Stabilizing Hierarchical Reinforcement Learning for the Periodic Capacitated Vehicle Routing Problem: A Comprehensive Framework Against Policy Collapse

1. Introduction: The Complexity Frontier in Neural Combinatorial Optimization

The field of Neural Combinatorial Optimization (NCO) has reached a critical inflection point. While Deep Reinforcement Learning (DRL) models have demonstrated superhuman capabilities in static routing tasks such as the Traveling Salesman Problem (TSP) and the Capacitated Vehicle Routing Problem (CVRP), the transition to dynamic, periodic, and long-horizon problems remains fraught with instability. The Periodic Capacitated Vehicle Routing Problem (PCVRP) represents the pinnacle of this challenge, demanding not only spatial optimization but also temporal planning, inventory management, and consistency across a multi-day horizon.[1]

The application of Hierarchical Reinforcement Learning (HRL) to the PCVRP—specifically through a "Manager-Worker" architecture—offers a theoretically elegant solution. By decomposing the intractable long-horizon problem into a high-level master problem (customer-to-day assignment) and a low-level subproblem (daily route construction), HRL aligns with the classical "divide-and-conquer" heuristic that has dominated Operations Research (OR) for decades.[1] However, practical implementations of Manager-Worker models in this domain are plagued by a pervasive failure mode known as **policy collapse**.

Policy collapse in the PCVRP context is not merely a stagnation of learning; it is a structural breakdown of the hierarchical interface. It occurs when the Manager, tasked with decomposing the problem, consistently generates subproblems that are infeasible for the Worker to solve given the strict constraints of vehicle capacity and time windows. The Worker, facing impossible tasks, returns negative or noisy reward signals. Consequently, the Manager's policy gradients vanish or become highly erratic, leading the agent to converge toward trivial, highly suboptimal, or invalid policies to avoid the penalties of exploration.[3]

This report provides an exhaustive analysis of the mechanisms driving policy collapse and proposes a robust, multi-faceted framework for stabilization. We synthesize state-of-the-art research on **Proactive Infeasibility Prevention (PIP)**, **Lagrangian Relaxation**, **Hybrid**

Neuro-Heuristic Workers, and **Curriculum Learning** to construct a blueprint for a stable, high-performance PCVRP solver. We argue that stability cannot be achieved through a single algorithmic "trick" but requires a holistic re-engineering of the constraint handling, reward shaping, and hierarchical communication protocols.

1.1 The Mathematical Anatomy of the PCVRP and Policy Collapse

To understand the severity of the stability crisis, one must rigorously define the PCVRP constraints that precipitate it. The problem is defined on a graph $G = (V, E)$ over a planning horizon $T = \{1, \dots, \tau\}$. Each customer $i \in V$ has a demand d_i, a service frequency f_i, and a set of allowable visit patterns P_i (e.g., specific days they can be visited).

The objective is to minimize the total routing cost over the horizon:

$$\min \sum_{t \in T} \sum_{k \in K} \text{Cost}(R_{tk})$$

Subject to three critical classes of constraints:

1. **Pattern Constraints:** Every customer must be assigned a valid pattern $p \in P_i$.[5]
2. **Capacity Constraints:** The sum of demands for all customers assigned to vehicle k on day t must not exceed Q.[6]
3. **Fleet Size Constraints:** The number of routes on day t cannot exceed the available fleet $|K|$.

In a Manager-Worker HRL framework, the Manager operates at the pattern selection level. At step i, the Manager selects a pattern p_i for customer i. Once all customers are assigned, the Worker solves τ independent CVRP instances, one for each day t.[1]

The Mechanism of Collapse:
The instability arises from the decoupling of feasibility and decision. The Manager selects patterns based on a high-level policy π_M. However, the feasibility of the resulting daily subproblems is only revealed after the Worker attempts to route them. If the Manager assigns a set of customers S_t to Day 1 such that $\sum_{j \in S_t} d_j \le |K| \cdot Q$ (theoretical capacity feasibility) but their spatial distribution precludes valid routing under time windows, the Worker fails.4
Standard RL algorithms (PPO, REINFORCE) rely on the assumption that the reward function is a continuous, shaped landscape. In PCVRP, the landscape is fractured by "cliffs" of infeasibility. When a Manager inadvertently pushes the state over such a cliff (creating an infeasible subproblem), the reward drops precipitously (e.g., to a large penalty value). If the Manager encounters these cliffs frequently during early exploration, the variance of the gradient estimator explodes. The policy update step then destroys the learned representations, resulting in a model that performs worse than random initialization—a

phenomenon termed "catastrophic forgetting" or policy collapse.[10]

1.2 Strategies for Stabilization

The remainder of this report details four pillars of stabilization that address these failure modes:

