目录

[The University of Hong Kong 1](#_Toc313487846)

[Group Member 1](#_Toc1012841631)

[Primary Supervisor: Ke Jintao 2](#_Toc1873906095)

[Abstract 3](#_Toc1892055410)

[1 Introduction 3](#_Toc1984914741)

[2 Literature Review（大家都写点） 5](#_Toc1451079489)

[3 Research Methodology 7](#_Toc1468676291)

[4 Experiment 17](#_Toc865384219)

[5 LLM For Driver-passenger Matching 19](#_Toc1753311249)

[6 Conclusion 19](#_Toc131557809)

[References 20](#_Toc1331423100)

**The University of Hong Kong**

**Master of Science in Artificial Intelligence**

**LLM-Powered Passenger-Driver Matching System**

**Group Member**

|  |  |
| --- | --- |
| **Fu Rao** | **3036216021** |
| **Su Qianhui** | **3036212879** |
| **Yang Wenhan** | **3036213536** |

**Primary Supervisor: Ke Jintao**

**Secondary Supervisor: Huang Chao**

**Abstract**

Driver-passenger matching is critical in modern transportation, aiming to optimize ride-sharing services by efficiently pairing drivers with passengers. This project begins by addressing the challenges of driver-passenger matching through the evaluation and comparison of several baseline approaches, including Random Assignment, Integer Linear Programming, Heuristic Algorithm, and Reinforcement Learning. Our experimental results reveal that reinforcement learning outperforms the other methods, particularly in dynamic and complex environments, demonstrating superior adaptability and performance. While these baselines provide a foundation, the core of our research is focused on exploring the potential of Large Language Models (LLMs) to further enhance the matching process. Specifically, we are utilizing the LangGraph framework, which allows for the integration of LLMs into complex decision-making pipelines. Through this approach, we aim to develop a more intelligent and adaptive driver-passenger matching system that can significantly improve the efficiency and effectiveness of ride-sharing services.

# 1 Introduction

## 1.1 Background

In recent years, the rapid growth of urbanization and the increasing demand for efficient transportation have made ride-sharing services an essential component of modern urban mobility. These services, provided by platforms such as Uber, Lyft, and Didi, have transformed the way people commute by offering a convenient and cost-effective alternative to traditional transportation modes. Central to the success of ride-sharing services is the driver-passenger matching process, which involves pairing drivers with passengers in real-time to optimize the use of available resources, minimize wait times, and maximize overall efficiency.

## 1.2 Problem Statement

Driver-passenger matching is inherently a complex problem, primarily due to the dynamic nature of the environment in which these services operate. Factors such as traffic conditions, demand variability, and the geographical distribution of drivers and passengers contribute to the complexity of making optimal matching decisions. Furthermore, the need to balance multiple objectives, such as minimizing travel distance, reducing passenger wait times, and ensuring fair distribution of work among drivers, adds another layer of difficulty to the problem. As a result, developing effective and efficient matching algorithms is crucial for the sustainability and scalability of ride-sharing platforms.

## 1.3 Existing Approaches

Over the years, various approaches have been proposed to tackle the driver-passenger matching problem. Traditional methods, such as Integer Linear Programming (ILP), have been widely used due to their ability to provide optimal solutions for well-defined, static problems. However, these methods often struggle with scalability and may not be suitable for real-time decision-making in highly dynamic environments. Heuristic algorithms, on the other hand, offer a more flexible approach by providing near-optimal solutions within a reasonable time frame, but they often lack the robustness required to handle the uncertainties inherent in real-world scenarios. Random assignment, while simple and fast, typically fails to achieve desirable levels of efficiency and effectiveness, making it less practical for large-scale applications.

In response to these challenges, reinforcement learning (RL) has emerged as a promising approach for solving the driver-passenger matching problem. RL algorithms are useful for dynamic environments because they learn from interactions with the environment and can adapt their strategies based on real-time feedback. By continuously updating their policies, RL-based models can effectively manage the trade-offs between competing objectives and optimize the overall performance of the ride-sharing system. Our preliminary experiments indicate that reinforcement learning outperforms traditional methods, especially in complex and dynamic settings, demonstrating its potential as a robust solution for driver-passenger matching.

## 1.4 Integration of LLMs

While reinforcement learning has shown significant promise, we believe that the integration of Large Language Models (LLMs) could further enhance the decision-making capabilities of ride-sharing systems. LLMs, with their advanced natural language processing (NLP) capabilities, can be leveraged to process and interpret vast amounts of unstructured data. By incorporating this additional layer of information into the matching process, LLMs can help create more informed and context-aware decisions, potentially leading to improved efficiency and user satisfaction.

To facilitate the integration of LLMs into the driver-passenger matching process, we are utilizing the LangGraph framework. LangGraph provides a flexible and scalable platform for incorporating LLMs into complex decision-making pipelines, enabling the dynamic generation and refinement of matching strategies. This approach allows for the seamless integration of structured and unstructured data, offering a comprehensive solution that leverages the strengths of both reinforcement learning and advanced NLP techniques.

## 1.5 Research Objectives

The objective of this research is to develop a more intelligent and adaptive driver-passenger matching system that can effectively handle the complex real-world environments. By combining the strengths of reinforcement learning and Large Language Models, we aim to create a system that can improve the efficiency and effectiveness of ride-sharing services and enhance the overall user experience.

This report will outline the current progress of our research, including the implementation and evaluation of baseline approaches including Random Assignment, Integer Linear Programming, Heuristic Algorithm, and Reinforcement Learning, and the initial steps towards integrating LLMs into the matching process.

The remainder of this report is structured as follows. Section 2 provides a comprehensive literature review, highlighting the existing research on driver-passenger matching and related optimization techniques. Section 3 details the research methodology, including the research design, data overview and the tools and techniques used in this study. Section 4 discusses the implementation of the baseline models, followed by an analysis of the experimental results. Section 5 presents the initial findings related to the integration of LLMs and outlines the future research directions. Finally, Section 6 concludes the report with a summary of the key insights and potential implications of this research.

# 2 Literature Review（大家都写点）

**Overview of existing research**

To provide more insights for our research on leveraging Large Language Models (LLMs) to assist in passenger-vehicle matching decisions, we conducted an extensive review of existing literature and methodologies. Among the numerous studies we examined, two papers have been particularly influential in shaping our approach.

The first paper, titled "LLMLight: Large Language Models as Traffic Signal Control Agents," explores the use of LLMs as decision-making agents in traffic signal control (TSC). While traditional reinforcement learning (RL) methods have been employed to optimize traffic conditions through adaptive strategies, these methods face challenges when scaling to large and complex urban environments.

To address these limitations, researchers have recently begun exploring the use of LLMs as decision-making agents within traffic signal control systems. LLMs, with their ability to learn and reason about complex environments from extensive datasets, have shown significant potential in generating sophisticated decision-making strategies. Their capacity to model contextual dependencies, process sequential data, and generate reasonable predictions makes them particularly well-suited to tackle traffic management challenges that traditional methods struggle with.

