# Learning-Based Pricing and Service Planning for Shared Automated Vehicle Services in a Competitive Multi-Provider Market

MSc Thesis

by

Yu Zhang

Student number: 5989086

Thesis committee: Theresia van Essen, TU Delft
Qiaochu Fan, TU Delft
Joris Bierkens, TU Delft

An electronic version of this thesis is available at
https://repository.tudelft.nl/.

TU Delft Delft University of Technology

# Contents


**1 Introduction** **2**

**2 Literature review** **4**
2.1 Planning and operational decisions . . . . . 4
2.2 Learning-based approaches to service planning . . . . . 5

**3 Problem formulation** **7**
3.1 Assumptions . . . . . 7
3.2 Network representation . . . . . 10
3.3 Demand representation and choice modelling . . . . . 11
3.4 Service quality . . . . . 13
3.5 Passenger and vehicle flow modelling . . . . . 14
3.6 Objective function . . . . . 16

**4 Learning-based solution** **17**
4.1 Reinforcement learning setting . . . . . 19
4.2 Soft Actor-Critic algorithm . . . . . 20
4.3 Overall workflow . . . . . 27

**5 Diagonalisation method** **30**
5.1 Diagonalisation method . . . . . 31
5.2 Maximum waiting time . . . . . 32
5.3 Linearisation of the binary logit model . . . . . 33
5.4 Linearisation of the floor function . . . . . 36
5.5 Linearisation of the service rate constraint . . . . . 36
5.6 Linearisation of the objective function . . . . . 36

**6 Numerical experiments** **38**
6.1 Parameter settings . . . . . 38
6.2 Results . . . . . 40
6.2.1 Sensitivity analysis on reward scaling . . . . . 40
6.2.2 SAC results . . . . . 44
6.2.3 Sensitivity analysis on random seeds . . . . . 51
6.2.4 Diagonalisation results . . . . . 54
6.2.5 Comparison of the SAC and diagonalisation methods . . . 56

**7 Conclusion and future research** **59**

# 1 Introduction

With the development of autonomous driving technology and shared mobility systems, shared autonomous vehicles (SAVs) have become a promising mobility service for supporting sustainable urban transportation. By providing on-demand and shared travel services, SAV systems have the potential to reduce dependence on private cars, improve vehicle utilisation, and mitigate negative urban traffic impacts, including parking shortages, traffic congestion, air pollution, energy waste, and traffic accidents (Fan et al., 2023). However, in a competitive market with multiple providers, the successful operation of SAV services depends on appropriate service planning, including pricing, fleet sizing, and service quality control, as well as operational decisions such as passenger assignment and vehicle routing. Therefore, how to coordinate pricing and service decisions with operational decisions, and how to determine effective competitive strategies when the decisions of different providers affect one another, are important considerations for SAV operation.

Regarding service planning and operational decision-making, existing studies have developed optimisation models for fleet planning and operations in shared mobility services (Fan et al., 2023; Li et al., 2025; Feng and Mei, 2023; Liu et al., 2024). Collectively, these models have examined fleet sizing, endogenous demand, passenger choice, request acceptance, pricing, and operational decisions concerning vehicle assignment, routing, parking, and relocation. In addition, some studies have considered competition between multiple mobility service providers (Najmi et al., 2023; Li et al., 2025; Yang et al., 2022; Sun et al., 2024; Chen et al., 2024). These studies show that provider decisions influence passenger demand, market share, and profit. However, studies that include waiting time and service quality mainly focus on markets with a single provider or treat these factors as affecting operations rather than as decisions made by service providers (Bruglieri et al., 2024; Lai and Li, 2023; Chen et al., 2024). Therefore, further investigation may provide additional insights into the effects of maximum waiting time and service rate on demand allocation and provider profits in markets with multiple competing providers.