1. **Proactive Infeasibility Prevention (PIP):** Moving constraint checking from the end of the process to the beginning using learnable masks and auxiliary decoders.[12]
2. **Advanced Lagrangian Relaxation:** Replacing static penalties with dynamic, learnable multipliers that adaptively shape the reward landscape, managed via Dual Gradient Descent.[9]
3. **Hybrid Worker Architectures:** stabilizing the lower level by replacing pure RL workers with **NeuroLKH**, a neural-guided heuristic that guarantees competence and stationarity.[14]
4. **Curriculum Learning & Reward Shaping:** Smoothing the optimization landscape through **Relaxed Exploration** and **Potential-Based Reward Shaping (PBRS)** to guide the agent away from cliffs.[16]

---

2. Advanced Constraint Handling: Proactive Infeasibility Prevention (PIP)

The most direct antidote to policy collapse is to prevent the Manager from ever selecting an infeasible action. In unconstrained environments, RL agents learn to avoid bad actions through trial and error. In the highly constrained PCVRP, the "bad" actions (infeasible assignments) vastly outnumber the "good" ones, making trial-and-error inefficient and unstable. **Proactive Infeasibility Prevention (PIP)** creates a "safety railing" around the policy, masking out invalid actions before they are sampled.[12]

2.1 Theoretical Foundations of PIP in HRL

Standard "validity masking" in RL involves deterministically masking actions that violate simple rules (e.g., "do not visit a visited node"). PIP extends this to complex, NP-Hard constraints by introducing a probabilistic or learned mask.
Formally, let the Manager's policy be $\pi_\theta(a_t | s_t)$. The PIP-modified policy is:

$$\pi_{PIP}(a_t | s_t) = \text{Softmax}\left(\frac{\mathbf{h}_t^\top \mathbf{W} \mathbf{e}_a}{\sqrt{d}} + M(s_t, a) \right)$$

Where $M(s_t, a)$ is the mask value. In reactive systems, M is heuristic. In PIP, M is derived from a predictive model that estimates the conditional probability of future infeasibility.19

This distinction is crucial for PCVRP. A reactive mask might prevent assigning a customer to a full vehicle. A proactive PIP mask prevents assigning a customer to a vehicle that has *space* but will later fail time window constraints due to the detour. By filtering these "trap" actions, PIP ensures the Worker always receives feasible subproblems, stabilizing the reward signal.[20]

2.2 The PIP-D Auxiliary Decoder Architecture

Calculating the ground-truth feasibility of an action in PCVRP is computationally expensive (equivalent to solving the VRP). Therefore, we cannot use exact solvers to generate masks during training. Instead, we employ the **PIP-D (Auxiliary Decoder)** framework, which trains a lightweight neural network to *predict* the mask.[13]

2.2.1 Network Architecture

The PIP-D module operates in parallel with the Manager's primary policy network.

- **Shared Encoder:** It utilizes the same node embeddings $\mathbf{h}_i$ generated by the Manager's Transformer encoder, ensuring it has access to the full context of demands, locations, and inventory levels.[18]
- **Dual Heads:** The decoder has two output heads:
  1. **Policy Head:** Outputs the log-probabilities for action selection (the standard Actor).
  2. **Auxiliary Head (PIP Decoder):** Outputs a binary feasibility score $\hat{y}_{ia} \in $ for each candidate action a.[18]
- Mechanism: The Auxiliary Head typically uses a Multi-Layer Perceptron (MLP) or a single attention block. It takes the current context embedding $\mathbf{h}_c$ (representing the partial solution/current day state) and the candidate embedding $\mathbf{h}_a$.

  $$\hat{y}_{ia} = \sigma(\text{MLP}([\mathbf{h}_c; \mathbf{h}_a]))$$

  Where σ is the sigmoid function. A value close to 1 indicates the action is predicted to be feasible; 0 indicates infeasibility.22

2.2.2 Training the PIP-D

The training of the PIP-D is a supervised learning task embedded within the RL loop.

1. **Data Collection:** As the Manager interacts with the environment, it generates trajectories τ. Some trajectories end in success (feasible routing), others in failure (infeasibility).
2. **Label Generation:** For failed trajectories, we retrospectively identify the first action that rendered the remaining problem unsolvable (using a fast heuristic checker or the Worker's feedback). This action constitutes a negative sample ($y=0$). Successful actions are positive samples ($y=1$).
3. Loss Function: The PIP-D is optimized via Binary Cross-Entropy (BCE):

$$\mathcal{L}_{PIP} = - \frac{1}{N} \sum_{i,a} [y_{ia} \log \hat{y}_{ia} + (1-y_{ia}) \log(1 - \hat{y}_{ia})]$$

Crucially, gradients from $\mathcal{L}_{PIP}$ are often stopped before reaching the shared encoder to prevent the auxiliary task from distorting the representations needed for the optimality policy.22

2.3 Adaptive Strategies for Preventative Masking

A critical insight from recent research is that blindly trusting the learned mask early in training can lead to sub-optimality. If the PIP-D is inaccurate, it might mask out the optimal action, forcing the Manager into a local optimum. To counter this, we implement **Adaptive Masking Strategies**.[13]

