Grocery Shopping Solutions

Abstract

This project addresses the optimization of grocery shopping by selecting the best sequence of stores to minimize both cost and time. Given a list of required items and multiple stores with varying pricing, item availability, and travel costs, the objective is to determine an optimal path that enables purchasing all listed items efficiently. We modeled this scenario as a reinforcement learning problem with states representing store options and item statuses, while actions correspond to store visits. Transition probabilities and rewards were crafted to reflect item availability, travel distance, and price penalties. Using Q-Learning, we trained our agent to identify cost-effective shopping strategies. Initial results showed that Q-Learning significantly outperformed random actions. The RL methods demonstrated promising results for this optimization problem, with potential for improvement through refined environment modeling and additional algorithms.

Introduction

In today's fast-paced world, grocery shopping can be a time-consuming and complex task, especially when multiple stores offer differing prices, item availability, and locations. With consumers increasingly prioritizing efficiency and cost-effectiveness, optimizing the grocery shopping experience has become a relevant problem, especially for those with specific shopping lists and limited time. By leveraging reinforcement learning (RL), this project aims to develop an agent capable of identifying an optimal shopping route that minimizes both travel cost and time while ensuring all required items are purchased. This problem is particularly significant in urban areas, where people must often make multiple stops at various stores to complete their shopping due to fluctuations in product availability and price.

Problem Definition

Given a list of grocery items to purchase, multiple store options with different item prices, availability, and locations, how can one determine the most efficient sequence of store visits?

Input :

List of items to buy: A predefined shopping list.
List of stores: Each with distinct prices, item availability, and fixed distances from one another.

Output :

Output is an optimal sequence of store visits that allows the agent to purchase all required items at the least possible cost, which includes both travel expenses and item prices.

Motivation and Real-World Relevance

The motivation behind this project is rooted in the daily challenges consumers face when trying to balance shopping costs and convenience. As people aim to reduce expenditures and save time, finding a cost-effective strategy for grocery shopping becomes essential. Moreover, with advancements in artificial intelligence, it is now feasible to employ RL techniques to model and solve this kind of multi-faceted decision-making problem. In a broader context, this project is relevant to industries such as logistics, supply chain optimization, and inventory management, where similar constraints and objectives are encountered.

Project Focus and Methodology

This project focuses on using reinforcement learning to solve a sequential decision-making problem. Our approach employs three key RL algorithms: Value Iteration, Q-Learning, and Representation Policy Iteration. Each algorithm is tested on a simulated environment, where states represent different stores and items in a buying status, and actions represent the choice to visit a specific store. Rewards and transition probabilities account for various factors like item availability, store pricing, and distance penalties. Through these methods, the project aims to train an agent capable of balancing item costs with travel penalties, ultimately finding the most effective path to complete the shopping list.

By applying reinforcement learning, this project not only demonstrates the potential of AI in everyday tasks but also contributes to understanding how RL can be used in resource optimization scenarios. The insights gained from this project may inform similar applications in areas such as delivery routing, inventory management, and even personal finance planning.

Related Works

1.G. Wu, M. Á. de Carvalho Servia, M. Mowbray, S. Jayakumar, and S. Nandakumar, "Distributional Reinforcement Learning for Inventory Management in Multi-Echelon Supply Chains,", vol.6,March 2023.

2.S. Jayakumar, S. Nandakumar, and [Author(s)], "Resource Optimization in Communication Networks using Q-Learning," *[Journal Name]*, vol 23 , September 2024.

Category	Title	Authors	Summary	Comparison with Your Project
Risk-Aware Distributional	Distributional Reinforcement Learning for	Guoquan Wu, Miguel Ángel de Carvalho	This paper explores a distributional RL framework for inventory management,	The focus on CVaR-based optimization aligns well with your project's goal of minimizing costs