Regarding the determination of effective competitive strategies, optimisation-based methods can be used to identify possible competitive outcomes. One approach is the diagonalisation method, which sequentially solves each provider's best-response problem while treating the latest decisions of its competitors as fixed inputs (Yang et al., 2022). However, the decisions of different providers jointly affect market demand and operational outcomes. As each provider responds directly to the latest decisions of its competitors, the diagonalisation method may produce oscillating best response updates and therefore serves as a heuristic for identifying a locally stable solution without guaranteeing convergence to a globally stable outcome. This motivates the use of a learning-based approach (Sun et al., 2024; Chen et al., 2024). A learning-based approach can learn providers' pricing and service decisions through repeated interaction with the market environment. By updating policies gradually during training and using entropy regularisation to encourage broader exploration of the continu-

ous strategy space, it may reduce oscillations caused by repeated best-response updates and explore solutions beyond the local stable points.

Following Fan et al. (2023), this thesis extends the SAV fleet sizing and management problem to a competitive market with multiple SAV providers and proposes a framework based on reinforcement learning. In this framework, each SAV provider is represented by an agent. Each agent selects its base fare, distance-based fare, maximum waiting time, and service rate. The passenger choice model then uses the joint decisions of all agents to compute passenger utility and provider choice probabilities, and then allocates demand among the providers. Once demand has been allocated among the providers, an integer linear programming model is solved separately for each provider to optimise fleet size, passenger assignment, and vehicle routing. The resulting profit is returned to the corresponding agent as its reward. Through repeated interaction with the environment, the agents learn pricing and service strategies that seek to improve their profits under competition.

The main contributions of this thesis are summarised as follows:

- Fare, maximum waiting time, and service rate are modelled as pricing and service decisions learned by each SAV provider during the reinforcement learning training process. Fleet size is optimised in the operational optimisation model together with passenger assignment and vehicle routing.
- A reinforcement-learning-based framework is proposed for competitive SAV service planning. In this framework, SAV providers learn pricing and service planning strategies through repeated interaction with the competitive market environment.
- A market feedback environment is developed by combining a passenger choice model with integer linear programming models. Given the strategic decisions of all providers, the environment computes demand, determines fleet sizing, assigns passenger and vehicle flows, and returns each provider's profit as the reward signal.

# 2 Literature review

Shared mobility service planning involves interactions among service provider decisions, traveller demand responses, and fleet operational constraints. To provide a structured overview of the relevant research, this section presents the literature review in two parts. The first part reviews studies on shared mobility planning and operations in both single-provider and multi-provider market settings and summarises the main decision variables considered in existing research. The second part reviews learning-based optimisation methods, especially reinforcement learning methods, for solving pricing and service planning problems in competitive mobility markets.

## 2.1 Planning and operational decisions

Existing studies on shared mobility operations generally consider two types of decisions: planning decisions and operational decisions. Planning decisions include fleet sizing, initial vehicle allocation, pricing, infrastructure deployment, waiting time and service rate, whereas operational decisions include vehicle assignment, routing, relocation, parking, charging, and vehicle–passenger matching.

Regarding planning decisions, Li et al. (2025) optimise the fleet size of business-owned SAVs and the rental price of SAV services in a hybrid-ownership SAV system. Li et al. (2025) develop a duopoly SAV market model in which two operators choose their respective fleet sizes and service prices to maximise their operating profits. Sun et al. (2024) formulate service pricing and passenger demand in a multi-provider Mobility-as-a-Service (MaaS) market as a multi-leader–multi-follower Stackelberg game, in which service providers set service prices and passengers determine their service demand accordingly. Chen et al. (2024) investigate competitive spatiotemporal pricing between an SAV platform and a human-driven ride-sourcing platform. The SAV platform determines service prices across different regions and time periods, while the human-driven platform determines both service prices and driver wages. Passenger waiting affects request cancellation in the simulation but is not treated as a platform decision variable. Regarding operational decisions, Feng and Mei (2023) jointly optimise SAV routing and parking to minimise vehicle kilometres travelled, the number of SAVs, and parking costs. Some studies jointly address multiple types of decisions. Fan et al. (2023) jointly optimise fleet size, initial fleet distribution, and service rate together with trip assignment, vehicle routing, parking, and relocation in a profit-maximisation model. Chen and Liu (2023) integrate charging facility deployment and initial vehicle allocation with vehicle assignment, relocation, and charging decisions to maximise expected net profit. Lai and Li (2023) formulate a profit-maximisation model for an autonomous mobility-on-demand (AMoD) platform that jointly determines spatiotemporal prices and idle-vehicle rebalancing strategies. The model incorporates waiting time as a determinant of passenger demand but does not treat maximum waiting time as a provider decision variable. Yang et al. (2022) jointly optimise the pricing and vehicle re-