Existing research in this field has demonstrated the potential of LLMs to enhance dynamic TSC. By integrating LLMs into traffic control frameworks, researchers aim to leverage their advanced language understanding capabilities to generate optimized traffic control strategies. Initial studies have explored hybrid approaches that combine RL with LLMs, proposing architectures where LLMs generate trajectories or suggestions that are further refined by action-value networks.

The second key paper in our research, "LLM-Assisted Light: Leveraging Large Language Model Capabilities for Human-Mimetic Traffic Signal Control in Complex Urban Environments," investigates an innovative approach to traffic signal control by integrating LLMs into the decision-making process. This method seeks to address the limitations of traditional rule-based TSC systems and existing RL-based methods by harnessing the human-like reasoning capabilities of LLMs to improve traffic management in complex urban settings.

The core idea presented in this paper is the LA-Light framework, which leverages the extensive knowledge and reasoning abilities of LLMs to enhance the decision-making processes of TSC systems. LLMs are tasked with interpreting complex traffic scenarios in real-time and generating signal control suggestions that are adjusted and verified by decision-making tools before being implemented in the traffic system. The framework operates within a closed-loop system, where various interoperable tools, including traffic analysis devices, sensors, and actuators, support real-time analysis and decision-making.

The paper demonstrates the effectiveness of LA-Light through simulations conducted on platforms such as SUMO, evaluating its performance against traditional and RL-based methods. The experiments highlight LA-Light’s superior ability to reduce average travel time (ATT) and average wait time (AWT) by generating more efficient, safer, and smoother traffic control decisions, particularly in complex and dynamic traffic environments.

This paper makes a significant contribution by introducing a real-time decision-making system that integrates LLMs into urban traffic signal control. The results indicate notable improvements over existing methods, suggesting that LA-Light and similar LLM-based approaches could play a pivotal role in future urban traffic management systems as LLM technology continues to advance.

# 3 Research Methodology

## 3.1 Research Design

Our research aims to enhance the efficiency and success rates of one-to-one passenger-vehicle matching, a critical problem in urban mobility. To address this, we have developed four baseline methods: random selection, heuristic approaches, integer linear programming (ILP), and reinforcement learning (RL). These baseline models will serve as benchmarks to evaluate the performance of large language model (LLM)-based agents in optimizing trip selection and matching processes.

Beyond traditional methods, we plan to integrate LangGraph to facilitate trip allocation decisions within both single-agent and multi-agent frameworks. This dynamic approach allows us to explore a more flexible and adaptable decision-making process in managing passenger-vehicle assignments, especially in complex urban environments.

To further optimize LLM performance for this specific task, we will first design simplified dispatch scenarios, such as binary classification problems, using prompt engineering techniques. This will help refine the output of LLMs before deploying them in more complex, real-time decision-making scenarios. In addition, to accelerate our experimentation and optimization process, we plan to work with smaller datasets, reduce the geographical scope of the simulation, and shorten the duration of simulation episodes, allowing for faster iteration and performance tuning.

If time permits, we intend to explore alternative datasets that may better align with the one-to-one matching paradigm. Additionally, we are investigating advanced fine-tuning techniques, such as Supervised Fine-Tuning (SFT) and Reinforcement Learning with Human Feedback (RLHF), to further tailor LLM behavior and enhance their ability to make efficient and successful matching decisions. Through this comprehensive approach, we hope to significantly improve the outcomes of passenger-vehicle matching in dynamic urban settings.

## 3.2 Data Overview

The data used in this study originates from Chengdu, China. The dataset is comprised of road network data, provided in GraphML file format, and trip data contained within a CSV file. These datasets were obtained through a proprietary dataset provided by a research partner. These datasets together form the basis for analyzing and optimizing driver-passenger matching in the context of ride-sharing services.

The road network data contains detailed information about the city's roads, intersections, and the connections between various locations, representing Chengdu's road infrastructure in a graph format. In this graph, each node represents an intersection or a point on the road, while the edges represent the road segments connecting these points.

The trip data contains detailed information for a series of ride-sharing requests, which is used as training or testing data for developing and optimizing driver-passenger matching strategies. The dataset includes the following columns:

|  |  |
| --- | --- |
| Attribute | Descrition |
| order\_id | A unique identifier for each ride request. |
| origin\_id | Identifiers for the origin and destination locations of each trip. |
| dest\_id |
| origin\_lat | Latitude and longitude coordinates for the trip's starting and ending points. |
| origin\_lng |
| dest\_lat |
| dest\_lng |
| trip\_distance | The distance between the origin and destination in kilometers. |
| start\_time | The time at which the trip request was made. |
| origin\_grid\_id | Grid identifiers for the origin and destination locations, used for spatial indexing and matching. |
| dest\_grid\_id |
| itinerary\_node\_list | A list of nodes representing the planned route from origin to destination. |
| itinerary\_segment\_dis\_list | A list of segment distances corresponding to the nodes in the itinerary. |
| trip\_time | The total travel time for the trip. |
| designed\_reward | The reward designed for completing the trip. |
| cancel\_prob | The probability of the trip being canceled. |

## 3.3 Simulator

**Objectives and Functionality**

The ride-sharing simulator is designed to replicate and optimize the process of matching passengers with vehicles within a ride-sharing ecosystem. The simulator aims to assess various strategies for improving matching efficiency and operational performance. It enables detailed analysis of how different dispatching algorithms and decision-making processes impact overall system effectiveness, providing valuable insights for optimizing ride-sharing operations.

**Components and Architecture**

The simulator is comprised of six key modules, each playing a distinct role in the simulation process:

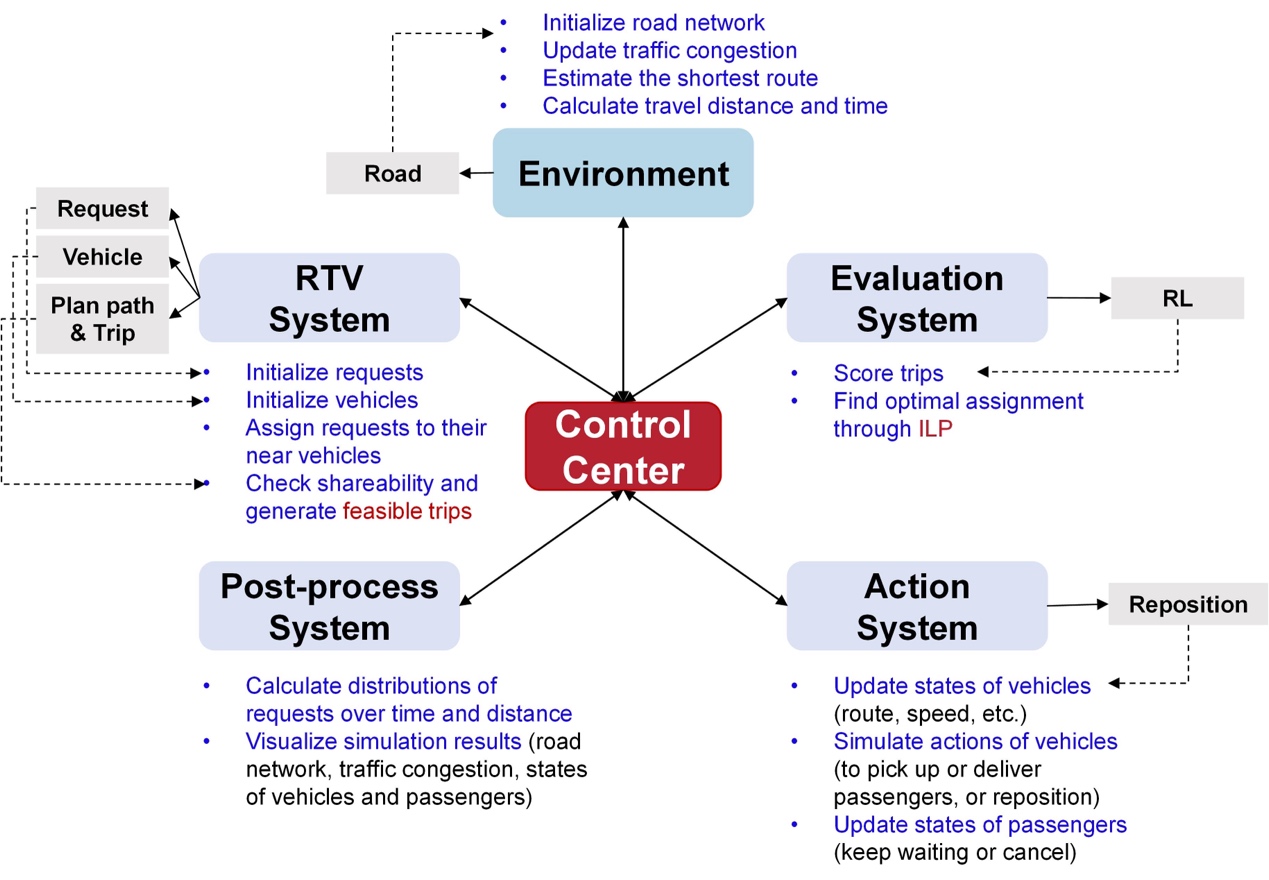


Figure n: **Architecture of Simulator**

* Action System: The Action System module is responsible for managing the dynamic actions of vehicles within the simulation. It handles crucial functions such as vehicle repositioning, path updates, and real-time action simulation. This module integrates with the environment model to ensure vehicles effectively respond to passenger requests and navigate the simulation space. Key responsibilities include managing vehicle states, simulating vehicle actions (e.g., picking up or delivering passengers), and updating passenger states (e.g., waiting or cancellation).
* Control Center: Serving as the central coordination unit, the Control Center module oversees the overall simulation management. It integrates with other subsystems to handle request allocation, trip generation, and vehicle management. Responsibilities include initializing the simulation, managing parameters, allocating requests to vehicles, generating feasible trips, and evaluating trip performance. The Control Center is crucial for processing unmatched requests, scoring decisions, and producing visualizations of simulation results.
* Environment: The Environment module sets up and manages the simulation's physical and spatial context. It includes initializing the road network, managing vehicle distributions, and simulating traffic conditions. This module tracks vehicle and request hot spots, calculates distances, and updates traffic congestion. The Environment module ensures realistic simulation of road conditions and travel dynamics.
* Evaluation System: The Evaluation System module focuses on assessing the simulation's performance. It calculates metrics such as service rates, average response times, and distances traveled by vehicles. This module evaluates the effectiveness of different matching strategies and provides insights into overall system performance. It uses various performance indicators to analyze and optimize the simulation results.
* RTV System: The RTV System module is responsible for initializing and managing requests and vehicles. It handles request allocation to vehicles, generates feasible trips, and manages vehicle repositioning. This module integrates components such as Trip, Path, and Reposition to ensure efficient handling of request data and path planning. It plays a critical role in reflecting realistic operational scenarios within the simulation.
* Post Process System: The Post Process System is a subsystem within the control center dedicated to visualizing simulation results. It provides functionality for calculating request distributions over time and distance, visualizing the road network, traffic congestion, and the states of vehicles and passengers. Additionally, it converts simulation time into real-time and creates videos from a series of images, facilitating a comprehensive review of the simulation outcomes.

**Simulator Setup**

**图形用户界面, 表格

描述已自动生成**

Figure: part of the setup of the simulator for the baseline methods

The initialization phase of the simulator involves several critical steps to establish a fully functional simulation environment. The process begins with loading the configuration settings from a specified file, which outlines essential parameters such as the time range for the simulation, request generation rates, and vehicle specifications. Once the configuration is loaded, the simulator proceeds to initialize the environment, which includes setting up the road network, distributing vehicles, and establishing traffic conditions based on the provided data.

Following the environment setup, the simulator configures various parameters that influence its operation. This includes defining time step intervals, specifying vehicle attributes, and ensuring that all components are correctly established to reflect realistic conditions. The environment configuration plays a crucial role in shaping the simulation, as it encompasses details such as the layout of the road network, the initial positions and capacities of vehicles, and the attributes of passenger requests, including their generation rates and locations. These configurations are integral to the simulation, as they affect traffic patterns, vehicle availability, and the handling of passenger requests throughout the simulation period.

