The Evolution of Neural Combinatorial Optimization: From Methodological Foundations to Generalized Foundation Models (2018–2026)

1. Introduction: The Algorithmic Paradigm Shift

The landscape of Combinatorial Optimization (CO) is undergoing a profound transformation, driven by the integration of machine learning (ML) technologies into a domain traditionally dominated by exact operations research (OR) methods and hand-crafted heuristics. For decades, problems such as the Traveling Salesman Problem (TSP), Vehicle Routing Problem (VRP), and Job Shop Scheduling (JSP) were tackled using solvers that, while mathematically rigorous, often faced insurmountable scalability barriers or required extensive, problem-specific engineering. The emergence of Neural Combinatorial Optimization (NCO) represents a paradigm shift where the algorithmic burden is transferred from human design to data-driven learning.

This report provides an exhaustive, expert-level analysis of the trajectory of NCO research from 2018 to 2026. By synthesizing findings from foundational surveys and over 100 seminal papers—ranging from the early adoption of attention mechanisms to the latest breakthroughs in Large Language Model (LLM)-driven heuristic design and unified foundation models—we chart the maturation of a field that is redefining the limits of computational efficiency and generalization. The analysis reveals a clear evolutionary arc: from the initial "learning to solve" approaches that treated optimization as a black-box supervised task, to "learning to search" hybridizations, and finally to the contemporary era of "learning to design," where AI systems autonomously architect the very algorithms used to solve complex logistical challenges.

2. Foundational Methodologies and Survey Landscapes (2018–2024)

The intellectual bedrock of modern NCO was laid in the late 2010s, characterized by a formalization of how ML could interface with discrete optimization structures.

2.1 The Methodological Manifesto (2018)

The 2018 survey, *Machine Learning for Combinatorial Optimization: a Methodological Tour d'Horizon* by Bengio et al., serves as the seminal manifesto for the field. It articulated the core premise that optimization instances are not isolated mathematical puzzles but samples drawn from a probability distribution.[1] This insight justified the use of ML to learn policies that exploit

the statistical regularities of these distributions—something classical exact solvers, which optimize for the worst-case scenario, often fail to do. The survey established a critical dichotomy in NCO approaches:

1. **Learning to Configure**: Using ML to tune the parameters of existing algorithms (e.g., branching decisions in Branch-and-Bound).
2. **End-to-End Learning**: Using neural networks, such as Graph Neural Networks (GNNs) or Sequence-to-Sequence models, to output solutions directly.[2]

This work highlighted that while ML might not guarantee optimality like exact methods, it offers a "natural candidate" for making computationally expensive decisions (like variable selection) instantaneously, essentially trading theoretical guarantees for practical speed and average-case performance.[1]

2.2 The Application-Specific Maturation (2022)

By 2022, the focus shifted from general methodologies to rigorous application, particularly in logistics. *Learning to Solve Vehicle Routing Problems: A Survey* (T-ITS, 2022) documented the explosion of NCO techniques specifically for VRP variants.[4] This period saw the solidification of Reinforcement Learning (RL) as the dominant training paradigm, largely because optimal labels for large VRP instances are computationally expensive to generate, making Supervised Learning (SL) less attractive.[5] The survey highlighted the emergence of "Construction-based" vs. "Improvement-based" approaches. Construction models, often based on attention mechanisms, build solutions sequentially, while improvement models (learning to search) refine initial solutions via learned operators (e.g., 2-opt, node swap).[6] The survey noted a critical gap: while NCO showed promise, it struggled to compete with highly optimized metaheuristics like Hybrid Genetic Search (HGS) on standard benchmarks, often limited to small instances (e.g., 100 nodes) due to the $O(N^2)$ complexity of attention mechanisms.[5]

Simultaneously, *A Review on Learning to Solve Combinatorial Optimisation Problems in Manufacturing* (2022) emphasized the industrial necessity of these methods. In dynamic manufacturing environments, problem definitions change rapidly (e.g., machine breakdowns, urgent orders), rendering rigid handcrafted heuristics obsolete. NCO offers the potential for "responsive" solvers that adapt to new data distributions in real-time, a capability critical for Industry 4.0.[7]

2.3 The Era of Troubleshooting and Robustness (2023–2024)

As the field entered 2023 and 2024, the discourse matured to address the systemic failures of early NCO models. *A Survey on Reinforcement Learning for Combinatorial Optimization* (AIC, 2023) and the 2024 comprehensive survey on VRP solvers [9] identified specific "troublemakers" hindering industrial adoption:

- **Poor Generalization**: Models trained on uniform random data failed catastrophically on clustered, real-world data (the "distribution shift" problem).