location strategies of competing companies in a car-sharing market, where each company seeks to maximise its profit and traveller demand is assigned through a stochastic user equilibrium model.

Overall, existing studies have identified fleet sizing, pricing, and service design as important components of shared mobility operations. However, provider-level service decisions in competitive multi-provider SAV markets appear to have received relatively limited attention. In particular, how competing providers determine their respective maximum waiting time and service rates, and how these decisions may affect passenger demand allocation and provider profits, have not yet been fully explored. Therefore, this thesis jointly considers fare, maximum waiting time, and service rate as provider-level decisions in a competitive multi-provider SAV market.

## 2.2 Learning-based approaches to service planning

Learning-based approaches have been increasingly used for pricing and operational decision-making in complex mobility systems. These approaches are useful when the relationship between service decisions, passenger demand, and operational profit is difficult to express analytically. They also allow providers to improve their strategies through repeated interactions with a market feedback environment.

Reinforcement learning has also been used for dynamic pricing and fleet management problems. In reinforcement learning, an agent learns a policy by interacting with the environment and receiving rewards from its actions. The methods can be broadly divided into critic-only methods, actor-only methods, and actor-critic methods (Grondman et al., 2012). Critic-only methods, such as Q-learning (Watkins and Dayan, 1992), learn a value function to evaluate actions. Actor-only methods, such as REINFORCE (Williams, 1992), directly update the policy parameters. Actor-critic methods combine these two ideas by using a critic to evaluate the policy and an actor to update the action-selection rule.

Turan et al. (2020) study dynamic pricing and fleet management for an electric AMoD system. Their model considers a profit-maximising platform that jointly decides ride prices, vehicle routing, and vehicle charging. The platform adjusts ride prices to influence passenger demand, while routing decisions include both serving passengers and relocating empty vehicles. Since the real-time problem has high-dimensional and continuous state and action spaces, the authors use Proximal Policy optimisation (PPO) to learn a near-optimal control policy. The study is conducted for a single platform, so competition between multiple providers is not considered.

Sun et al. (2024) study vehicle dispatching service pricing in a MaaS market with multiple competing mobility service providers. They formulate the interaction between providers and passengers as a two-stage Stackelberg game and use a multi-agent deep reinforcement learning framework to solve the pricing problem. The main decision variable is the unit-distance price set by each provider. In the first stage, Multi-Agent Deep Q-Network (MADQN) is used to learn the

pricing strategy of each provider. In the second stage, the Stackelberg game is used to determine passenger service demand and select the provider that serves the request. Their results show that the MADQN-based pricing strategy can converge in a competitive multi-agent setting and can improve provider profit compared with benchmark pricing methods.

Chen et al. (2024) study a duopoly ride-sourcing market with one SAV platform and one human-driven vehicle platform. The study focuses on competitive pricing, where the SAV platform decides the regional ride prices and the human-driven vehicle platform decides both regional ride prices and the driver payout ratio. These decisions are learned in a multi-agent simulation environment that considers passenger choice, vehicle matching, routing, cruising, parking, and passenger waiting.