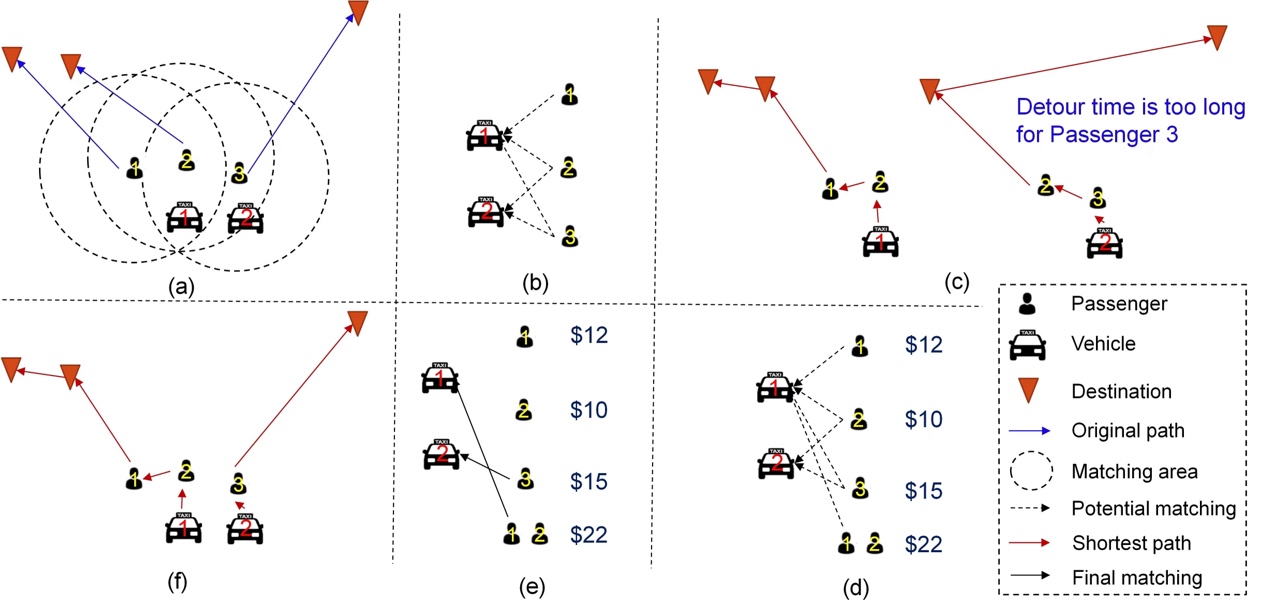
**Simulation Process**

The simulation process is designed to effectively model the interactions between vehicles and passengers over time, capturing the dynamics of the system in a structured manner. It begins with the simulation of time progression, where the simulator advances through discrete time steps to represent the passage of time within the simulation period. This approach allows for a detailed examination of the behavior of both vehicles and passengers at each time interval.

During each time step, the simulator processes a variety of events that influence the overall system. These events include the generation of new passenger requests, updates to vehicle positions based on their movements, and the allocation of requests to available vehicles. The simulator is equipped to handle these events by applying a series of rules and constraints that reflect real-world operational challenges, such as route planning, pickup and delivery times, and detour constraints.

The decision-making process within the simulation involves several key aspects. The simulator assesses the feasibility of vehicle assignments to passenger requests by evaluating potential routes and ensuring that constraints such as pickup times and detour durations are met. This process is crucial for optimizing the allocation of vehicles and ensuring that requests are handled efficiently. The simulator also incorporates decision-making algorithms that determine the best course of action for each vehicle, considering factors such as route efficiency and operational constraints.

**Dispatching and Matching Algorithms**

****

**Figue n: Dispatching Algorithm**

1. **Preassignment of Passengers to Vehicles:** Passengers are initially preassigned to vehicles based on their spatial proximity within designated matching areas. This preassignment step narrows down potential vehicle-passenger pairs, facilitating a more targeted evaluation of vehicle allocation.
2. Scheduling Multiple Passengers: Each vehicle is then assessed for its capacity to accommodate multiple passengers. This evaluation involves determining whether a single vehicle can efficiently manage several requests, considering constraints such as vehicle capacity and operational limits.
3. Verification of Shareability: The shareability of potential passengers assigned to the same vehicle is scrutinized. This involves planning the shortest routes and ensuring that pickup and detour time constraints are adhered to. For instance, while passengers 1 and 2 may be compatible for sharing a vehicle, passengers 2 and 3 might not be feasible for the same vehicle due to detour time limitations.
4. RTV-Graph Construction: An RTV-graph is constructed to link all potential trips, which may involve multiple requests, to available vehicles. This graph provides a systematic framework for evaluating possible matches and ensures comprehensive consideration of all trip combinations.
5. Optimal Matching Approaches: The optimal vehicle-passenger matching is achieved through various methods, each serving as a baseline model for comparison:
6. Random Selection: Vehicles are assigned to requests randomly, without considering optimization criteria. This approach serves as a baseline for comparison with more sophisticated methods.
7. Reinforcement Learning (RL): Machine learning algorithms are employed to optimize matching through iterative training and evaluation. RL techniques learn from the simulation environment to improve matching decisions over time.
8. Integer Linear Programming (ILP): The matching problem is formulated as an integer linear program, which is solved to find the optimal solution. This approach considers various criteria, such as minimizing travel time and maximizing vehicle utilization, to determine the best vehicle-passenger assignments.
9. Heuristic Methods: Practical strategies or rules of thumb are applied to assign vehicles efficiently. These methods leverage domain-specific knowledge to provide effective, though not necessarily optimal, solutions.
10. Execution of Vehicle Assignments: Vehicles are directed to pick up and deliver passengers according to the optimal matching results obtained from the chosen method and the pre-planned shortest routes. This ensures that vehicle-passenger assignments are executed efficiently while adhering to the constraints and objectives established in the dispatching algorithm.

**Evaluation Methods**

For our current task, which focuses on a one-to-one matching scenario (where each vehicle is paired with a single passenger), rather than ride-pooling, the following evaluation methods are particularly important and representative:

* Service Rate (Non-Ride-Pooling): This metric measures the system's service capability in a non-ride-pooling mode, reflecting the ratio of successfully matched passengers to vehicles. In our context, this indicator is crucial for assessing the system's fundamental efficiency, as it directly impacts service coverage and overall matching effectiveness.
* Average Assignment Time: This metric describes the average time required to assign passengers to vehicles. It reflects the system's efficiency in handling requests and allocating resources. A lower assignment time indicates that the system can respond quickly to passenger requests, thereby improving service speed.
* Average Pick-Up Time: The average pick-up time represents the average duration from when a passenger requests a ride to when they are picked up by the vehicle. This metric directly affects the passenger experience, with shorter pick-up times generally indicating higher service quality.
* Total Travel Distance of All Vehicles: This metric shows the total distance traveled by all vehicles to complete the assigned tasks. It helps in understanding the overall resource consumption in the transportation process and can be used to evaluate the impact of optimized matching strategies on operational costs.

## 3.4 **Model Description**

### 3.4.1 Random Assignment

The random baseline method employs a straightforward approach to trip selection by randomly assigning feasible trips to vehicles. For each vehicle, this method selects a trip at random from the available feasible options, without considering the quality or suitability of the trips. The decision is based solely on a random index within the list of feasible trips.

While the random baseline method is both simple and computationally inexpensive, its effectiveness as a standalone solution is limited by its lack of optimization and contextual consideration. This method does not account for the relative merits of different trips or adapt to specific vehicle requirements, potentially leading to suboptimal performance. It primarily serves as a benchmark for comparing the effectiveness of more sophisticated trip selection strategies.

### 3.4.2 Integer Linear Programming

As one of the baseline methods, we implemented a trip assignment strategy based on Integer Linear Programming (ILP) to optimize the matching between vehicles and passengers. This method formulates the trip assignment problem as an integer linear programming problem, with the objective of maximizing the assignment score while adhering to vehicle and passenger constraints.