- **Scale Invariance**: A model trained on 50 nodes could not solve 500-node problems effectively.
- **Constraint Rigidity**: Architectures were often hard-coded for specific constraints (e.g., capacity), requiring complete retraining for variants (e.g., time windows).[9]

The 2024 survey categorized solutions into four distinct taxonomies: Learning to Construct, Improve, Predict-Once, and a meta-category of "Learning to Configure".[10] It called for a move towards "Generalizable Neural Solvers" that decouple the problem logic from the neural architecture, setting the stage for the unified solvers of 2025.[9]

3. The LLM Revolution in Heuristic Design (2025–2026)

The most disruptive innovation in the 2025–2026 period is the integration of Large Language Models (LLMs) not just as solvers, but as *meta-solvers* that design algorithms. This represents a transition from "parameter optimization" (tuning weights) to "code optimization" (writing logic).

3.1 Automated Heuristic Design (AHD)

The premise of AHD is to utilize the code-generation and reasoning capabilities of LLMs to automate the trial-and-error process of heuristic design, traditionally performed by PhD-level experts.

3.1.1 Evolution of Heuristic Sets (EoH-S)

A critical limitation of early LLM-based design (like **EoH** [11] and **ReEvo** [12]) was the search for a *single* omnipotent heuristic. The 2026 paper *EoH-S: Evolution of Heuristic Set using LLMs* challenges this, recognizing that diverse problem instances require diverse strategies.[13]

- **Mechanism**: EoH-S employs an evolutionary framework where the "population" consists of Python code snippets generated by LLMs. Unlike standard evolution that converges to a single solution, EoH-S utilizes a "complementary population management" strategy. It explicitly maintains heuristics that perform well on instances where the current elite set fails.
- **Insight**: This mirrors the "mixture of experts" concept in neural networks but applied to symbolic code. By evolving a *set* of heuristics, EoH-S achieves a coverage of the problem space that no single heuristic can match, effectively automating the design of algorithm portfolios.[13]

3.1.2 Monte Carlo Tree Search for Heuristic Discovery

The 2025 paper *Monte Carlo Tree Search for Comprehensive Exploration in LLM-Based Automatic Heuristic Design* addresses the "local optima" problem in evolutionary AHD. Genetic algorithms often discard promising but immature code snippets.

- **Mechanism**: This work replaces the genetic algorithm with **Monte Carlo Tree Search**

(MCTS). The search space of possible heuristics is modeled as a tree. The LLM acts as the policy, proposing modifications (mutations/crossovers) to existing code.[14]

- **Innovation**: The method introduces "Tree-Path Reasoning," where the LLM analyzes the *sequence* of modifications along a branch to infer why performance improved or degraded. This "hindsight" allows the model to learn design principles dynamically, rather than just blindly mutating code. The use of MCTS allows for "backtracking" and deeper exploration of radical design changes that might initially degrade performance before leading to a breakthrough.[15]

3.1.3 VRPAGENT: Operator Discovery

While EoH-S evolves entire algorithms, *VRPAGENT* (2025) focuses on the granular discovery of *operators* within a Large Neighborhood Search (LNS) framework.

- **Contribution**: VRPAGENT uses LLMs to write "Destroy" and "Repair" operators—the core components of LNS that determine which parts of a route to reshuffle. A genetic algorithm refines these operators based on their contribution to solution quality.[17]
- **Key Finding**: The system discovered novel operators that human experts had not conceived, outperforming handcrafted baselines on CVRP and Prize-Collecting VRP. Notably, this approach is computationally efficient, running on a single CPU core, proving that intelligent code logic can rival GPU-heavy neural computations.[18]

3.2 LLMs as Solvers and Data Generators

Beyond writing code, LLMs are being used to manipulate the problem data itself.