2.3.1 Bias Annealing

We introduce a scaling factor β that controls the influence of the predicted mask on the policy logits.

$$\text{Logits}_{final} = \text{Logits}_{policy} + \beta \cdot \log(\hat{y}_{ia})$$

Initially, $\beta \approx 0$, allowing the agent to explore even "risky" actions. As the PIP-D accuracy improves (monitored via validation accuracy on the binary classification task), β is linearly increased, gradually enforcing the predicted constraints. This prevents the "blind leading the blind" scenario in early training.22

2.3.2 Thresholding

Instead of soft masking, a hard threshold δ can be used.

$$M(s, a) = \begin{cases} 0 & \text{if } \hat{y}_{ia} > \delta \\ -\infty & \text{if } \hat{y}_{ia} \le \delta \end{cases}$$

The threshold δ is also adaptive. In the "Relaxed Exploration" phase (see Section 6), δ is low (0.1), permitting almost any action. In the "Deployment" phase, δ is high (0.9), ensuring strict feasibility.24

2.4 Impact on PCVRP Stability

Implementing PIP-D transforms the Manager's learning landscape. Instead of stepping off a cliff and receiving a reward of -1000, the Manager sees a "fence" (the mask) and is forced to choose a safe path. This has three profound effects:

1. **Gradient Variance Reduction:** Since most episodes result in valid routes (even if suboptimal), the reward variance decreases, stabilizing the gradient estimator.

2. **Sample Efficiency:** The agent wastes fewer episodes exploring inherently invalid parts of the search space.
3. **Decoupling:** The Manager can focus on optimizing *cost* (distance), trusting the PIP-D to handle *feasibility*.[18]

---

3. Advanced Lagrangian Relaxation and Dual Gradient Descent

While PIP handles structural feasibility (binary yes/no), the PCVRP also involves "soft" constraints or objectives that compete with cost, such as minimizing route duration or balancing workload across days. For these, **Lagrangian Relaxation** is the superior tool. It integrates constraints into the reward function dynamically, avoiding the pitfalls of fixed penalty coefficients.[9]

3.1 The Constrained MDP (CMDP) Formulation

We formulate the PCVRP as a Constrained Markov Decision Process (CMDP). The goal is to find a policy π_θ that maximizes the expected return $J_R(\pi)$ while keeping the expected cost of constraint violations $J_{C_k}(\pi)$ below a limit L_k.

$$\max_\pi J_R(\pi) \quad \text{s.t.} \quad J_{C_k}(\pi) \le L_k, \quad k=1,\dots,K$$

In PCVRP, C_k could represent the "Capacity Violation on Day k" or "Time Window Violation Magnitude".26
Standard approaches often use a fixed penalty λ: $R' = R - \lambda C$. This is brittle. If λ is too small, the agent ignores the constraint. If too large, the agent becomes overly conservative or fails to learn (sparse reward problem).

3.2 The Lagrangian Primal-Dual Method

To resolve this, we treat the Lagrangian multipliers λ_k as learnable parameters in a min-max game against the policy θ.

$$\min_{\lambda \ge 0} \max_\theta \mathcal{L}(\theta, \lambda) = J_R(\pi_\theta) - \sum_k \lambda_k (J_{C_k}(\pi_\theta) - L_k)$$

3.2.1 Primal Update (Policy Optimization)

The policy θ is updated to maximize the Lagrangian objective. We use PPO (Proximal Policy Optimization) or REINFORCE. The effective reward for the agent at episode i becomes:

$$r_i' = r_i - \sum_k \lambda_k \cdot c_{k,i}$$

where $c_{k,i}$ is the observed violation. This teaches the agent to trade off reward against the current cost of violating constraints.26

3.2.2 Dual Update (Multiplier Optimization)

Simultaneously, we update λ_k to minimize the objective (i.e., maximize the penalty for violations). This is done via Dual Gradient Ascent:

$$\lambda_k^{(t+1)} \leftarrow \max\left(0, \lambda_k^{(t)} + \eta_{dual} \cdot (J_{C_k}(\pi_\theta) - L_k)\right)$$

Here, η_{dual} is the learning rate for the dual variables.

- **Mechanism:** If the agent violates constraint k consistently ($J_{C_k} > L_k$), the term $(J_{C_k} - L_k)$ is positive. λ_k increases. The penalty grows.
- **Feedback Loop:** As λ_k grows, the agent is incentivized to reduce J_{C_k}. Once $J_{C_k} \le L_k$, the gradient becomes negative, and λ_k decreases. This creates a self-regulating "thermostat" that finds the minimum penalty required to enforce feasibility.[28]

3.3 PID-Controlled Lagrangian Updating

Simple gradient ascent for λ can lead to oscillations (the "overshoot" problem). The multiplier grows too large, the agent collapses to an overly safe policy, the multiplier drops to zero, and the agent returns to violation. To stabilize this in PCVRP, we employ a **PID (Proportional-Integral-Derivative) Controller** for the dual update.[30]

The update rule becomes:

$$\Delta \lambda_k = K_P e_t + K_I \sum e_t + K_D (e_t - e_{t-1})$$

Where error $e_t = J_{C_k}(\pi) - L_k$.

