Optimizing Last-Mile Delivery with Electric Vehicles

A Multi-Objective Approach to Routing and Energy Management

Subhasis Panda[1], Lokesh Vardhan Dontamsetti[1]

Indian Institute of Science

October, 2024

Abstract

This project explores optimizing electric vehicle (EV) routing for last-mile logistics, integrating both goods delivery and energy sales to the grid. The goal is to minimize fleet trip costs while meeting customer delivery and energy sale constraints. Two approaches, Mixed Integer Linear Programming (MILP) and Reinforcement Learning (RL), are compared for their effectiveness in solving this multi-objective routing problem.

Figure 1: illustration of first mile and last mile delivery

1. Introduction

The adoption of electric vehicles (EVs) in last-mile logistics has emerged as a promising solution to meet both environmental and operational demands. EVs not only offer lower operational costs but also present a sustainable alternative to traditional delivery methods. With the increasing integration of e-commerce into everyday life, efficient last-mile delivery has become crucial, accounting for more than 50% of total logistics expenses. The combination of delivering goods and potentially selling energy back to the grid opens a novel opportunity to increase fleet efficiency while contributing to the energy demand during peak periods.

This dual-service approach introduces a new type of routing problem where EVs must meet both delivery demands and energy sale opportunities within specific time windows. The challenge is to optimize routes in such a way that the fleet meets customer requirements and contributes to the grid, all while minimizing operational costs. Managing these complexities effectively requires sophisticated routing algorithms that can handle multi-objective constraints. This project aims to explore and develop such algorithms using both Mixed Integer Linear Programming (MILP) and Reinforcement Learning (RL) to identify the most cost-effective solutions.

1.1. Problem Description

The growing use of EVs in last-mile delivery presents a unique set of challenges. The primary objective is to minimize the total trip cost for an EV fleet while satisfying both delivery schedules and potential energy demands. Each EV must complete its deliveries within a specified time frame and, where possible, sell energy to the grid during peak demand periods. The problem is made more complex by the need to manage battery charge levels and avoid unnecessary stops for recharging.

Additionally, the dynamic nature of customer demand and energy sale opportunities requires a flexible routing solution that can optimize fleet operations in real time. Traditional vehicle routing problems (VRPs) are already computationally challenging, and the addition of energy delivery constraints further complicates the problem. This project seeks to model and compare two optimization approaches—MILP and RL—in order to find the most effective way to minimize trip costs while ensuring all deliveries and energy transactions are completed efficiently.

2. Mathematical Formulation

The routing problem for electric vehicles (EVs) in last-mile delivery is a multi-objective optimization problem. The fleet operator needs to minimize both the total distance traveled and the energy consumed by the EVs, while potentially selling excess energy back to the grid during peak demand periods. The objective function can be expressed as follows:

$$\min \sum_k \sum_i \sum_j \left(\alpha d_{ij} x_{ij}^k + \beta e_{ij} x_{ij}^k - \gamma y_{ij}^k \right) \quad (1)$$

Where:

- x_{ij}^k is a binary variable that equals 1 if EV k travels from node i to node j, and 0 otherwise.
- d_{ij} is the distance between nodes i and j.
- e_{ij} is the energy consumed by EV k when traveling from node i to node j.
- α and β are weights for distance and energy consumption, respectively.
- γ represents the revenue from selling power back to the grid.
- y_{ij}^k is a binary variable that equals 1 if EV k sells power to the grid at node j.

This objective function minimizes the total cost, which consists of three factors:

1. The distance factor, $\alpha d_{ij} x_{ij}^k$, minimizes the total distance covered by the fleet, aiming to reduce travel time and associated operational costs.
2. The energy factor, $\beta e_{ij} x_{ij}^k$, minimizes the energy consumption of the EVs, crucial for optimizing battery usage and preventing the need for recharging stops.
3. The revenue factor, $-\gamma y_{ij}^k$, maximizes the potential revenue from selling energy back to the grid, providing an additional stream of income for the fleet operator.

The problem is subject to the following constraints:

2.1. Fulfilling All Customer Orders

Each customer node must be visited exactly once by an EV:

$$\sum_k \sum_j x_{ij}^k = 1 \quad \forall i \in \text{Customers} \quad (2)$$

This constraint ensures that all customer deliveries are completed, with no customer left unvisited.