3.2.1 Bridging Synthetic and Real Worlds (EvoReal)

A pervasive issue in NCO is the "Synthetic Gap"—models trained on random coordinates fail on real-world maps. *Bridging Synthetic and Real Routing Problems via LLM-Guided Instance Generation* (2026), or **EvoReal**, utilizes LLMs to generate training data.[19]

- **Mechanism**: Rather than relying on static datasets, EvoReal uses an LLM-guided evolutionary module to synthesize VRP instances. The LLM analyzes the structural properties of real-world data (e.g., clustering, depot centrality) and evolves synthetic instances to mimic these statistical fingerprints.
- **Progressive Adaptation**: A neural solver is then trained on a curriculum of these evolved instances, progressively adapting from simple random graphs to complex, realistic topologies. This "Generator-based Adaptation" achieves state-of-the-art generalization without requiring access to massive proprietary real-world datasets.[19]

3.2.2 Direct Solving

The paper *Large Language Models as End-to-end Combinatorial Optimization Solvers* (2025) explores the limits of LLMs in directly outputting solutions (e.g., "Visit node 1, then 5, then 3").[20] While LLMs struggle with arithmetic precision on large scales, this work suggests they

serve as effective "high-level planners" or solvers for small, semantic-heavy constraints (e.g., "driver must visit a pharmacy") that are hard to encode mathematically.

4. The Rise of Unified and Foundation Models (2025)

The fragmentation of NCO—where a different model is needed for every VRP variant (e.g., CVRP, VRPTW, PDVRP)—is being addressed by "Unified Solvers" that aim to be the "GPT of Routing."

4.1 URS: The Zero-Shot Generalist

URS: A Unified Neural Routing Solver (2025) is a landmark attempt to create a single model capable of solving over 100 VRP variants.[21]

- **Unified Data Representation (UDR)**: URS replaces problem-specific encoders with a standardized input format that can represent any combination of constraints (capacity, time windows, pickup-delivery) as feature channels.[22]
- **Mixed Bias Module (MBM)**: To handle the diverse geometric relationships across variants, URS employs an MBM that learns relational biases (e.g., how "tight" a time window is relative to distance).
- **Significance**: This model achieves zero-shot generalization to *unseen* variants. For example, a model trained on CVRP and TSP can solve a VRP with Backhauls without fine-tuning, simply by interpreting the unified data input. This moves NCO closer to a "Foundation Model" paradigm.[22]

4.2 RouteFinder and Foundation Architectures

Similarly, *RouteFinder* (2024/2025) proposes a foundation model framework using "Global Attribute Embeddings".[23] It argues that VRP variants should be treated as subsets of a "Universal VRP." By learning embeddings for attributes like "Time Window" or "Open Route," the model can compose these embeddings to understand complex new problem types.

- **Uni-Environment**: RouteFinder introduces a parallelized environment capable of simulating any combination of attributes, enabling "Mixed Batch Training" where the model sees diverse problem types in a single training batch, preventing overfitting to one specific logic.[24]

4.3 SHIELD: Structural Decomposition

SHIELD (2025) takes a theoretically rigorous approach to unification via **State-Decomposable MDPs (SDMDP)**.[25]

- **Mechanism**: SHIELD posits that complex VRPs are compositions of "basis" problems. It decomposes the state space into a Cartesian product of basis states. The solver learns "basis policies" (experts) for fundamental constraints (e.g., a "Capacity Expert", a "Time Window Expert") and uses a hierarchical gating mechanism to mix their outputs

dynamically.[25]

- **Benefit**: This sparsity and hierarchy allow the model to scale efficiently. Instead of a dense giant model, it activates only the relevant experts for the specific constraints present in the current instance, enhancing both efficiency and interpretability.[26]

5. Architectural Innovations: Scale, Modality, and Efficiency

To support these unified models, the underlying neural architectures have evolved beyond standard Transformers.