First, for each vehicle and its corresponding feasible trips, we created a binary decision variable that indicates whether the vehicle selects a particular trip. Each variable's coefficient is determined by the score of the trip, and the objective function is the sum of the scores of all selected trips. By solving this objective function, we aim to assign the optimal trip to each vehicle at every time step.

To ensure the rationality of the assignment, we introduced two main constraints. The first constraint ensures that each vehicle can select at most one trip within a time step, i.e., the sum of all decision variables for each vehicle must equal one. The second constraint ensures that each passenger’s request can only be fulfilled by one trip; hence, for all trips containing a particular passenger, the sum of the corresponding decision variables must be less than or equal to one.

By solving the model with a local solver, we generate an optimal matching solution between vehicles and trips. After solving the model, we extract the optimal solution and convert it into actual vehicle trip assignments. If no valid solution is found at a certain time step, a default assignment scheme is returned.

This ILP-based method provides an effective global optimization approach, ensuring that vehicles are assigned to trips with the highest possible scores under certain constraints. However, while the ILP model guarantees optimality, its computational cost may be high in large-scale scenarios.

### 3.4.3 Heuristic Algorithm

The heuristic-based trip selection method aims to optimize vehicle-to-trip assignments by leveraging a heuristic approach to maximize overall efficiency. This method involves selecting trips for vehicles based on a heuristic evaluation of the available options, taking into account the quality of the trips and the need to avoid conflicts with previously assigned requests.

Methodology: In this approach, for each vehicle, the method iterates through the list of feasible trips, evaluating each trip based on its score. The heuristic prioritizes trips that have not already been assigned to other vehicles, ensuring that no request is duplicated across multiple vehicles. Specifically, the method maintains a record of assigned requests and skips any trip whose requests intersect with those already allocated. Among the remaining feasible trips, the one with the highest score is selected for the vehicle.

Process

1. Initialization: The method initializes an empty set to keep track of assigned requests.
2. Trip Selection: For each vehicle, it evaluates each trip based on its score and checks if the trip's requests have already been assigned. It selects the trip with the highest score that does not overlap with previously assigned requests.
3. Assignment: The chosen trip and its corresponding path are added to the final lists. The requests of the chosen trip are then updated in the assigned requests set.
4. Fallback: If no suitable trip is found, the method assigns a default "null" trip with a score of zero.

The heuristic method offers a balance between simplicity and effectiveness by incorporating a basic optimization criterion—selecting the highest-scoring trip while avoiding request duplication. However, it is limited by its heuristic nature and may not always achieve the optimal global solution. The method's performance depends on the quality of the scoring mechanism and its ability to handle complex, overlapping request scenarios.

### 3.4.4 Reinforcement Learning

Reinforcement Learning (RL) is a key component of our driver-passenger matching system, designed to optimize decision-making in a dynamic and complex urban environment. Unlike traditional optimization methods that rely on predefined rules or static models, RL allows the system to learn and adapt through interactions with the environment, making it useful for scenarios where conditions change frequently and unpredictably.

The RL model used in this study is primarily based on a Q-learning framework, where the objective is to learn a policy that maximizes the expected cumulative reward for each state-action pair. The policy is represented by a Q-function, which estimates the expected reward for taking a specific action in a given state. The Q-function is approximated using a neural network model.

The neural network architecture for the Q-function approximation consists of an input layer, three fully connected layers, each responsible for processing different aspects of the state information, and an output layer, as illustrated in Figure n.

The input layer takes in positional and temporal data, such as the current location of vehicles and passengers, as well as the current time step. These inputs are then processed through the three fully connected layers, which are designed to capture complex relationships and dependencies within the state information. The output layer generates a set of Q-values, each corresponding to a potential action the agent can take in the given state. The Q-value represents the expected cumulative reward for taking that action, guiding the agent towards decisions that optimize the driver-passenger matching process.

图示, 示意图

描述已自动生成

Figure n: Q-Value Approximation Network

# 4 Experiment

## 4.1 Experimental Setup

The experiments were conducted on a high-performance computing setup with the following specifications:

**● Processor**:

**● Memory**:

**● Graphics Processing Unit (GPU)**:

**● Operating System**:

**● Software**:

**Python 3.9**: The programming language used for model development and simulation.

**PyTorch**: Used for implementing the neural network models and RL algorithms.

**NetworkX**: Employed for handling and analyzing the GraphML road network data.

**OpenAI Gym**: Used for creating and managing the simulation environment.

（这里按实际改）

**Matplotlib**: For visualization of the results.

The use of a GPU was critical in accelerating the training process, particularly for the reinforcement learning model and large language model.

## 4.2 Preliminary Results

The preliminary results of our experiments offer valuable insights into the effectiveness of various driver-passenger matching strategies under the condition where all users opt for non-pooling rides (i.e., one car per passenger). The primary metrics used for evaluating and comparing the models are the service rate (representing the proportion of trip requests successfully fulfilled), average assigning time (minutes), average pick-up time (minutes), average detour time (minutes), and total income of all vehicles (USD).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Method | **Service Rate** | Average assigning time (min) | Average pick-up time (min) | Average detour time (min) | Total income of all vehicles (USD) |
| Random Assignment | 0.4268 | 3.49 | 3.40 | 4.56 | 25786.00 |
| Integer Linear Programming | 0.3739 | 2.51 | 3.40 | 2.66 | 25921.20 |
| Heuristic Algorithm | 0.3855 | 2.38 | 3.56 | 2.45 | 26376.40 |
| Reinforcement Learning | 0.4291 | 2.29 | 3.27 | 1.54 | 28779.54 |

Baseline Method Comparison

The Random Assignment method exhibits a relatively higher service rate of 0.4268, indicating that it successfully matches a larger proportion of requests compared to other methods. However, this method ranks lowest across all other performance indicators. The high service rate observed can be attributed to its purely random nature, which, while ensuring that some requests are matched, does not consider optimization. This lack of optimization results in inefficiencies and lower overall performance.

Integer Linear Programming (ILP) and Heuristic Algorithm methods show intermediate performance across the metrics. ILP performs better in terms of average pick-up time (3.40 minutes) compared to the Heuristic Algorithm. This suggests that ILP might be more effective in minimizing travel delays, thereby improving the efficiency of vehicle routes. Conversely, the Heuristic Algorithm demonstrates a better service rate of 0.3855 and a lower average assigning time of 2.38 minutes compared to ILP. This indicates that heuristic methods might offer a good balance between speed and service quality, likely due to their practical rules and adjustments based on domain knowledge.

Reinforcement Learning (RL) clearly outperforms all other methods across every metric analyzed. It achieves the highest service rate of 0.4291, the shortest average assigning time of 2.29 minutes, the lowest average pick-up time of 3.27 minutes, the lowest average detour time of 1.54 minutes, and the highest total income of $28,779.54. RL’s superior performance can be attributed to its dynamic learning and adaptation capabilities, which optimize decision-making and resource allocation over time. This results in superior overall efficiency and effectiveness in managing passenger-vehicle matching.