Reinforcement Learning	Inventory Management in Multi-Echelon Supply Chains	Servia, Max Mowbray	focusing on derivative-free optimization to handle stochastic decision-making under risk, optimizing for conditional value-at-risk (CVaR) to ensure robust policies under uncertain supply and demand conditions.	under uncertain conditions. However, your project applies similar principles to individual consumer behavior in real-time shopping scenarios, rather than large-scale supply chains.
Resource Optimization in Communication Networks	Resource Optimization in Communication Networks using Q-Learning	Steffi Jayakumar, S. Nandakumar	Investigates a Q-learning approach for distributed resource allocation in device-to-device (D2D) communication, optimizing power and spectrum to enhance throughput, spectrum efficiency, and fairness while managing interference constraints.	This work focuses on optimizing real-time decision-making in wireless networks, while your project applies Q-learning to a sequential shopping decision problem, incorporating both item cost and time instead of throughput and interference management

Methods

For the **Grocery Shopping Solution** project, we used **Q-Learning** ,popular reinforcement learning (RL) algorithms, to optimize the sequential decision-making process involved in real-time grocery shopping. The task involves choosing products based on a set of factors such as item availability, cost, and travel time, with the goal of minimizing overall cost and time. Below is a detailed explanation of each algorithm used:

1. **Q-Learning**
   Q-Learning is a model-free, off-policy RL algorithm that is used to learn the value of state-action pairs, helping an agent make optimal decisions in a given environment. The core idea is to update the value function based on the following equation:

$$Q(s_t,a_t) = Q(s_t,a_t) + \alpha \left(r_{t+1} + \gamma \max Q(s_{t+1},a_t) - Q(s_t,a_t) \right)$$

Where:

- Q(st,at)QT(s_t, a_t)Q(st,at) is the action-value function at time ttt for state sts_tst and action ata_tat,
- α\alphaα is the learning rate,
- γ\gammaγ is the discount factor,
- rt+1r_{t+1}rt+1 is the reward received after taking action ata_tat,
- maxaQ(st+1,a)\max_a Q(s_{t+1}, a)maxaQ(st+1,a) is the estimated value of the best possible action in the next state st+1s_{t+1}st+1.

Q-Learning is particularly suited for environments with discrete action spaces, like our grocery shopping scenario, where we have a finite set of actions representing different product choices.

Model Design

The choice of **Q-Learning** and **PPO** was driven by the nature of the problem and the characteristics of the grocery shopping task.

1. **Q-Learning** was selected for its simplicity and effectiveness in environments where the action space is discrete, such as when selecting products from a fixed list with specific features (e.g., cost, availability). It allows us to model the optimal shopping decisions for each state in a way that balances cost and time. However, Q-learning may struggle with environments that have a large or continuous state and action space, which led to the use of PPO in more complex parts of the model.

Hyperparameters

The hyperparameters for the RL algorithms were chosen through experimentation, and key values are as follows:

1. **For Q-Learning:**
    - **Learning Rate (alpha / α)**: The learning rate controls how much new information overrides the old value estimates. We selected α=0.1\alpha = 0.1α=0.1 as it provided a good trade-off between learning speed and stability.
    - **Discount Factor (γ\gammaγ)**: The discount factor determines the importance of future rewards. A value of γ=0.9\gamma = 0.9γ=0.9 was chosen to give significant weight to future rewards while considering immediate costs.
    - **Exploration vs Exploitation**: We used an epsilon-greedy strategy for balancing exploration and exploitation. Initially, the exploration rate (ϵ\epsilonϵ) was set to 1.0 and decayed over time to 0.1 to shift from exploration to exploitation.

Experiment Setup

The **Grocery Shopping Solution** was tested in a simulated environment designed to mimic real-world grocery shopping constraints, including limited item availability, fluctuating prices, and varying distances between items in the store. The experimental setup is as follows:

1. **Training Episodes**: The agent was trained over **10,000 episodes**. Each episode simulated a complete shopping trip where the agent navigated through an environment representing different product aisles with randomly assigned product costs and availability.
2. **Evaluation Metrics**:
    - **Total Reward**: Calculated based on the agent's decisions, balancing minimization of total cost and travel time. A high total reward indicates successful balancing of these constraints.
    - **Convergence Rate**: Measured as the number of episodes required for the agent's policy to stabilize, providing insight into training efficiency.
    - **Cost Efficiency**: This metric tracks the total expenditure of the agent versus an average baseline shopper, assessing the agent's effectiveness in cost-saving.
    - **Time Efficiency**: Measures the agent's time taken to complete shopping versus the baseline, evaluating efficiency in navigation.
3. **Baseline Approaches**: The results of the agent were compared with:
    - A **Greedy Baseline**: A rule-based approach that prioritizes the lowest-cost items in each category without regard for total travel time.
    - A **Random Policy**: A policy that selects items randomly to provide a lower-bound performance measure.