- **Proportional (K_P):** Reacts to current violation.
- **Integral (K_I):** Reacts to accumulated past violations (eliminating steady-state error).
- **Derivative (K_D):** Dampens the reaction to prevent overshoot.

Implementing PID-Lagrangian control is a key recommendation for stabilizing the Manager-Worker HRL model, as it smooths the shifting reward landscape, preventing the sudden shocks that cause policy collapse.[29]

3.4 Table: Comparison of Constraint Handling Strategies

Strategy	Mechanism	Pros	Cons	Best Application
Reactive Masking	Heuristic rules (e.g., visited set)	Simple, zero training cost	Cannot handle NP-Hard constraints; myopic	Simple VRPs
Fixed Penalty	$R' = R - \beta C$	Easy to implement	Highly sensitive to β; unstable	Minor soft constraints
PIP / PIP-D	Learned predictive mask	Proactive; prevents failed episodes; stabilizes gradient	Requires auxiliary training; complexity	**Hard Constraints (Capacity, Windows)**
Lagrangian Relaxation	Dynamic λ (Primal-Dual)	Adaptive; guarantees optimality in limit	Can oscillate; requires tuning η	**Soft Constraints (Balance, Duration)**

Table 1: Comparative analysis of constraint handling mechanisms. Data synthesized from.[9]

---

4. The Hybrid Worker: Grounding the Hierarchy with NeuroLKH

In classical HRL, both the Manager and Worker are RL agents trained simultaneously. This introduces **non-stationarity**: the Manager is learning to assign subgoals to a Worker that is itself changing. In PCVRP, if the Worker explores poorly and fails to route a valid cluster, the Manager "learns" that the cluster is bad, even if it was actually optimal. This false feedback loop is a primary driver of instability.[3]

To stabilize the PCVRP model, we propose a **Hybrid HRL** architecture where the Worker is not a pure RL agent but a **Neural-Guided Heuristic**, specifically **NeuroLKH**.

4.1 Why Heuristics Stabilize HRL

Replacing the RL Worker with a robust heuristic like LKH-3 (Lin-Kernighan-Helsgaun) eliminates the non-stationarity of the lower level.

1. **Stationarity:** The heuristic's behavior is deterministic or statistically stable. A specific subproblem S always yields cost C.
2. **Competence:** LKH-3 is a state-of-the-art solver. It guarantees that if a feasible route exists, it will likely be found.
3. **Signal Fidelity:** The feedback to the Manager becomes a true reflection of the *quality of the decomposition*, not the *incompetence of the router*.[14]

4.2 NeuroLKH: Bridging the Gap

While LKH-3 is stable, it can be slow. **NeuroLKH** enhances LKH-3 by using a Deep Neural Network (Sparse Graph Network - SGN) to predict which edges are likely to be part of the optimal solution. It then restricts LKH-3's search to this sparse subgraph, speeding up execution by orders of magnitude while retaining solution quality.[15]

4.2.1 SGN Architecture

The Sparse Graph Network (SGN) used in NeuroLKH is a Graph Convolutional Network (GCN) or Graph Attention Network (GAT).

- **Input:** The subproblem graph defined by the Manager's assignment for Day t. Features include node coordinates and demands.
- Edge Scoring: The SGN outputs a probability matrix $P_{ij} \in $ representing the likelihood of edge (i, j) being in the optimal tour.

  $$P_{ij} = \text{Sigmoid}(\mathbf{u}_i^T \mathbf{W} \mathbf{u}_j)$$
- **Sparsification:** We select the top-K neighbors for each node based on P_{ij} (e.g., $K=20$).
- **Worker Execution:** The LKH-3 algorithm is run *only* on these candidate edges.

4.2.2 Integration into Manager-Worker Loop

In the FAHH (Feasibility-Aware Hybrid Hierarchical) framework:

1. **Manager Action:** Decomposes the periodic demand into daily sets $\{S_1, \dots, S_T\}$.
2. **Worker Inference:**
   - Pass each S_t through the pre-trained SGN.
   - Generate Candidate Sets C_t.
   - Run LKH-3 on C_t to get route cost L_t.
3. **Reward:** $R = - \sum L_t$.
4. **Update:** Only the Manager (and potentially the SGN via supervised fine-tuning) is updated. The core LKH logic remains fixed.