To conclude, while the Random Assignment method yields a higher service rate, it underperforms in other metrics, rendering it the least effective overall. The ILP and Heuristic methods offer a mix of strengths and weaknesses, with ILP excelling in specific aspects of route efficiency and Heuristic methods providing a better balance between speed and service quality. However, RL stands out as the most effective method, demonstrating the highest performance across all metrics. Its ability to learn and adapt to complex scenarios makes it the superior choice for optimizing passenger-vehicle matching.

# 5 LLM For Driver-passenger Matching

## 5.1 Work-flow of LLM for Driver-passenger Matching

## 5.2 Input to LLM from Simulator

To mitigate the decision-making burden on LLMs, our simulator employs a sophisticated process of information simplification and filtration. This refined data is presented to the LLM, with subsequent supplementation of decision-relevant information as needed. Additionally, the LLM is empowered to autonomously acquire information through the utilization of tools such as POI (Point of Interest) finding.

Primarily, we provide the LLM with essential passenger and taxi information, structured in dictionary format. This foundational data serves as the basis for subsequent decision-making processes.

Secondarily, we consider feasible trips, employing a two-stage filtering process. Initially, we eliminate driver-passenger pairs that exceed a predetermined distance threshold. Subsequently, we conduct a more granular analysis on a per-vehicle basis. This refined data structure comprises:

Information on passengers who have already boarded the vehicle (e.g., Passenger A)

Data on passengers who have made reservations but have not yet boarded, having already undergone the decision-making process (e.g., Passenger B)

Information on passengers awaiting matching (e.g., Passenger C)

This design paradigm accounts for the complex dynamics of ride-sharing, including potential queue-jumping scenarios. For instance, Route No. 1 might encompass the following sequence: Current vehicle location → B's pickup point → A's destination → C's pickup point → B's destination → C's destination

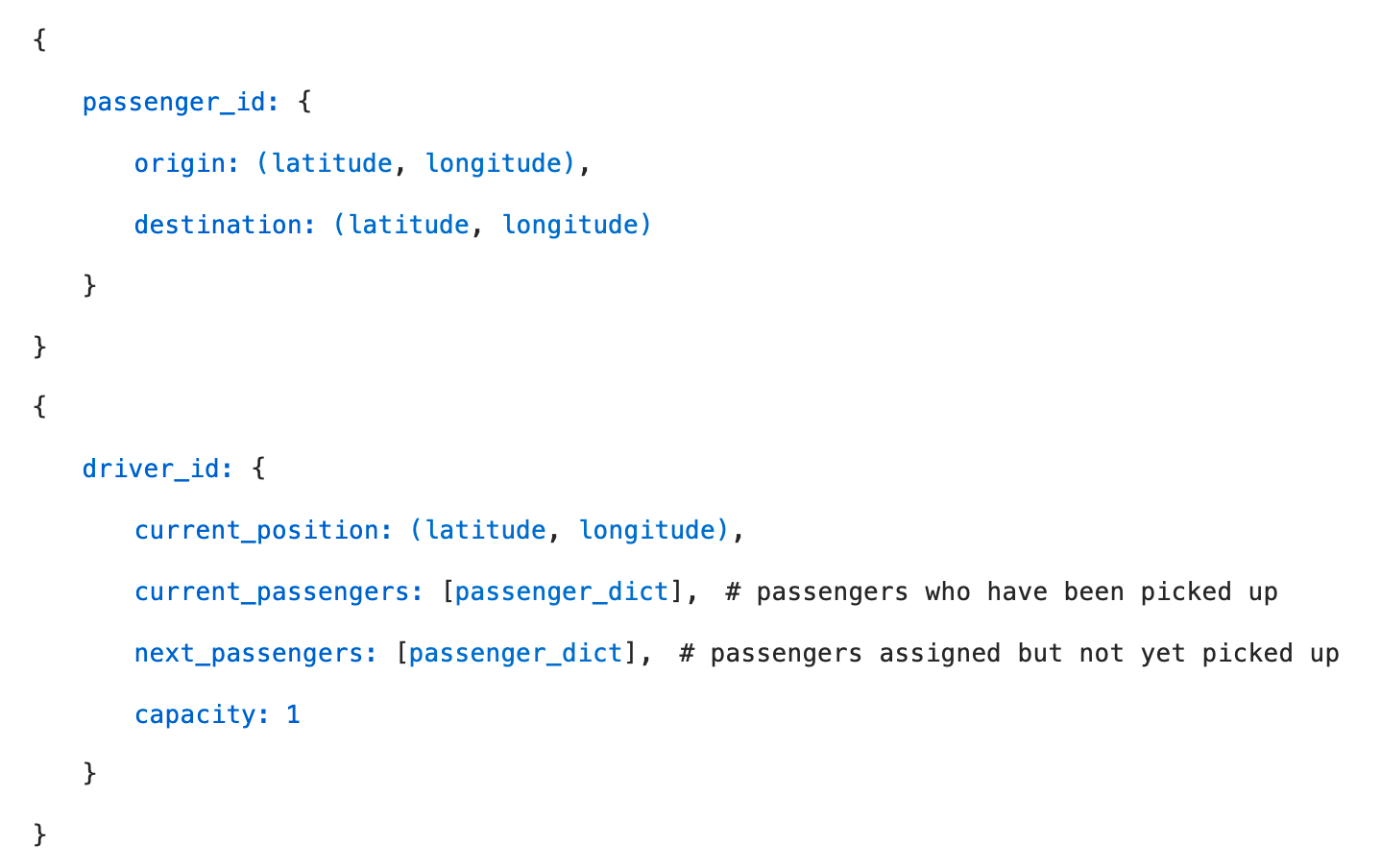
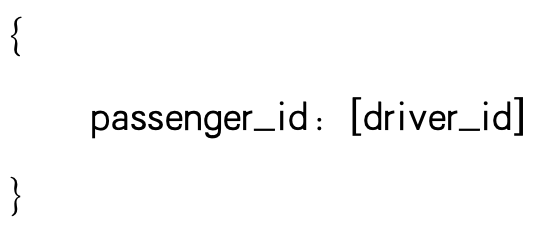
However, if the decision-maker determines that the vehicle's current position is in closer proximity to Passenger C before picking up Passenger B, Route No. 2 becomes a viable alternative:

Current vehicle location → C's pickup point → B's pickup point → A's destination → B's destination → C's destination

It's crucial to note that Route No. 2 may result in a significant delay for Passenger A. If this delay exceeds a predefined tolerance threshold, Route No. 2 would be deemed unfeasible and subsequently disregarded.

While route planning undoubtedly influences the matching decision process, for the sake of simplification, we do not consider multiple route permutations arising from passengers within the same decision space. Our primary objective is to achieve efficient passenger-to-driver matching.