2.2. Battery Capacity Constraint

Each EV must complete its delivery route without exceeding its battery capacity, Q_k:

$$\sum_i \sum_j e_{ij} x_{ij}^k \le Q_k \quad \forall k \in \text{EVs} \quad (3)$$

This ensures that the energy consumed along the route does not surpass the vehicle's battery capacity.

2.3. Flow Conservation

Each EV must start and end its route at the depot:

$$\sum_j x_{0j}^k = 1, \quad \sum_i x_{i0}^k = 1 \quad \forall k \in \text{EVs} \quad (4)$$

$$t_i^k + S_i + d_{ij} \le t_j^k, \quad \forall i \in \text{Customers}, \forall k \in \text{EVs}, x_{ij}^k = 1 \quad (5)$$

$$\text{Ready}_i \le t_i^k \le \text{Due}_i, \quad \forall i \in \text{Customers} \quad (6)$$

This constraint ensures that each vehicle begins and finishes at the designated depot, preserving logistical consistency.

2.4. Optional Energy Sale to the Grid

EVs may optionally sell energy to the grid during their routes. This is represented by an additional binary variable y_{ij}^k, which equals 1 if EV k sells energy to the grid at node j:

$$y_{ij}^k \in \{0, 1\} \quad \forall i, j, k \quad (7)$$

This optional component introduces flexibility, allowing the fleet to act as a mobile energy source when the EVs are not fully utilizing their battery capacity.

2.5. Customer Time Windows

Each EV must arrive at a customer within their time window, and the service at each customer must start after the EV arrives. This can be formulated as:

$$t_i^k + S_i + d_{ij} \le t_j^k \quad \forall i, j \in \text{Customers}, \forall k \in \text{EVs}, x_{ij}^k = 1$$

Additionally, ensure that the start time at node i is between the ready time and the due date:

$$\text{Ready}_i \le t_i^k \le \text{Due}_i \quad \forall i \in \text{Customers}$$

- t_i^k: Continuous variable representing the start time at node i for EV k.
- $\text{Ready}_i, \text{Due}_i$: Ready time and due date for node i.
- S_i: Service time at node i

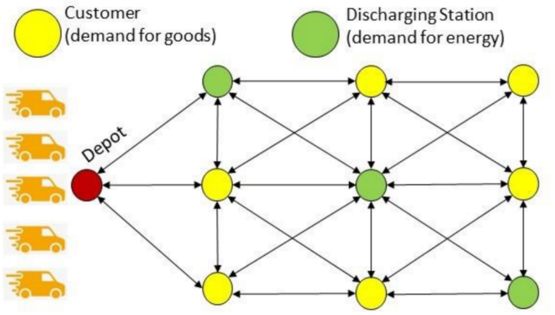

Figure 2: route mapping for optimal delivery

3. Theory of Traffic Route Optimization for Last-Mile Delivery

The mathematical formulation described above represents a variant of the classical Vehicle Routing Problem (VRP), commonly encountered in logistics. In this case, the objective is not just to minimize delivery costs but also to incorporate the opportunity for energy sale, leveraging EV fleets as both delivery vehicles and mobile energy sources.

The use of electric vehicles in logistics brings additional complexity. Traditional VRP approaches optimize routes based solely on factors like distance and time. However, with EVs, energy consumption and battery capacity constraints must also be factored into route planning. This requires a multi-objective optimization approach that balances the trade-off between delivering goods efficiently and generating revenue from energy sales.

By utilizing both MILP (Mixed Integer Linear Programming) and RL (Reinforcement Learning), we can develop and compare sophisticated routing algorithms that account for the dual objectives of minimizing trip costs and maximizing revenue from energy sales. These methods allow for real-time adjustments in routing based on dynamic factors such as customer demand, energy prices, and grid capacity.

4. Modifying the Solomon Dataset

The original Solomon dataset primarily includes customer locations, demands, distances, and time windows, but does not account for the additional constraints posed by electric vehicles (EVs). Therefore, the first step in modifying the dataset was to introduce an energy consumption column. This was done by calculating energy usage as a function of distance between customer locations and the vehicle's characteristics, such as its efficiency and load (assuming a linear relationship between distance and energy consumption).