5.1 Hierarchical Scaling (Scale-Net)

The $O(N^2)$ complexity of attention maps limits most NCO models to <100 nodes. *Scale-Net* (2026) introduces a **Hierarchical U-Net Framework** for routing.[27]

- **Architecture**: Inspired by computer vision U-Nets, Scale-Net processes the graph at multiple resolutions. It uses a "coarsening" phase to aggregate nodes into clusters (solving the global structure) and a "refinement" phase to detail the local routes.
- **Impact**: This allows the model to maintain global context without the prohibitive cost of full attention, enabling effective generalization to large-scale instances (e.g., 1000+ nodes) that were previously intractable for end-to-end neural solvers.[27]

5.2 Efficiency via Recurrence (Recurrent State Encoders)

Recurrent State Encoders (2025) attacks the latency bottleneck.[28] Standard constructive solvers re-encode the entire graph embedding at every step (N times).

- **Insight**: The state change between step t and $t+1$ is minimal (one node becomes "visited"). Re-computing the full embedding is redundant.
- **Solution**: The authors propose a recurrent update rule where the embedding at step $t+1$ is a function of the embedding at step t and the action taken. This reduces the depth of the encoder required and slashes inference latency by 4x without sacrificing accuracy, making NCO viable for real-time dispatching.[28]

5.3 Unified Modalities (UniteFormer)

UniteFormer (2025) addresses the disconnect between node-centric and edge-centric models. VRPs involve both node features (demand) and edge features (traffic).

- **Mixed Encoder**: UniteFormer integrates GNNs (for edges) and Attention (for nodes) into a single block. It captures "cross-modal interactions"—how an edge attribute might suppress or enhance a node's saliency. This unification leads to SOTA performance on datasets where both node and edge information are critical.[29]

6. Emerging Learning Paradigms

Beyond architecture, the fundamental *way* models learn to construct solutions is being reimagined.

6.1 Learning to Insert (L2C-Insert)

The dominant paradigm has been "Appending"—adding the next node to the end of the tour. *L2C-Insert* (2025) argues this is rigid and prone to error propagation.[31]

- **Paradigm**: This model learns an **Insertion Heuristic**. At each step, it predicts not just *which* node to add, but *where* in the partial tour to insert it.
- **Advantage**: This mimics the flexibility of human planners and classical heuristics (like Cheapest Insertion). It allows the model to correct or refine the global shape of the tour dynamically, yielding significantly better solutions than appending-based baselines like POMO.[31]

6.2 Preference Optimization (PO & POCO)

Standard RL uses scalar rewards (e.g., negative tour length). *Preference Optimization* (2025) and *POCO* (2025) introduce a paradigm inspired by RLHF (Reinforcement Learning from Human Feedback).[32]

- **Problem**: In late-stage training, reward differences between actions are microscopic, leading to vanishing gradients.
- **Solution**: These methods train the policy to maximize the probability of generating the "preferred" solution in a pair (i.e., $A > B$). This qualitative ranking is invariant to reward scale. POCO further introduces a loss function based on objective value differences, stabilizing training and improving sample efficiency significantly compared to REINFORCE.[32]

6.3 Segmentation (L2Seg)

For massive problems, *Learning to Segment* (L2Seg, 2025) introduces a "First-Segment-Then-Aggregate" strategy.[35]

- **Concept**: Local search heuristics often waste time re-evaluating "stable" parts of a route that are already optimal. L2Seg trains a network to identify these stable sub-segments and collapse them into single "hyper-nodes."
- **Result**: This reduces the effective problem size for the solver, accelerating iterative heuristics (like LKH-3) by 2x-7x.[35]

7. Generative and Diffusion Models

Moving beyond constructive solvers, generative models are gaining traction.