Consequently, the initial input to the LLM is structured as follows:

**Figue n: Basic Information(left)**

**Decision Space (prioritizing passengers in a supply-exceeds-demand scenario, selecting one vehicle from multiple options)**

This structured approach to data presentation and decision-making enables the LLM to efficiently process and optimize ride-sharing scenarios, balancing the complexities of multi-passenger routing with the need for computational efficiency.

## 5.3 Preliminary Analysis for LLM

Given the complexity of the agent framework, which is still in the development, detailed outputs are provided in the Appendix. Initial debugging results indicate that when utilizing distance and time-related tool, the LLM (DeepSeek v2) demonstrates proficiency in task explanation, step-by-step planning, tool execution and summary. Even with limited information, the model provides matching recommendations.

However, the LLM's execution process is excessively detailed, exhaustively exploring every possible combination through tool calls. This leads to prolonged processing times, with the model invoking analysis even for seemingly obvious answers. This thoroughness, while comprehensive, presents challenges in terms of computational efficiency.

# 6 Conclusion

## 6.1 Summary of Findings

This study primarily focused on exploring the potential of Large Language Models (LLMs) in enhancing driver-passenger matching decisions within dynamic urban environments. While we have successfully tested and analyzed the performance of baseline models, including Random Assignment, Integer Linear Programming, Heuristic Algorithms, and Reinforcement Learning, our ultimate goal is to leverage LLMs to further improve the decision-making process. Among the baseline models, the RL model demonstrated the best performance in service rate. The baseline models provided a solid foundation and valuable insights for further research into LLM-enhanced strategies.

## 6.2 Limitations of the Study

While the focus on LLMs presents exciting possibilities, the study has certain limitations that should be acknowledged. Firstly, the research is still in the exploratory phase, with RL models being the primary method tested so far. The full integration and testing of LLMs in this context are ongoing. Furthermore, the study focused on a non-pooling ride-sharing context, and the findings may not fully generalize to scenarios where ride-pooling is allowed. Additionally, the reliance on simulated environments for testing may not fully capture the complexities of real-world conditions. Finally, the computational demands of LLMs, particularly in real-time applications, remain a challenge that needs to be addressed in future work.

## 6.3 Future Work

1. 对于决策过于详细而耗时的问题，我们或许可以在输入给大模型之前预先调用。

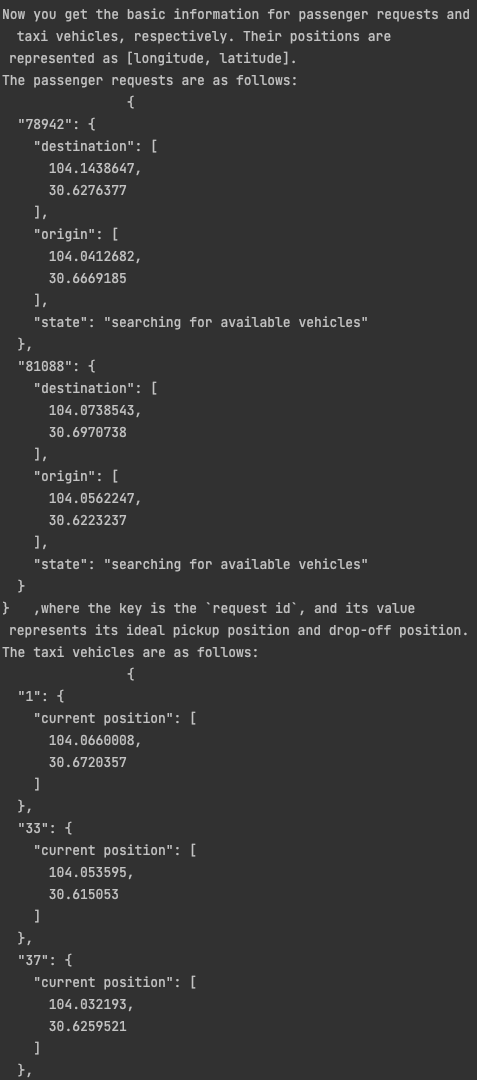
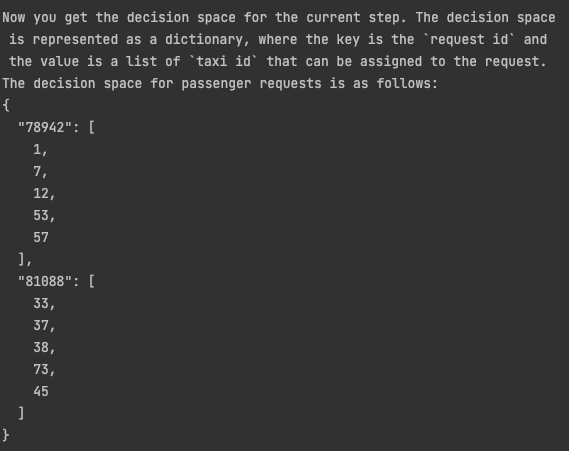
2. 目前大模型的决策依据 是直接从所有的 决策空间 遍历得来，或许我们可以 将这种较为简单的循环任务交给本地小模型做，而将复杂的任务规划及问题范例识别 交给更强大的商业大模型解决。

3. 我们的work-flow 方案表明我们还有很艰巨的任务去做，涉及到大量工程和prompt 调试

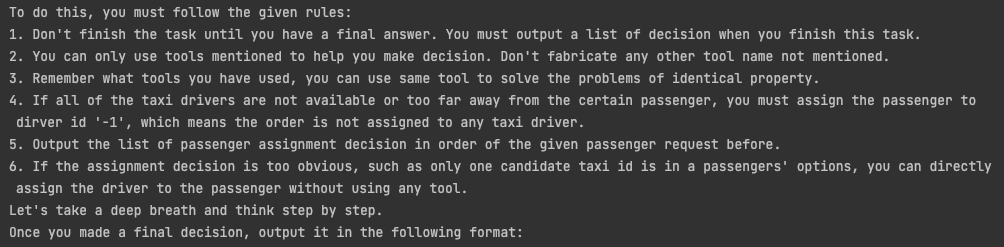
4. 为了提高执行效率，节约token 花销，我们想追求让小的LLM比肩大规模LLM。在计划中，我们让70B Llama 3.1产生 SFT数据，或者作为老师或评委， 进而微调8B Llama 3.1 ，让LLM掌握更高级的任务规划能力，记住工具指令的相关知识。具体的训练实验环境预计和Huawei HKRC 合作，使用Ascend 910B NPU。