Next, we added a binary attribute for nodes where grid interaction is possible, marking them as optional charging points within the network. This allows the model to make strategic decisions about whether to recharge at these points during a route. Lastly, we adjusted the vehicle capacity constraints to incorporate a battery limit for each EV. This modification ensures that the model takes into account both the energy limitations of the vehicles and the demands of customer locations.

5. Solving the MILP in Gurobi Solver

With the modified dataset, we set up the MILP model in Gurobi, which uses mathematical optimization to calculate the answer to a problem. We took five vehicles, battery capacity as 100 (in energy units), alpha (Weight for distance) as 5, beta (Weight for energy consumption) as 25, and gamma (Revenue from selling power per unit) as 4. For solution accuracy, we used the "MIPGap" attribute, which gives the difference between the best found solution and the best possible bound. After minimizing and extracting the solution, results are tabled below:

DATASET	OC	CT	SA
RC101	11369.09	420.65 sec	0.0000%
RC103	10385.84	275.58 sec	0.9990
RC102	10091.114	569.36 sec	0.9992%
C101	9263.44	0.48 sec	0.0000%

Table 1: Results from solving MILP using Gurobi. OC: Objective Cost, CT: Computational Time, SA: Solution Accuracy

6. RL Representation

The Electric Vehicle Routing Problem (EVRP) is a complex logistics challenge where the goal is to minimize operational costs while fulfilling customer delivery requirements and managing energy consumption. In this study, we use Reinforcement Learning (RL) to model the EVRP, treating it as a Markov Decision Process (MDP). The RL approach allows the agent (the EV) to learn optimal routing strategies by interacting with the environment.

6.1. Problem Formulation

The EVRP in the context of RL can be represented as follows:

- **State Space:** The state at any given time includes the current location of the vehicle, its battery level, time window constraints, and the status of deliveries (whether a customer has been served or not). This allows the agent to make informed decisions based on both the spatial and temporal aspects of the problem.

- **Action Space:** The action space consists of all possible moves the vehicle can take, such as traveling to a customer or a grid node (charging station). Actions are discrete, with each action corresponding to visiting a particular customer or charging point.
- **Reward Function:** The reward is designed to incentivize the vehicle to minimize energy consumption, meet delivery deadlines, and optimize the usage of energy. A negative reward is applied for high energy consumption, delays, or failing to complete deliveries within the required time window. The reward function can be expressed as:

  $$R(s,a) = \text{Energy Cost} + \lambda \cdot \text{Penalty for Delay} - \mu \cdot \text{Energy Sale Reward}$$

  where λ and μ are scaling factors, and the energy sale reward is earned when excess energy is sold back to the grid.
- **Transition Dynamics:** The transition from one state to another occurs when the vehicle takes an action, such as moving between locations, charging, or making a delivery. Each action leads to a new state, with the vehicle's battery being updated based on the distance traveled and the energy consumed.

By framing the EVRP as an MDP, the RL agent can learn to make optimal decisions that minimize the total energy cost and ensure timely deliveries.

6.2. Deep Q-Network (DQN) Agent Design

6.2.1. Agent Architecture

The RL agent is implemented using a Deep Q-Network (DQN), a neural network-based method that approximates the Q-value function, which estimates the future reward of taking a certain action in a given state. The DQN agent interacts with the environment to update its Q-values through reinforcement learning. The architecture of the agent is as follows:

- **Input Layer:** The input consists of a state vector containing key information such as the vehicle's current location, battery level, remaining delivery tasks, and the status of each grid node.
- **Hidden Layers:** The model has two hidden layers with 12 neurons each, employing the ReLU activation function to introduce non-linearity and enable the network to learn complex patterns.
- **Output Layer:** The output layer consists of n neurons, where n corresponds to the number of possible actions (i.e., visiting any customer or charging station). Each output represents the Q-value for that action.

The Q-value is used to determine the most optimal action for the agent to take, where the action with the highest Q-value is selected. The network is trained using the Adam optimizer and Mean Squared Error (MSE) loss function.

6.2.2. Exploration vs Exploitation

The DQN agent utilizes an epsilon-greedy strategy for exploration and exploitation. Initially, the agent explores the action space by selecting random actions with a high probability ϵ. As training progresses, ϵ decays, allowing the agent to increasingly exploit its learned strategy.