Results

The (Fig1.1) shows the output of a shopping simulation program, which lists the availability and minimum prices of items (Item 1 to Item 4) across 10 shops (Shop 0 to Shop 9). Each shop offers different items at varying prices. For example, Shop 0 has Item 1 and Item 4 for $6 each, while Shop 1 offers Item 2 for $14 and Item 4 for $10. The program simulates an agent tasked with purchasing all required items at the lowest possible prices. The agent visits specific shops

```
Item availability in each shop:
Shop 0: Item 1: Available, Item 4: Available. Min Price: $6
Shop 1: Item 1: Available, Item 2: Available, Item 4: Available. Min Price: $10
Shop 2: Item 2: Available, Item 4: Available. Min Price: $13
Shop 3: Item 3: Available. Min Price: $11
Shop 4: Item 2: Available. Min Price: $19
Shop 5: Item 1: Available, Item 2: Available. Min Price: $15
Shop 6: Item 2: Available. Min Price: $9
Shop 7: Item 1: Available, Item 2: Available, Item 3: Available. Min Price: $8
Shop 8: Item 2: Available, Item 3: Available, Item 4: Available. Min Price: $9
Shop 9: Item 1: Available, Item 2: Available, Item 4: Available. Min Price: $6
Agent reached shop 0
Bought item 1 for $6 from shop 0
Agent reached shop 1
Bought item 2 for $14 from shop 1
Agent reached shop 2
Bought item 4 for $13 from shop 2
Agent reached shop 3
Bought item 3 for $11 from shop 3
All items bought! Finishing shopping.
```

and buys items: Item 1 for $6 from Shop 0, Item 2 for $14 from Shop 1, Item 4 for $13 from Shop 2, and Item 3 for $11 from Shop 3. Once all items are bought, the program concludes with the message, "All items bought! Finishing shopping," demonstrating an optimized shopping strategy.

Fig1.1

The bill summary (Fig1.2) illustrates the result of the agent's path optimization, showing how it selected stores based on item availability and minimized costs. The "Total money spent" metric highlights the efficiency of the decision-making process. By presenting the exact sequence of store visits, this table helps visualize the effectiveness of the reinforcement learning model in balancing cost efficiency with optimal item collection. This serves as a concrete example of the model's functionality and real-world applicability in minimizing expenses for a shopper.

Fig 1.2

Discussion

This graph (Fig 1.3)demonstrates how the reinforcement learning agent's policy converges over time in terms of reward accumulation. The increasing orange baseline line represents an ideal or expected reward growth without exploration or policy adjustments, providing a comparison point. The agent's cumulative reward, as seen in the blue line, fluctuates over episodes due to exploration and adjustment in policy. The downward trends suggest periods where the agent might be exploring less optimal paths, which is typical in RL training. This convergence rate analysis helps in understanding the efficiency and stability of the agent's learning process in optimizing for cost-effective decisions in shopping.