7.1 Hybrid-Balance GFlowNet (HBG)

HBG (2025) applies Generative Flow Networks (GFlowNets) to VRP.[36]

- **Innovation**: Standard GFlowNets use Trajectory Balance (TB) for global consistency but miss local structure. HBG integrates **Detailed Balance (DB)**—ensuring consistency between individual state transitions—with TB. This hybrid loss captures both global optimality and local feasibility.
- **Depot-Guided Inference**: Recognizing the asymmetry of VRP (depot vs. customers), HBG applies stochastic sampling primarily at the depot (where decisions are pivotal) and greedy logic at customers, balancing exploration and exploitation.[37]

7.2 Diffusion Solvers (DIFUSCO, IDEQ)

Papers like *DIFUSCO* (2023) and *IDEQ* (2025) treat the solution matrix as an image to be "denoised".[38]

- **IDEQ**: This "Improved Diffusion Model" refines the generation process, likely by constraining the diffusion to the valid manifold of permutations or using discrete diffusion steps to handle the combinatorial nature of TSP/VRP directly, rather than continuous approximations.[39]

8. Multi-Agent and Dynamic Scenarios

Real-world routing is often dynamic and involves multiple agents.

8.1 Parallel Autoregressive Policies (PARCO)

PARCO (2025) addresses the bottleneck of decoding actions for multiple vehicles sequentially.[40]

- **Parallelism**: PARCO employs a "Multiple Pointer Mechanism" to decode actions for all agents simultaneously.
- **Conflict Resolution**: A "Priority-based Conflict Handling" scheme resolves collisions (e.g., two agents claiming the same task). This parallel decoding is crucial for scaling Multi-Agent VRP (MAVRP) to real-time constraints.[40]

8.2 Time-Dependent and Asymmetric TSP

New 2025 papers tackle variants like *Time Dependent TSP* (where costs change over time) and *Asymmetric TSP* (cost $A \to B \neq B \to A$).

- **Trace-Guided Cost Augmentation**: For Asymmetric TSP, a 2025 study uses "Trace-Guided Cost Augmentation" to transform the asymmetric cost matrix into a form more amenable to symmetric solvers, effectively bridging the gap between specific and

general solvers.[41]

9. Theoretical Insights and Probing

Finally, a crucial layer of 2025 research is introspection. *Probing Neural Combinatorial Optimization Models* (2025) asks: *What do these models actually learn?*.[27]

- **Findings**: Diagnostic probes reveal that many NCO models rely on superficial features (e.g., proximity of neighbors) rather than deep structural reasoning (e.g., bottleneck identification). This explains their fragility on OOD data. This insight is driving the move towards architectures that enforce structural reasoning, such as the hierarchical Scale-Net and logic-driven AHD.

10. Conclusion and Future Outlook

The trajectory of Neural Combinatorial Optimization from 2018 to 2026 illustrates a maturation from experimental curiosity to industrial-grade foundation.

1. **From Solving to Designing**: The most significant trend is the shift towards using AI (LLMs) to *design* the algorithms (AHD) rather than just *executing* them. This leverages the interpretability and efficiency of code while harnessing the creativity of AI.
2. **From Fragmentation to Unification**: The era of problem-specific models is ending. Unified solvers like URS and SHIELD, utilizing unified data representations and modular experts, are paving the way for "Foundation Models of Optimization."
3. **From Rigid to Flexible**: Architectures are breaking free from the "appending" dogma, embracing insertion, segmentation, and preference-based learning to mimic the fluidity of human problem-solving.
4. **From Synthetic to Real**: Through techniques like EvoReal and Test-Time Projection (TTPL), the gap between academic benchmarks and real-world chaos is finally being bridged.

Future Work will likely focus on:

- **Interactive Design**: Human-AI collaboration where experts guide the LLM's heuristic search.
- **Symbolic-Neural Hybrids**: Combining the speed of neural intuition with the guarantees of symbolic logic (e.g., constraints encoded as masks).
- **Lifelong Learning**: Solvers that evolve continuously in deployment, using mechanisms like *Self-Improvement* (SIL) and *Curriculum Learning* to adapt to shifting logistics landscapes without catastrophic forgetting.