Lu et al. (2021) propose a deep reinforcement learning-based charging pricing framework for an AMoD system. The problem is formulated as a non-cooperative Stackelberg game, where multiple charging station operators act as leaders and set charging prices, while the AMoD operator acts as the follower and determines vehicle assignment, vehicle repositioning, and charging scheduling decisions. To solve the pricing problem, the authors develop a multi-agent Soft Actor-Critic (SAC) algorithm, where each charging station operator is treated as an agent. The action of each agent is the charging price of its charging stations, and the reward is the corresponding charging profit. Their results show that SAC achieves higher returns and faster convergence than PPO and Deep Deterministic Policy Gradient (DDPG) in the multi-agent pricing setting.

These studies demonstrate the potential of learning-based methods for service planning problems in which the relationships among provider decisions, passenger demand, and profits are difficult to characterise analytically. Although several studies consider competition among multiple agents, they mainly use learning-based methods to determine pricing strategies rather than to jointly determine pricing and service planning decisions. Thus, this thesis applies a multi-agent learning framework in which each competing provider jointly determines its own pricing and service planning decisions.

# 3 Problem formulation

This section first introduces the main assumptions regarding the market setting, traveller behaviour, and SAV operations, and then presents the fleet planning and operational model for an individual provider.

## 3.1 Assumptions

In this thesis, we consider a future scenario in which only SAV services and active modes of transport, such as bicycles, are available for urban travel. Travellers can request transportation services from any location in a city using SAV service applications on their smartphones by providing trip information. Upon receiving the trip information, the provider decides whether to accept or reject the trip request.

Before formally introducing the model, we present the main assumptions. Assumptions (a)–(e) follow Fan et al. (2023).

(a) The proposed model serves a strategic planning purpose. During the study period, the total mobility demand in the urban area is assumed to be fixed and known in advance, enabling the SAV operator to make optimal planning decisions. This setting differs from a real-time SAV operating system, where future demand is uncertain and operational decisions must be continuously updated when new requests arrive.

(b) We assume that travellers have perfect information about the transportation network status and will make rational choices among different SAV providers based on their perceived travel time.

(c) We assume that the SAVs considered in this study operate at SAE Level 5 and can travel throughout the entire network without a human driver.

(d) Privately-owned vehicles and human-driven vehicles are not considered as travel options in this study.

(e) Pooled services are not considered in this study.

(f) SAV fares, maximum waiting time, and service rate are treated as exogenously given service settings in the fleet planning and operational model. In the reinforcement learning part, these settings will be treated as decision variables or actions for each learning agent.

(g) Congestion effects are not considered in this study.

(h) Bicycles are considered an alternative travel mode. The availability of bicycles as an alternative allows passengers to exit the SAV market when high fares, long waiting times, or poor service quality make SAV services less attractive, thereby discouraging SAV providers from excessively raising fares to increase profits.

To provide a comprehensive understanding of the proposed model, Fig. 1 illustrates the connection between the learning agent of SAV provider $p$ and the operational environment. It shows the decision variables selected by the agent, the input information used by the environment, and the output information returned to the agent. In the figure, the network setting includes nodes, road links, time periods, time-space links, parking nodes, link lengths, and road link

travel time. The unitary operating costs include the unit driving operating cost per kilometre and the depreciation cost per vehicle.

More specifically, each provider has its own fleet planning and operational model, and the decisions of other providers are included as competitors' decisions. At each decision round, the learning agent of provider $p$ selects an action, including fare, service rate, and maximum waiting time. Then, these action values are fed to the fleet planning and operational model, as the fare and service settings of provider $p$.

Given these settings, together with the network setting, demand information, competitors' decisions, travellers' income profiles, and operating settings, the fleet planning and operational model is solved to determine the fleet size and the corresponding profit, which is then returned to the reinforcement learning framework as the reward signal.

Additionally, the main mathematical notations are summarised in Table 1 for easy reference. In the table, the decisions generated by the reinforcement learning agent, such as fares, maximum waiting time, and service rate are treated as input parameters from the perspective of the fleet planning and operational model.