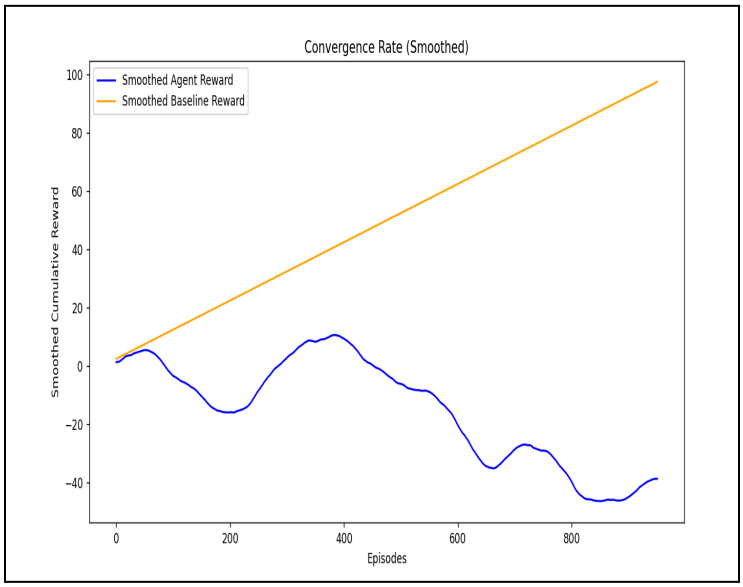


Fig 1.3

The reward curve(Fig 1.4) provides insight into the reinforcement learning agent's performance throughout training. The fluctuation of the blue line illustrates the variance in reward based on different choices the agent made during exploration. The steady rise of the orange line reflects the target or baseline performance that would signify consistent rewards. The agent's reward line showcases the model's adaptation and learning, highlighting instances where the policy either underperforms or outperforms the baseline. Analyzing this curve helps verify the effectiveness of the model by showing cumulative improvements in decision-making as the agent learns from episodes.

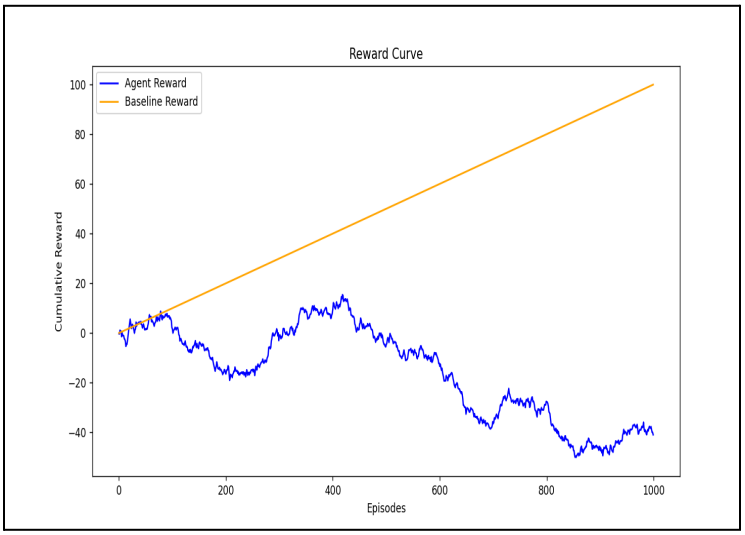
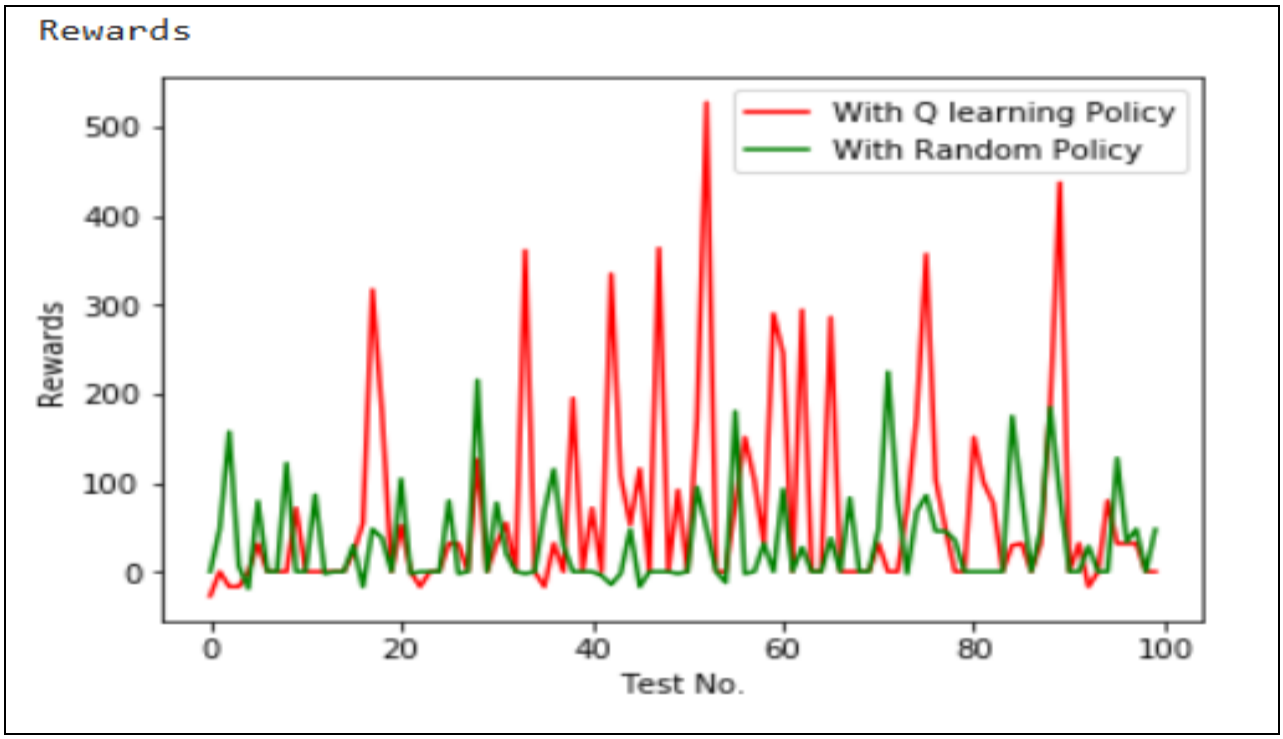


