-Paper.pdf
20. Reinforcement Learning for Solving the Vehicle Routing Problem, accessed December 22, 2025, http://papers.neurips.cc/paper/8190-reinforcement-learning-for-solving-the-vehicle-routing-problem.pdf
21. Hierarchical Deep Reinforcement Learning for Vehicle Routing Problem - OpenReview, accessed December 22, 2025, https://openreview.net/pdf?id=6G7cF9RNzP
22. Hierarchical reinforcement learning for vehicle routing problems with time windows, accessed December 22, 2025, https://nrc-publications.canada.ca/eng/view/object/?id=e02634fa-53d9-4666-8876-5db877efe04a
23. Neural Combinatorial Optimization Algorithms for Solving Vehicle Routing Problems: A Comprehensive Survey with Perspectives | Request PDF - ResearchGate, accessed December 22, 2025, https://www.researchgate.net/publication/381126299_Neural_Combinatorial_Optimization_Algorithms_for_Solving_Vehicle_Routing_Problems_A_Comprehensive_Survey_with_Perspectives
24. Learning to Segment for Vehicle Routing Problems - arXiv, accessed December 22, 2025, https://arxiv.org/html/2507.01037v2
25. Neural Large Neighborhood Search for the Capacitated Vehicle Routing Problem - Ecai 2020, accessed December 22, 2025, http://ecai2020.eu/papers/786_paper.pdf
26. CVRPLib - Dataset - LDM, accessed December 22, 2025, https://service.tib.eu/ldmservice/dataset/cvrplib

27. Routing Arena: A Benchmark Suite for Neural Routing Solvers - arXiv, accessed December 22, 2025, https://arxiv.org/pdf/2310.04140
28. Results on Cordeau et al. (2001) MDVRPTW instances | Download Table - ResearchGate, accessed December 22, 2025, https://www.researchgate.net/figure/Results-on-Cordeau-et-al-2001-MDVRPTW-instances_tbl2_230846314
29. Cordeau_al_1997_MDVRP - VRP-REP: the vehicle routing problem repository, accessed December 22, 2025, http://www.vrp-rep.org/datasets/item/cordeau-al-1997-mdvrp.html
30. Cordeau_al_2001_PVRPTW - VRP-REP: the vehicle routing problem repository, accessed December 22, 2025, http://www.vrp-rep.org/datasets/item/cordeau-al-2001-pvrptw.html
31. Vehicle routing instances and other resources - Cirrelt, accessed December 22, 2025, https://w1.cirrelt.ca/~vidalt/en/VRP-resources.html
32. UDC: A Unified Neural Divide-and-Conquer Framework for Large-Scale Combinatorial Optimization Problems - NIPS papers, accessed December 22, 2025, https://papers.nips.cc/paper_files/paper/2024/file/0b8e4c8468273ee3bafb288229c0acbc-Paper-Conference.pdf
33. Deep Reinforcement Learning Approach to Solve Dynamic Vehicle Routing Problem with Stochastic Customers, accessed December 22, 2025, https://cdn.aaai.org/ojs/6685/6685-40-9914-1-10-20200521.pdf
34. Hierarchical reinforcement learning for vehicle routing problems with time windows Wang, Yunli - NRC Publications Archive, accessed December 22, 2025, https://nrc-publications.canada.ca/eng/view/ft/?id=e02634fa-53d9-4666-8876-5db877efe04a
35. A Combined Diffusion Model and Reinforcement Learning Approach for Solving the Vehicle Routing Problem With Multiple Soft Time Windows - ResearchGate, accessed December 22, 2025, https://www.researchgate.net/publication/393114462_A_Combined_Diffusion_Model_and_Reinforcement_Learning_Approach_for_Solving_the_Vehicle_Routing_Problem_with_Multiple_Soft_Time_Windows
36. A Combined Diffusion Model and Reinforcement Learning Approach for Solving the Vehicle Routing Problem With Multiple Soft Time Windows - IEEE Xplore, accessed December 22, 2025, https://ieeexplore.ieee.org/iel8/6287639/10820123/11053837.pdf
37. Towards Generalizable Neural Solvers for Vehicle Routing Problems via Ensemble with Transferrable Local Policy - arXiv, accessed December 22, 2025, https://arxiv.org/html/2308.14104v3
38. A Foundation Model for Vehicle Routing Problems 1 INTRODUCTION - TRISTAN 2025, accessed December 22, 2025, https://tristan2025.org/proceedings/TRISTAN2025_ExtendedAbstract_247.pdf
39. PolyNet: Learning Diverse Solution Strategies for Neural Combinatorial Optimization - arXiv, accessed December 22, 2025, https://arxiv.org/html/2402.14048v1

40. Neural Combinatorial Optimization for Robust Routing Problem with Uncertain Travel Times - NIPS papers, accessed December 22, 2025, https://papers.nips.cc/paper_files/paper/2024/file/f30e35a09ac622dec1c121a13dd809d4-Paper-Conference.pdf
41. Solving the Vehicle Routing Problem with Stochastic Travel Cost Using Deep Reinforcement Learning - MDPI, accessed December 22, 2025, https://www.mdpi.com/2079-9292/13/16/3242
42. arXiv:2102.10012v1 [cs.LG] 19 Feb 2021, accessed December 22, 2025, https://arxiv.org/pdf/2102.10012
43. A tale of two goals: leveraging sequentiality in multi-goal scenarios - arXiv, accessed December 22, 2025, https://arxiv.org/html/2503.21677v1
44. Day 80/100: Hierarchical Reinforcement Learning – Teaching Agents to Think in Goals and Subgoals | by Sebastian Buzdugan | Medium, accessed December 22, 2025, https://medium.com/@sebuzdugan/day-80-100-hierarchical-reinforcement-learning-teaching-agents-to-think-in-goals-and-subgoals-05e2e7a419eb
45. Goal-Conditioned Reinforcement Learning - NeurIPS 2025, accessed December 22, 2025, https://neurips.cc/virtual/2023/workshop/66519