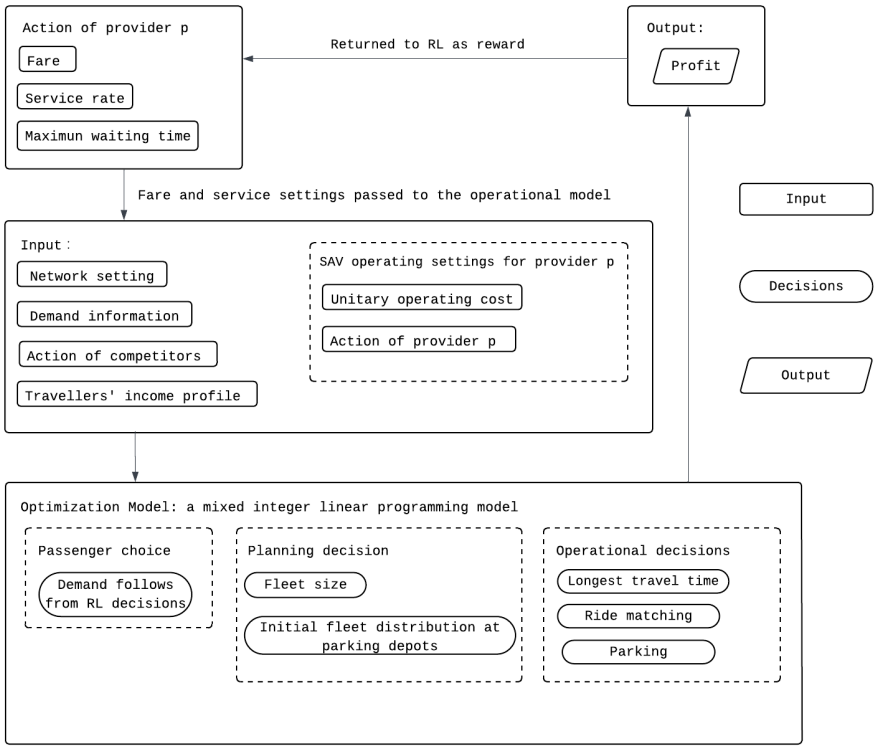



Figure 1: Structure of the learning-based model.

Table 1: Notation.

| Notation | Description |
|---|---|
| **Sets** | |
| $T$ | $\{0, 1, 2, \ldots, \mathcal{T}\}$. Set of time instants in the operation period. |
| $N$ | Set of nodes. |
| $L$ | Set of road links between nodes in set $N$. |
| $G$ | Set of links in the time-space network. |
| $P$ | $P = \{1, 2, \ldots, \lvert P \rvert\}$. Set of SAV providers. |
| $N_A^p$ | Set of nodes allowing parking for provider $p$, with $N_A^p \subseteq N$. |
| $R$ | Set of groups of trips, where each group of trips $r \in R$ has the same origin, destination and preferred departure time. |
| **Choice model** | |
| *Parameters* | |
| $OM_p^r$ | Monetary cost of travellers in group $r \in R$ when using SAV provider $p \in P$, in euros. |
| $OM_B^r$ | Monetary cost for travellers in group $r \in R$ using bike, in euros. |
| $\beta_0$ | Parameter converting monetary cost into utility, in utility/euro. |
| $\beta_1$ | Parameter converting service rate into utility. |
| $VOT^r$ | Travellers' value of in-vehicle travel time in group $r \in R$, in euros/time step. |
| $VOT_B^r$ | Travellers' value of bicycle travel time in group $r \in R$, in euros/time step. |
| $\bar{w}_p$ | Maximum waiting time allowed for each served trip by SAV provider $p \in P$, in time steps. |
| $st^r$ | Shortest travel time assuming free-flow speed for group of trips $r \in R$, in time steps. |
| $\alpha_p$ | Service rate of SAV provider $p \in P$, $\alpha_p \in [0, 1]$. |
| $n^r$ | Total number of trips in group $r \in R$. |
| $T_p^r$ | Perceived travel time for travellers in group $r \in R$ when using SAV provider $p \in P$. |
| $V_p^r$ | Deterministic systematic component of travellers' utility for choosing SAV provider $p \in P$ in group $r \in R$. |
| $T_B^r$ | Perceived travel time for travellers in group $r \in R$ when using bike. |
| $V_B^r$ | Deterministic systematic utility of travellers in group $r \in R$ when choosing the bike alternative. |
| $\mathcal{P}_p^r$ | Probability to choose SAV provider $p \in P$ for the trips in group $r \in R$. |
| $D_B^r$ | Total number of trips in group $r \in R$ using bike. |