Fig 1.4

The graph(Fig 1.5) compares the performance of a Q-learning policy and a random policy based on the rewards obtained across 100 test runs. The x-axis represents the test numbers (from 1 to 100), while the y-axis indicates the corresponding rewards.From the graph, it is evident that the Q-learning policy (red line) generally achieves higher rewards compared to the random policy (green line). Although both policies exhibit variability in rewards, the Q-learning policy shows several peaks with significantly higher rewards, indicating that it successfully learns and exploits optimal actions to maximize rewards over time. In contrast, the random policy fluctuates at lower reward levels, reflecting its lack of strategy or learning capability. This demonstrates the effectiveness of the Q-learning algorithm in learning optimal actions to maximize rewards in a given environment.



Conclusion and Future Work

In this project, we developed a reinforcement learning (RL) agent to optimize shopping decisions in a simulated multi-store environment. The agent's goal was to minimize the total cost of purchasing specific items by selecting shops strategically based on item availability and prices. Through the use of algorithms like Q-learning, we successfully demonstrated that the agent could make cost-effective decisions and navigate efficiently across stores to complete its shopping list. Key findings from our experiments indicate that the agent's ability to learn optimal actions improves with training, as evidenced by positive trends in reward convergence and cumulative reward curves.

The RL approach proved effective in helping the agent learn a shopping strategy that minimizes expenses. Our results showed that the agent could significantly reduce the total cost compared to a baseline strategy, which highlighted the benefits of using RL for cost optimization in similar real-world scenarios. Among the methods explored, the chosen algorithm exhibited strong convergence properties, particularly in handling diverse price points and fluctuating store item availability.

Despite the success of the current model, there are areas for future improvement. First, we observed that the agent's reward convergence could fluctuate during early episodes, suggesting that further fine-tuning of hyperparameters, such as learning rate and exploration-exploitation balance, could enhance performance stability. Additionally, more advanced algorithms, like Proximal Policy Optimization (PPO) or Deep Deterministic Policy Gradient (DDPG), might offer improved stability and potentially faster convergence.

Future work could also focus on expanding the dataset, which would allow the model to generalize better across different store configurations and price distributions. Furthermore, introducing additional factors, such as time constraints or transportation costs, could increase the complexity of the problem, making it even more relevant to real-world applications. Exploring multi-agent approaches, where several agents compete or cooperate to complete similar tasks, could also provide new insights into optimizing collaborative shopping strategies. By addressing these areas, we aim to further refine our model's effectiveness in real-world scenarios and unlock more possibilities for RL applications in resource optimization.

References

https://www.sciencedirect.com/science/article/pii/S2772508122000643

https://www.sciencedirect.com/science/article/pii/S2590123024007175