The epsilon decay is given by:

$$\epsilon_{\text{new}} = \max(\epsilon_{\text{min}}, \epsilon_{\text{start}} \cdot \epsilon_{\text{decay}}^{\text{episode}})$$

where $\epsilon_{\text{start}} = 1.0$ is the initial exploration rate, and $\epsilon_{\text{min}} = 0.05$ is the minimum exploration rate.

6.3. Replay Memory and Q-Value Update

The agent employs experience replay, storing its interactions (state, action, reward, next state, done) in a memory buffer. A mini-batch of experiences is sampled from this buffer to update the Q-values, which helps to break the correlation between consecutive experiences and stabilizes training.

The Q-value update follows the Bellman equation:

$$Q_{\text{target}} = \begin{cases} -reward & \text{if done} \\ -reward + \gamma \cdot \max Q_{\text{next}} & \text{if not done} \end{cases}$$

where γ is the discount factor that determines the importance of future rewards.

6.4. Training Process

The training process consists of the following steps:

1. The agent observes the current state s and selects an action a using the epsilon-greedy policy.
2. The agent performs the action, transitions to a new state s', and receives a reward r.
3. The agent stores the transition $(s, a, r, s', done)$ in the replay memory.

4. The agent samples a mini-batch from the replay memory and performs a Q-value update based on the Bellman equation.
5. This process repeats for a predefined number of episodes, where each episode consists of a trip in the environment.

The agent is trained over multiple episodes to improve its policy and minimize the energy consumption while fulfilling delivery requirements and energy sale opportunities.

6.5. Energy Parameters and Cost Calculation

The energy parameters are defined as follows:

- **Energy Selling Rate:** 0.00001 (rate at which excess energy is sold back to the grid).
- **Energy Consumption Rate:** 30 (energy consumed per unit distance traveled).
- **Battery Capacity:** 50 (maximum battery capacity of the EV).

The energy cost is calculated by factoring in the distance traveled, the energy consumed, and any energy sold back to the grid. The reward function penalizes high energy consumption while rewarding energy efficiency and timely deliveries.

6.6. Evaluation and Performance Metrics

The performance of the RL agent is evaluated based on its ability to minimize the total energy cost and meet delivery deadlines. The following metrics are used for performance evaluation:

- **Total Cost:** The cumulative energy cost incurred by the agent during an episode.
- **Energy Efficiency:** The efficiency with which the agent utilizes energy to complete deliveries.
- **Delivery Timeliness:** The ability of the agent to complete deliveries within the specified time windows.

By tracking these metrics, we assess the success of the RL approach in solving the EVRP and compare it with traditional methods.

7. Evaluation of Reinforcement Learning Approach

To test the performance of the aforementioned algorithms, we use the Solomon datasets as benchmarks, modified to suit the EVRPTW-D problem. These datasets are classified into two categories: Type 1 (vehicles with low capacities) and Type 2 (vehicles with larger capacities). Within each type, the dataset can have customer locations that are either clustered (CL), random (RA), or a combination of both (RC). For each standard dataset, we convert some of the existing customers into charging stations with a pre-specified timing window corresponding to the grid peak demand. For example, the standard dataset CL1100 comprises 90 customers and 10 discharging stations.

The experiment results are captured in Tables 2. Each row describes the average results obtained for datasets of a specific type (1 or 2), number of customers and discharging stations (25, 50, 100), and location distribution of customers (CL, RA, RC). We make comparisons using the objective value M, which is averaged over all datasets corresponding to a particular row.

DataSet	RL_OC	RL_CT	RL_SA
RC101	15143.32	0.9565 sec	0.0000%
RC103	15603.56	2.2880 sec	0.18%
RC102	15630.95	128.60 sec	0.04%
C101	16171.68	541.73 sec	0.0000%

Table 2: Performance results of RL on specific instances of Solomon datasets

From the table, we observe the following: - **RL** performs effectively with small gaps in solution accuracy, while being computationally efficient. - For instance, **RC101** yields an objective value of 15143.32 in just 0.9565 seconds, showing the efficiency of the **RL** approach.

These results highlight the potential of **RL** for rapid, near-optimal solutions to the EVRPTW-D problem, making it well-suited for real-time applications with tight computational constraints.