This hybrid approach provides the **best of both worlds**: the learning capability of RL for the complex, high-level periodic assignment, and the stability/reliability of OR heuristics for the low-level routing.[18]

---

5. Curriculum Learning and Relaxed Exploration

A major reason for policy collapse is the "cold start" problem. In the initial phases of training, a random Manager policy will almost certainly violate the strict periodic and capacity constraints of PCVRP. If we apply strict penalties immediately, the agent sees a wall of negative rewards and stops exploring.
Curriculum Learning, specifically the Relaxed Exploration Constrained RL (RECRL) paradigm, addresses this by shaping the environment difficulty over time.16

5.1 The CLiC (Curriculum for Learning in Constraints) Schedule

We propose a four-phase curriculum schedule designed to gently guide the PCVRP agent from unconstrained exploration to fully constrained optimization.

Phase 1: Topology Learning (Unconstrained)

- **Constraints:** Relax capacity $Q \to \infty$ and allow any visit pattern.
- **Objective:** Minimize total distance.
- **Outcome:** The Manager learns spatial clustering—grouping nearby customers together regardless of load. This establishes the geometric primitives of the policy.

Phase 2: Soft Capacity Training (Lazy Masking)

- **Constraints:** Introduce capacity Q.
- **Mechanism:** Use **Lazy Masking**. The agent is allowed to generate infeasible assignments (e.g., overloading a vehicle).
- **Penalty:** Apply a soft Lagrangian penalty $\lambda \cdot \max(0, \text{Load} - Q)$.
- **Outcome:** The agent learns the trade-off between clustering tightness and vehicle load. It begins to "split" large clusters. The "Lazy" aspect ensures the reward signal is not cut off; the agent gets a distance reward *minus* a penalty, rather than a null reward.[35]

Phase 3: Periodic Consistency

- **Constraints:** Enforce service frequency f_i and pattern P_i.
- **Mechanism:** Introduce the PIP-D module, initially with low influence ($\beta \approx 0$).
- **Outcome:** The agent learns to manage the calendar, distributing demand across the week to smooth the daily workload.

Phase 4: Hard Constraints (Deployment Mode)

- **Constraints:** Strict Q, strict P_i, strict Time Windows.

- **Mechanism:** Full PIP masking (high β, strict threshold). High Lagrangian multipliers.
- **Outcome:** The policy is fine-tuned to ensure 100% feasibility.

5.2 Lazy Masking vs. Proactive Masking

A key distinction in RECRL is the use of **Lazy Masking** during training phases. In Proactive Masking (PIP), we prevent the agent from picking invalid actions. In Lazy Masking, we *let* them pick invalid actions but punish them.

- **Why Lazy?** It allows the agent to explore the "near-infeasible" boundaries of the state space. Sometimes, the optimal policy lies right on the edge of the feasible region. If we strictly mask exploring agents, they may never get close enough to that edge to find the optimum.[16]
- **Transition:** The curriculum must transition from Lazy (exploration) to Proactive (safety) as training converges.

5.3 Table: Curriculum Schedule for PCVRP

Phase	Epochs	Capacity Constraint	Pattern Constraint	Masking Type	Goal
1	0-100	None (∞)	Relaxed	None	Learn Spatial Clustering
2	100-500	Soft (λ_{low})	Relaxed	Lazy	Learn Load Balancing
3	500-1000	Ramp up λ	Soft (λ_{pat})	PIP-D (Low β)	Learn Temporal Smoothing
4	1000+	Hard (Strict)	Hard (Strict)	PIP-D (High β)	Feasibility & Optimality

Table 2: Proposed Curriculum Learning Schedule. Adapted from.[16]

---

6. Potential-Based Reward Shaping (PBRS)

In PCVRP, the primary reward (total distance) is often sparse, received only after the Worker completes the routes for the entire horizon. This sparsity delays learning and makes credit assignment difficult. **Potential-Based Reward Shaping (PBRS)** densifies the reward signal by providing heuristic feedback at every step of the Manager's decision process.[17]

6.1 Theoretical Guarantee: Policy Invariance

The critical property of PBRS is that it does not alter the optimal policy. As proved by Ng et al. (1999), if the shaping reward F follows the potential difference form:

$$F(s, a, s') = \gamma \Phi(s') - \Phi(s)$$

Then the optimal policy π^* maximizing the shaped reward is identical to the policy maximizing the original reward.36 This allows us to inject domain knowledge safely.

6.2 Designing Potentials for PCVRP

We propose two specific potential functions Φ to stabilize the Manager.

6.2.1 The Capacity Potential Φ_{cap}

This potential discourages the Manager from "filling up" days too early, preserving flexibility.

$$\Phi_{cap}(s) = - \sum_{t=1}^T \left(\frac{\text{CurrentLoad}_t}{Q \times |K|} \right)^2$$

As the Manager adds customers to a day, the potential $\Phi(s)$ decreases (becomes more negative). The shaping reward $F = \Phi(s') - \Phi(s)$ is negative, penalizing the action of adding load. This provides an immediate gradient pushing for load balancing, rather than waiting for the final capacity violation penalty.38

6.2.2 The Spatial Potential Φ_{dist}

This potential estimates the "remaining work" in terms of distance.

$$\Phi_{dist}(s) = - \text{MST}(\text{Unassigned Customers})$$

Using the Minimum Spanning Tree (MST) length of unassigned customers provides a lower-bound estimate of the remaining route length. When the Manager assigns a customer that is far from the current cluster (increasing the MST of the remaining set or the current route), Φ drops sharply, penalizing the spatial outlier immediately.