# Appendx

## 7.1 prompt

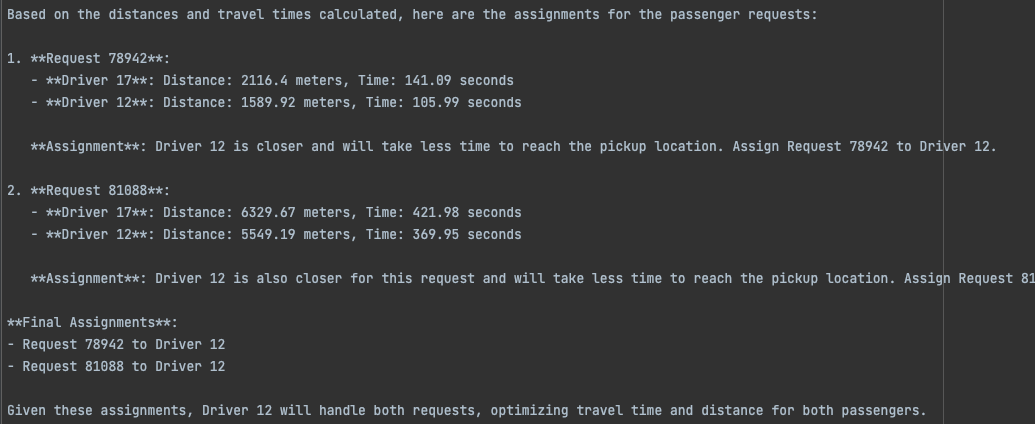
Input prompt for basic information (left) and decision space (right)



Query instruction for input，which highlight the considerations for LLM

## 7.2 Output from LLM

**Decision Output for version 1（white）**，only tested on small scale and LLM with only Get Travel Time and Distance between two IDs of Location as a single tool



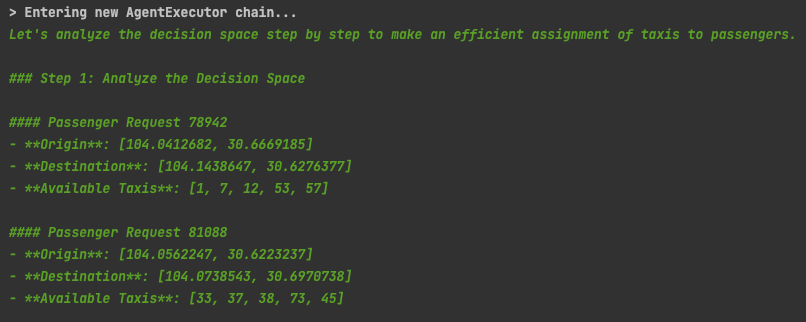
**Decision Output for version 2 （green）**formally developed on decision space in larger scale from simulator, combined with

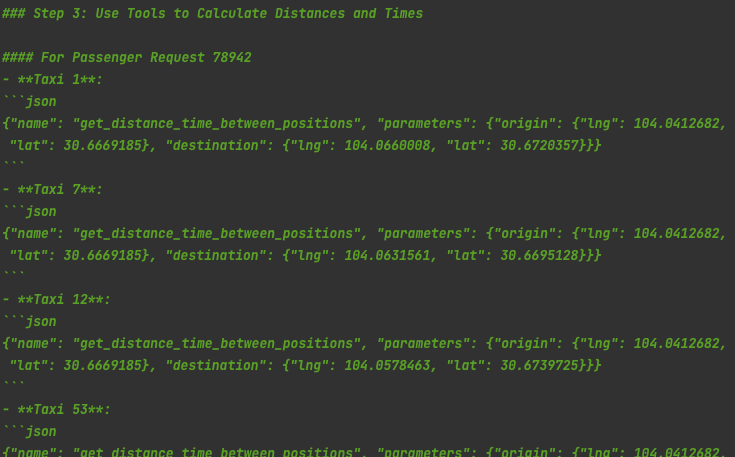
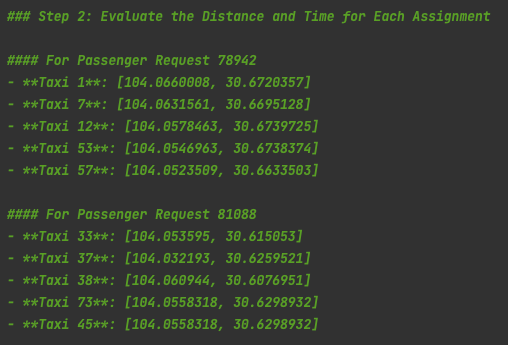
Tool 1: Get Travel Time and Distance between two Locations

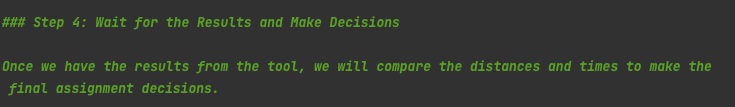
Tool 2: Get PoI by Location

The LLM tried to solve the batch of assignment step by step.

1. Analyze the question first and reformulate it in a more intelligible way
2. Plan to get the essential knowledge to make the decision
3. Generate tool calling instructions
4. Make final decision by combining the fetched results from tools.



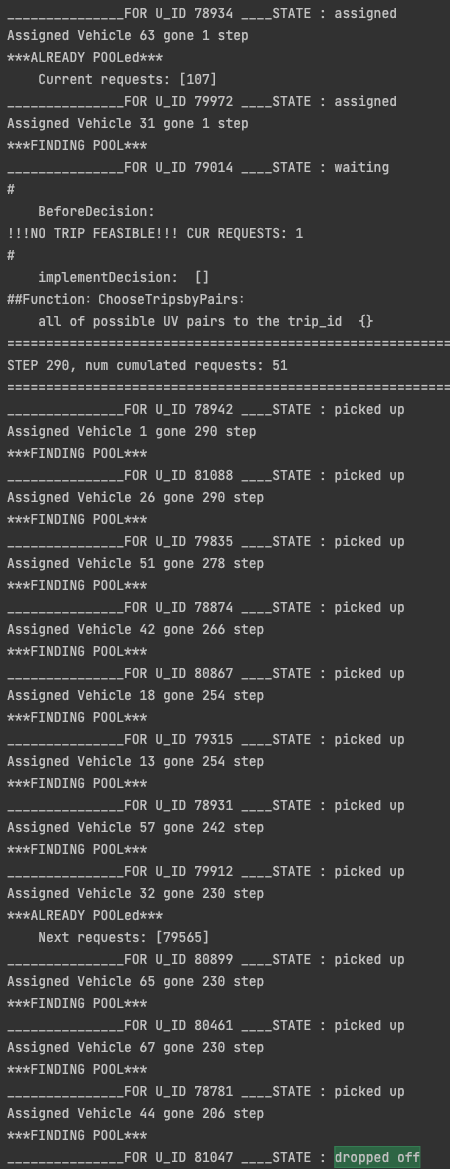




You can check the running log by LangSmith framework here :

<https://smith.langchain.com/public/916c6f69-1b94-4b6a-9df3-1c989e26dd3f/r>

## 7.2 simulator log



The log displays the passenger request state within two simulation step, in the middle epoch, many are picked up, most of vehicles are rode by only one passenger.

# 

# References