8. Comparison of MILP and RL Results

In this section, we compare the performance of the Mixed Integer Linear Programming (MILP) approach and the Reinforcement Learning (RL) approach applied to the same Solomon datasets modified for the EVRPTW-D problem. The comparison focuses on the objective cost (OC), computation time (CT), and solution accuracy (SA) for both approaches.

8.1. Comparison on Objective Cost (OC)

Table 8.4 shows the objective cost values obtained by both MILP and RL for different datasets. As expected, the MILP approach provides optimal or near-optimal solutions with very small MIP gaps. However, the RL approach, although not always optimal, achieves competitive objective costs, especially for larger datasets.

8.2. Comparison on Computation Time (CT)

A key advantage of the RL approach is its significantly lower computation time. Table 8.4 presents the computation times for each dataset. MILP requires several seconds to minutes to converge, while the RL approach consistently performs in fractions of a second to a few seconds. This makes RL highly suitable for real-time applications where computational efficiency is crucial.

8.3. Comparison on Solution Accuracy (SA)

Regarding solution accuracy, MILP achieves near-zero gaps in solution accuracy, indicating its optimality. On the other hand, the RL approach exhibits small MIP gaps, but these gaps are generally within acceptable ranges for practical use. For instance, for dataset RC101, the RL approach shows a gap of 0.0000% (optimal solution) and for dataset RC103, the gap is 0.18%, which is still very close to optimal.

8.4. Summary of Results

- **Objective Cost (OC):** MILP provides optimal or near-optimal solutions with objective costs consistently lower than those from RL. However, the gap is not significant enough to undermine the RL approach.

- **Computation Time (CT):** RL outperforms MILP by a large margin in terms of computational efficiency. While MILP requires several seconds to minutes for convergence, RL achieves results in seconds or less.

- **Solution Accuracy (SA):** MILP achieves near-zero solution accuracy gaps, confirming the optimality of its results. The RL approach, while having small gaps (e.g., 0.18% for RC103), is still considered highly accurate for practical applications.

In conclusion, while MILP provides optimal solutions with high accuracy, the RL approach excels in computational efficiency, making it a strong candidate for real-time decision-making in dynamic environments.

Dataset	MILP_OC	MILP_CT	MILP_SA	RL_OC	RL_CT
RC_103	15550.95	5288.22 sec	0.0000%	15143.32	0.9565 sec
RC_203	15603.56	6387.00 sec	0.18%	16401.52	2.2880 sec
R_107	5448.31	8.84 sec	0.0000%	6122.50	2.1772 sec
R_205	15630.95	128.60 sec	0.04%	16401.52	2.7567 sec
C_106	16171.68	541.73 sec	0.0000%	15460.00	3.2580 sec

9. References

References

[1] M. Solomon, *Algorithms for Vehicle Routing and Scheduling*, Operations Research, vol. 35, no. 2, pp. 211–222, 1987.

[2] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed., MIT Press, 2018.

[3] Z. Zhang and Z. Xu, *Electric Vehicle Routing Problem with Time Windows: A Review*, Transportation Research Part C, vol. 116, 2020, Art. no. 102648.

[4] J. Qian and S. Huang, *Optimizing Charging Station Locations in Electric Vehicle Routing Problems*, IEEE Transactions on Smart Grid, vol. 12, no. 3, pp. 2506–2517, 2021.

[5] C. Blum and A. Roli, *Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison*, ACM Computing Surveys, vol. 35, no. 3, pp. 268–308, 2003.

[6] E. W. Dijkstra, *A Note on Two Problems in Connexion with Graphs*, Numerische Mathematik, vol. 1, no. 1, pp. 269–271, 1959.

[7] T. Silver, D. Precup, and S. Singh, *Applications of Reinforcement Learning in Operations Research*, Journal of Artificial Intelligence Research, vol. 40, pp. 775–811, 2011.

[8] G. Laporte, *The Vehicle Routing Problem: An Overview of Exact and Approximate Algorithms*, European Journal of Operational Research, vol. 59, no. 3, pp. 345–358, 1992.

[9] V. Mnih et al., *Human-Level Control Through Deep Reinforcement Learning*, Nature, vol. 518, no. 7540, pp. 529–533, 2015.