6.3 Implementation of PBRS

The total reward at step t becomes:

$$R_{total}(s_t, a_t, s_{t+1}) = R_{env} + \lambda_{PBRS} \cdot (\gamma \Phi(s_{t+1}) - \Phi(s_t))$$

Using PBRS is particularly effective in the "Lazy Masking" phase of the curriculum, as it provides the directional guidance needed to navigate the relaxed constraint landscape.40

---

7. Comprehensive State Representation for PCVRP

Stability is impossible if the agent is "blind" to critical state variables. In PCVRP, the state is high-dimensional and dynamic. The Manager must perceive not just the graph topology, but the temporal evolution of inventory and constraints.[10]

7.1 The Feature Vector

We define a comprehensive feature set for customer i at time t:

$$\mathbf{x}_{it} =$$

1. **Inventory Level (I_{it}):** In Inventory Routing contexts, this is dynamic. $I_{i, t+1} = I_{it} - \text{Usage}_i + \text{Delivery}_i$. The agent must know I_{it} to decide if a visit is urgent.[43]
2. **Visit History (H_{it}):** A binary mask or counter indicating how many times the customer has been visited vs. required frequency f_i.
3. **Constraint Pressure (λ_t):** Crucially, we include the current Lagrangian multiplier values λ in the state. This makes the policy **"penalty-aware."** The agent learns: "When λ is high, I must be conservative; when low, I can be aggressive." This prevents the oscillation often seen in Primal-Dual methods.[42]

7.2 Contextual Encoder Architecture

The Manager should use a **Dual-Aspect Collaborative Transformer (DACT)** or similar architecture.

- **Self-Attention:** Processes the relationship between customers (spatial clustering).
- **Cross-Attention:** Processes the relationship between customers and the "Day Nodes" (temporal assignment).
- **Dynamic Embedding:** The features I_{it} and λ_t are re-embedded at every step, allowing the attention weights to shift dynamically. For example, as a customer's inventory drops (state I_{it} decreases), the attention mechanism should implicitly increase the weight of assigning a visit.[44]

---

8. Integrated Framework Proposal & Benchmarking Strategy

Synthesizing the above research, we propose the **Feasibility-Aware Hybrid Hierarchical (FAHH)** framework for PCVRP.

Architecture Summary:

- **Manager:** Context-Aware Transformer with state inputs including inventory and Lagrangian multipliers.
- **Constraint Guard:** PIP-D Auxiliary Decoder for proactive feasibility masking.
- **Worker:** NeuroLKH (SGN + LKH-3) for stable, high-quality routing.
- **Training:** RECRL Curriculum (Relaxed $\to$ Hard) + PBRS (Capacity/Spatial Potentials) + Dual Gradient Descent (PID).

8.1 Implementation Roadmap

To successfully implement FAHH, we recommend the following phased rollout:

1. **Phase 0: Worker Pre-training.** Train the SGN on millions of random PCVRP subproblems using LKH-3 labels. Freeze the SGN.
2. **Phase 1: Unconstrained Manager.** Train the Manager using Phase 1 Curriculum (Topology). No PIP, no λ. Verify spatial clustering.
3. **Phase 2: PIP-D Warmup.** Introduce PIP-D. Train it via supervised learning on trajectories from Phase 1. Do not use it to mask yet.
4. **Phase 3: Soft Constraint Integration.** Enable Phase 2 Curriculum. Turn on Lagrangian Dual Descent. Enable PBRS.
5. **Phase 4: Full Deployment.** Enable Phase 4 Curriculum. Activate strict PIP masking. Fine-tune end-to-end.

8.2 Expected Performance Comparison

Based on the referenced literature, we project the following performance gains:

Metric	Pure RL (AM/POMO)	OR-Tools / Heuristic	FAHH (Proposed)	Source Logic
Solution Cost	High (+15% gap)	Medium (+5% gap)	**Low (Near Optimal)**	NeuroLKH optimality [31]
Infeasibility Rate	High (20-50%)	Zero (by design)	**< 0.5%**	PIP-D masking [12]

Inference Time	Fast (Seconds)	Slow (Minutes/Hours)	**Fast (Seconds)**	SGN acceleration [14]
Stability	Poor (Collapse prone)	High	**High**	Dual Descent + PBRS [28]

Table 3: Projected performance metrics based on component analysis.

8.3 Conclusion

The instability of Manager-Worker HRL models in PCVRP is a solvable structural problem. It stems from the mismatch between the difficulty of the constraints and the exploration capabilities of the agent. By integrating **Proactive Infeasibility Prevention** to filter the action space, **Lagrangian Relaxation** to shape the reward landscape, and **Hybrid Workers** to ground the learning process, we can prevent policy collapse. This framework does not merely "patch" the HRL model; it fundamentally realigns the learning dynamics with the mathematical reality of the Periodic Capacitated Vehicle Routing Problem. Future work should focus on the efficient online adaptation of the PIP-D module to handle real-time disruptions (e.g., vehicle breakdowns), moving closer to a truly autonomous logistics planning system.