Table 1: Comparative Analysis of Major NCO Paradigms (2025-2026)

Paradigm	Representative Papers	Core Innovation	Strengths	Weaknesses
Unified Solvers	URS [21], RouteFinder [23]	Unified Data Representation, Global Embeddings	Zero-shot generalization to 100+ variants	Input complexity; potential "jack of all trades" dilution
Automated Heuristic Design	EoH-S [13], VRPAGENT [17]	LLM as Code Generator, Evolutionary Search	Interpretable code, novel operator discovery	Dependence on LLM reasoning quality; verification overhead
Hierarchical Scaling	Scale-Net [27], L2Seg [35]	Multi-scale processing, Segmentation	Scales to 1000+ nodes, accelerates local search	Architectural complexity; requires careful tuning of hierarchy
Insertion Learning	L2C-Insert [31]	Insertion-based construction	Superior solution quality, flexible correction	More complex training target than appending
Preference Optimization	PO [32], POCO [34]	Learning from relative preferences	Robust gradients, sample efficiency	Requires generating comparison pairs
Generative/Diffusion	HBG [36], DIFUSCO [38]	Hybrid Balance, Denoising	Diversity of solutions, global+local consistency	Slower inference speed than constructive models

Table 2: Taxonomy of 2025-2026 Papers by Theme

Theme	Papers
LLM & AHD	EoH-S [13], VRPAGENT [17], MCTS-AHD [16], ReEvo [12], RedAHD [41], PoH [41]
Unified/Generalist	URS [21], RouteFinder [23], SHIELD [25], Omni-VRP [43], UDC [41], GOAL [44]
Architectures	Scale-Net [27], Recurrent State Encoders [28], UniteFormer [29], EFormer [41], Pointerformer [45]
Learning Paradigms	L2C-Insert [31], PO [32], POCO [34], L2Seg [35], TTPL [46], MTL-KD [41]
Search/Hybrid	NeuralGLS [47], GFACS [41], DeepACO [48], PARCO [40], TuneNSearch [41]
Generative	HBG [36], DIFUSCO [38], IDEQ [39], AGFN [41]

Works cited

1. Machine Learning for Combinatorial Optimization: a Methodological Tour d'Horizon - arXiv, accessed December 20, 2025, https://arxiv.org/abs/1811.06128
2. Machine Learning for Combinatorial Optimization - Emergent Mind, accessed December 20, 2025, https://www.emergentmind.com/papers/1811.06128
3. Machine Learning for Combinatorial Optimization: a Methodological Tour d'Horizon, accessed December 20, 2025, https://www.researchgate.net/publication/328997287_Machine_Learning_for_Combinatorial_Optimization_a_Methodological_Tour_d'Horizon
4. Machine Learning to Solve Vehicle Routing Problems: A Survey - Semantic Scholar, accessed December 20, 2025, https://www.semanticscholar.org/paper/Machine-Learning-to-Solve-Vehicle-Routing-Problems%3A-Bogyrbayeva-Meraliyev/7407839cdaef3ded180ea0f3c114fb90452e1a9f
5. Metaheuristic and Reinforcement Learning Techniques for Solving the Vehicle Routing Problem: A Literature Review, accessed December 20, 2025, https://jtte.chd.edu.cn/cn/article/pdf/preview/2d453e1a-80c9-4d0f-b3cf-9ac04842281c.pdf
6. Machine Learning to Solve Vehicle Routing Problems: A Survey - ResearchGate, accessed December 20, 2025, https://www.researchgate.net/publication/377095847_Machine_Learning_to_Solv