Works cited

1. Hierarchical reinforcement learning in network routing optimization - DiVA portal, accessed December 22, 2025, http://www.diva-portal.org/smash/get/diva2:1955666/FULLTEXT01.pdf
2. Hierarchical Deep Reinforcement Learning for Vehicle Routing Problem - OpenReview, accessed December 22, 2025, https://openreview.net/forum?id=6G7cF9RNzP
3. RBG: Hierarchically Solving Large-Scale Routing Problems in Logistic Systems via Reinforcement Learning, accessed December 22, 2025, https://fi.ee.tsinghua.edu.cn/public/publications/6910245c-644d-11ee-84fe-0242ac120002.pdf
4. Hierarchical Reinforcement Learning for Vehicle Routing Problems with Time Windows, accessed December 22, 2025, https://www.researchgate.net/publication/352724597_Hierarchical_Reinforcement_Learning_for_Vehicle_Routing_Problems_with_Time_Windows
5. Variable Neighborhood Search for Multi-Cycle Medical Waste Recycling Vehicle Routing Problem with Time Windows - MDPI, accessed December 22, 2025, https://www.mdpi.com/1660-4601/19/19/12887
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. [2110.02629] Deep Reinforcement Learning for Solving the Heterogeneous

Capacitated Vehicle Routing Problem - arXiv, accessed December 22, 2025, https://arxiv.org/abs/2110.02629
8. A Hierarchical Reinforcement Learning Based Optimization Framework for Large-scale Dynamic Pickup and Delivery Problems, accessed December 22, 2025, https://proceedings.neurips.cc/paper/2021/file/c6a01432c8138d46ba39957a8250e027-Paper.pdf
9. [PDF] Learning to Solve Soft-Constrained Vehicle Routing Problems with Lagrangian Relaxation | Semantic Scholar, accessed December 22, 2025, https://www.semanticscholar.org/paper/adacde03b38fe2b38b98e9be28191a5762114eb8
10. Deep Reinforcement Learning in Inventory Management - Essay - UT Student Theses, accessed December 22, 2025, https://essay.utwente.nl/85432/1/Geevers_MA_BMS.pdf
11. Leveraging Transfer Learning in Deep Reinforcement Learning for Solving Combinatorial Optimization Problems Under Uncertainty - IEEE Xplore, accessed December 22, 2025, https://ieeexplore.ieee.org/iel8/6287639/10380310/10766597.pdf
12. Learning to Handle Complex Constraints for Vehicle Routing Problems - NIPS papers, accessed December 22, 2025, https://proceedings.neurips.cc/paper_files/paper/2024/file/a9d2a5fd12d34250c21b5e4fa8d906b0-Paper-Conference.pdf
13. Learning to Handle Complex Constraints for Vehicle Routing Problems - ResearchGate, accessed December 22, 2025, https://www.researchgate.net/publication/397199385_Learning_to_Handle_Complex_Constraints_for_Vehicle_Routing_Problems
14. liangxinedu/NeuroLKH - GitHub, accessed December 22, 2025, https://github.com/liangxinedu/NeuroLKH
15. NeuroLKH: Combining Deep Learning Model with Lin-Kernighan-Helsgaun Heuristic for Solving the Traveling Salesman Problem, accessed December 22, 2025, https://proceedings.neurips.cc/paper/2021/file/3d863b367aa379f71c7afc0c9cdca41d-Paper.pdf
16. Relaxed Exploration Constrained Reinforcement Learning - UT Austin Computer Science, accessed December 22, 2025, https://www.cs.utexas.edu/~pstone/Papers/bib2html-links/shahaf_shperberg_AAMAS_2024.pdf
17. Comprehensive Overview of Reward Engineering and Shaping in Advancing Reinforcement Learning Applications - arXiv, accessed December 22, 2025, https://arxiv.org/html/2408.10215v1
18. Learning to Handle Complex Constraints for Vehicle Routing Problems - arXiv, accessed December 22, 2025, https://arxiv.org/html/2410.21066v1
19. jieyibi/PIP-constraint: [NeurIPS 2024] Learning to Handle Complex Constraints for Vehicle Routing Problems - GitHub, accessed December 22, 2025, https://github.com/jieyibi/pip-constraint

20. Learning to handle complex constraints for vehicle routing problems, accessed December 22, 2025, https://ink.library.smu.edu.sg/sis_research/9814/
21. [2410.21066] Learning to Handle Complex Constraints for Vehicle Routing Problems - arXiv, accessed December 22, 2025, https://arxiv.org/abs/2410.21066
22. UCPO: A Universal Constrained Combinatorial Optimization Method via Preference Optimization - arXiv, accessed December 22, 2025, https://arxiv.org/html/2511.10148v1
23. Learning to Handle Complex Constraints for Vehicle Routing Problems, accessed December 22, 2025, https://neurips.cc/media/neurips-2024/Slides/95638_iCr6fb9.pdf
24. LMask: Learn to Solve Constrained Routing Problems with Lazy Masking - arXiv, accessed December 22, 2025, https://arxiv.org/html/2505.17938v1
25. Learning to Handle Complex Constraints for Vehicle Routing Problems - OpenReview, accessed December 22, 2025, https://openreview.net/forum?id=Ktx95ZuRjP&referrer=%5Bthe%20profile%20of%20Zhiguang%20Cao%5D(%2Fprofile%3Fid%3D~Zhiguang_Cao1)
26. arXiv:2207.09860v3 [cs.AI] 29 Jul 2022, accessed December 22, 2025, https://arxiv.org/pdf/2207.09860
27. [2207.09860] Learning to Solve Soft-Constrained Vehicle Routing Problems with Lagrangian Relaxation - arXiv, accessed December 22, 2025, https://arxiv.org/abs/2207.09860
28. Dynamic, fair, and efficient routing for cooperative autonomous vehicle fleets, accessed December 22, 2025, https://www.researchgate.net/publication/379795083_Dynamic_fair_and_efficient_routing_for_cooperative_autonomous_vehicle_fleets
29. Machine Learning — Lagrange multiplier & Dual decomposition | by Jonathan Hui - Medium, accessed December 22, 2025, https://jonathan-hui.medium.com/machine-learning-lagrange-multiplier-dual-decomposition-4afe66158c9
30. Adaptive Constrained Optimization for Neural Vehicle Routing | OpenReview, accessed December 22, 2025, https://openreview.net/forum?id=cCIQSqhuo2
31. A Deep Reinforcement Learning-Based Decision-Making Approach for Routing Problems, accessed December 22, 2025, https://www.mdpi.com/2076-3417/15/9/4951
32. NeuroLKH: Combining deep learning model with Lin-Kernighan-Helsgaun heuristic for solving the traveling salesman problem - Institutional Knowledge (InK) @ SMU, accessed December 22, 2025, https://ink.library.smu.edu.sg/context/sis_research/article/9163/viewcontent/NeurIPS_2021_neurolkh_combining_deep_learning_model_with_lin_kernighan_helsgaun_heuristic_for_solving_the_traveling_salesman_problem_Paper.pdf
33. Refining Hybrid Genetic Search for CVRP via Reinforcement Learning-Finetuned LLM, accessed December 22, 2025, https://arxiv.org/html/2510.11121v1
34. Neural Combinatorial Optimization Algorithms for Solving Vehicle Routing Problems: A Comprehensive Survey with Perspectives - arXiv, accessed December 22, 2025, https://arxiv.org/html/2406.00415v1

35. LMask: Learn to Solve Constrained Routing Problems with Lazy Masking - ChatPaper, accessed December 22, 2025, https://chatpaper.com/paper/140339
36. Reward shaping to improve the performance of deep reinforcement learning in perishable inventory management - Lirias, accessed December 22, 2025, https://lirias.kuleuven.be/retrieve/636198
37. Deep Reward Shaping from Demonstrations - BCU Open Access Repository - Birmingham City University, accessed December 22, 2025, https://www.open-access.bcu.ac.uk/4140/1/Deep%20Reward%20Shaping%20from%20Demonstrations.pdf
38. Deep Policy Dynamic Programming for Vehicle Routing Problems - OpenReview, accessed December 22, 2025, https://openreview.net/pdf/1f50c3757764d3b0f9ed70b0cd5c99c782d34ea9.pdf
39. Deep Policy Dynamic Programming for Vehicle Routing Problems - arXiv, accessed December 22, 2025, https://arxiv.org/pdf/2102.11756
40. Timing the Match: A Deep Reinforcement Learning Approach for Ride-Hailing and Ride-Pooling Services - arXiv, accessed December 22, 2025, https://arxiv.org/html/2503.13200
41. A UAV Maneuver Decision-Making Algorithm for Autonomous Airdrop Based on Deep Reinforcement Learning - PMC - NIH, accessed December 22, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC8004906/
42. 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
43. 1 Introduction - arXiv, accessed December 22, 2025, https://arxiv.org/html/2402.04463v1
44. Reinforcement Learning Approach to Stochastic Vehicle Routing Problem With Correlated Demands - IEEE Xplore, accessed December 22, 2025, https://ieeexplore.ieee.org/iel7/6287639/10005208/10223206.pdf
45. Enhanced multi-task deep reinforcement learning for the integrated inventory-routing problem under VMI mode - ResearchGate, accessed December 22, 2025, https://www.researchgate.net/publication/396080568_Enhanced_multi-task_deep_reinforcement_learning_for_the_integrated_inventory-routing_problem_under_VMI_mode