e_Vehicle_Routing_Problems_A_Survey
7. Bengio, Y., Lodi, A. and Prouvost, A. (2021) Machine Learning for Combinatorial Optimization A Methodological Tour Dhorizon. European Journal of Operational Research, 290, 405-421. - References - Scirp.org, accessed December 20, 2025, https://www.scirp.org/reference/referencespapers?referenceid=3907596
8. (PDF) Neural combinatorial optimization with reinforcement learning in industrial engineering: a survey - ResearchGate, accessed December 20, 2025, https://www.researchgate.net/publication/389008680_Neural_combinatorial_optimization_with_reinforcement_learning_in_industrial_engineering_a_survey
9. Neural Combinatorial Optimization Algorithms for Solving Vehicle Routing Problems: A Comprehensive Survey with Perspectives | AI Research Paper Details - AIModels.fyi, accessed December 20, 2025, https://www.aimodels.fyi/papers/arxiv/neural-combinatorial-optimization-algorithms-solving-vehicle-routing
10. Chunguo Wu - CatalyzeX, accessed December 20, 2025, https://www.catalyzex.com/author/Chunguo%20Wu
11. ReEvo: Large Language Models as Hyper-Heuristics with Reflective Evolution - arXiv, accessed December 20, 2025, https://arxiv.org/html/2402.01145v1
12. ReEvo: Large Language Models as Hyper-Heuristics with Reflective Evolution - NIPS papers, accessed December 20, 2025, https://papers.nips.cc/paper_files/paper/2024/file/4ced59d480e07d290b6f29fc8798f195-Paper-Conference.pdf
13. EoH-S: Evolution of Heuristic Set using LLMs for Automated Heuristic Design, accessed December 20, 2025, https://scholars.cityu.edu.hk/en/publications/eoh-s-evolution-of-heuristic-set-using-llms-for-automated-heurist/
14. Monte Carlo Tree Search for Comprehensive Exploration in LLM-Based Automatic Heuristic Design - OpenReview, accessed December 20, 2025, https://openreview.net/pdf/cbb3f1eaa457593c174bf432c6f6af80f3a4f9c9.pdf
15. Monte Carlo Tree Search for Comprehensive Exploration in LLM-Based Automatic Heuristic Design - ChatPaper, accessed December 20, 2025, https://chatpaper.com/chatpaper/paper/100577
16. Bryan Hooi - DBLP, accessed December 20, 2025, https://dblp.org/pid/169/9975
17. VRPAgent: LLM-Driven Discovery of Heuristic Operators for Vehicle Routing Problems | Request PDF - ResearchGate, accessed December 20, 2025, https://www.researchgate.net/publication/396329883_VRPAgent_LLM-Driven_Discovery_of_Heuristic_Operators_for_Vehicle_Routing_Problems
18. VRPAgent: LLM-Driven Discovery of Heuristic Operators for Vehicle Routing Problems, accessed December 20, 2025, https://chatpaper.com/paper/197292
19. Bridging Synthetic and Real Routing Problems via LLM-Guided Instance Generation and Progressive Adaptation - arXiv, accessed December 20, 2025, https://arxiv.org/pdf/2511.10233
20. Large Language Models as End-to-end Combinatorial Optimization Solvers - NeurIPS 2025, accessed December 20, 2025, https://neurips.cc/virtual/2025/poster/115835

21. URS: A Unified Neural Routing Solver for Cross-Problem Zero-Shot Generalization, accessed December 20, 2025, https://www.researchgate.net/publication/395970745_URS_A_Unified_Neural_Routing_Solver_for_Cross-Problem_Zero-Shot_Generalization
22. URS: A Unified Neural Routing Solver for Cross-Problem Zero-Shot Generalization - arXiv, accessed December 20, 2025, https://arxiv.org/html/2509.23413v1
23. RouteFinder Towards Foundation Models for Vehicle Routing Problems - OpenReview, accessed December 20, 2025, https://openreview.net/pdf?id=hCiaiZ6e4G
24. RouteFinder: Towards Foundation Models for Vehicle Routing Problems - GitHub Pages, accessed December 20, 2025, https://ai4co.github.io/routefinder/
25. Multi-Task Vehicle Routing Solver via Mixture of Specialized Experts under State-Decomposable MDP - arXiv, accessed December 20, 2025, https://arxiv.org/pdf/2510.21453
26. Multi-Task Vehicle Routing Solver via Mixture of Specialized Experts under State-Decomposable MDP - arXiv, accessed December 20, 2025, https://arxiv.org/html/2510.21453v1
27. SMU – Publications - Zhiguang Cao, accessed December 20, 2025, https://zhiguangcaosg.github.io/publications/
28. Recurrent State Encoders for Efficient Neural Combinatorial Optimization - arXiv, accessed December 20, 2025, https://arxiv.org/html/2509.05084
29. NeurIPS Poster UniteFormer: Unifying Node and Edge Modalities in Transformers for Vehicle Routing Problems, accessed December 20, 2025, https://neurips.cc/virtual/2025/poster/119374
30. UniteFormer: Unifying Node and Edge Modalities in Transformers for Vehicle Routing Problems - OpenReview, accessed December 20, 2025, https://openreview.net/pdf/708cb8fb259d28e0829eac27646ee59d7696cafd.pdf
31. Learning to Insert for Constructive Neural Vehicle Routing Solver - arXiv, accessed December 20, 2025, https://arxiv.org/html/2505.13904v3
32. Preference Optimization for Combinatorial Optimization Problems - OpenReview, accessed December 20, 2025, https://openreview.net/forum?id=8QkpCRio53
33. [2503.07580] Neural Combinatorial Optimization via Preference Optimization - arXiv, accessed December 20, 2025, https://arxiv.org/abs/2503.07580
34. Neural Combinatorial Optimization via Preference Optimization - arXiv, accessed December 20, 2025, https://arxiv.org/pdf/2503.07580?
35. LEARNING TO SEGMENT FOR VEHICLE ROUTING PROBLEMS - OpenReview, accessed December 20, 2025, https://openreview.net/pdf/2017878084c32eef1f37fe7a64b907196f661d10.pdf
36. Hybrid-Balance GFlowNet for Solving Vehicle Routing Problems - arXiv, accessed December 20, 2025, https://arxiv.org/html/2510.04792v1
37. Hybrid-Balance GFlowNet for Solving Vehicle Routing Problems - ResearchGate, accessed December 20, 2025, https://www.researchgate.net/publication/396250383_Hybrid-Balance_GFlowNet_for_Solving_Vehicle_Routing_Problems
38. [2302.08224] DIFUSCO: Graph-based Diffusion Solvers for Combinatorial

Optimization - arXiv, accessed December 20, 2025, https://arxiv.org/abs/2302.08224
39. Deep Learning Based Symbolic Regression with D3PM Discrete Token Diffusion, accessed December 20, 2025, https://www.researchgate.net/publication/396373150_Symbolic-Diffusion_Deep_Learning_Based_Symbolic_Regression_with_D3PM_Discrete_Token_Diffusion
40. PARCO: Learning Parallel Autoregressive Policies for Efficient Multi-Agent Combinatorial Optimization - arXiv, accessed December 20, 2025, https://arxiv.org/html/2409.03811v1
41. dongjinkun/COPZoo: Neural Combinatorial Optimization - GitHub, accessed December 20, 2025, https://github.com/dongjinkun/COPZoo
42. San Diego Spotlights - NeurIPS 2025, accessed December 20, 2025, https://neurips.cc/virtual/2025/loc/san-diego/events/spotlights-2025
43. Towards Omni-generalizable Neural Methods for Vehicle Routing Problems - Proceedings of Machine Learning Research, accessed December 20, 2025, https://proceedings.mlr.press/v202/zhou23o/zhou23o.pdf
44. [PDF] Machine Learning for Combinatorial Optimization: a Methodological Tour d'Horizon, accessed December 20, 2025, https://www.semanticscholar.org/paper/Machine-Learning-for-Combinatorial-Optimization%3A-a-Bengio-Lodi/3f13a5148f7caa51ea946193d261d4f8ed32d81a
45. (PDF) Pointerformer: Deep Reinforced Multi-Pointer Transformer for the Traveling Salesman Problem - ResearchGate, accessed December 20, 2025, https://www.researchgate.net/publication/371934216_Pointerformer_Deep_Reinforced_Multi-Pointer_Transformer_for_the_Traveling_Salesman_Problem
46. Improving Generalization of Neural Combinatorial Optimization for Vehicle Routing Problems via Test-Time Projection Learning - arXiv, accessed December 20, 2025, https://arxiv.org/html/2506.02392v2
47. NeuralGLS: learning to guide local search with graph convolutional network for the traveling salesman problem | Request PDF - ResearchGate, accessed December 20, 2025, https://www.researchgate.net/publication/374586025_NeuralGLS_learning_to_guide_local_search_with_graph_convolutional_network_for_the_traveling_salesman_problem
48. DeepACO: Neural-enhanced Ant Systems for Combinatorial Optimization | OpenReview, accessed December 20, 2025, https://openreview.net/forum?id=cd5D1DD923¬eId=fIymzLkwlJ