| Notation | Description |
|---|---|
| $D_p^r$ | Total number of trips in group $r \in R$ using SAV provider $p \in P$. |
| **Fleet planning and operational model** | |
| *Parameters* | |
| $\Delta t$ | Time step. |
| $l_{ij}$ | Length of road link $(i,j) \in L$, in kilometres. |
| $o^r$ | Origin node for group of trips $r \in R$. |
| $d^r$ | Destination node for group of trips $r \in R$. |
| $a^r$ | Trip request time of group $r \in R$. |
| $sd^r$ | Shortest travel distance for group of trips $r \in R$, in kilometres. |
| $f_p^0$ | Initial base fare for using SAV provider $p \in P$, in euros/trip. |
| $f_p$ | Travel distance-based fare for using SAV provider $p \in P$, in euros/km. |
| $co$ | Unit driving operational cost of an SAV, in euros/km. |
| $cf$ | Depreciation cost for using an SAV, in euros/vehicle. |
| *Decision variables* | |
| $S_p^r$ | Total number of trips served by SAV provider $p \in P$ from group $r \in R$. |
| $PF_{i_{t_1} j_{t_2}}^{r,p}$ | Passenger flow of travellers in group $r \in R$ served by SAV provider $p \in P$ on road link $(i,j)$, from time instant $t_1$ to $t_2$. |
| *Auxiliary variables* | |
| $M_p$ | Fleet size of SAV provider $p \in P$. |
| $M_i^p$ | Initial distribution of SAVs of provider $p \in P$ at parking node $i \in N_A^p$ at the beginning of the operation period. |
| $E_t^{r,p}$ | Total number of passengers in group of trips $r \in R$ served by provider $p \in P$ and arriving at time $t \in T$. |
| $F_{i_{t_1} j_{t_2}}^p$ | Vehicle flow of SAV provider $p \in P$ on road link $(i,j)$, from time instant $t_1$ to $t_2$, where $(i_{t_1}, j_{t_2}) \in G$. When $t_1 = 0$, $i \in N_A^p$, meaning that SAVs depart from parking nodes at the beginning of the operation period. |
| $W_{i_t}^p$ | Number of vehicles of SAV provider $p \in P$ parked at node $i \in N_A^p$ from time instant $t$ to $t+1$, with $t \in T$. |

## 3.2 Network representation

To describe vehicle movements and operational decisions, we use a time-space network. A time-space network is a time-expanded version of the directed physical network $(N, L)$, where $N$ and $L$ denote the set of physical nodes and road links, respectively. The time dimension is discretised into $\mathcal{T}$ periods, with each period having a duration $\Delta t$, referred to as the time step, hereinafter.

Consequently, an index set of time periods $T = \{0, 1, 2, \ldots, \mathcal{T}\}$ is defined within the study horizon. At each time instant $t \in T$, the physical network is replicated, thus multiple networks are defined along with the time period, as shown in Fig. 2. We use $G$ to denote the set of links in the resulting time-space network.

Different from the traditional physical network, the status of vehicles in the time-space network is described by both the vehicle movement and the time required for that movement. A time-space link $(i_{t_1}, j_{t_2}) \in G$ represents a movement from node $i \in N$ at time instant $t_1 \in T$ to node $j \in N$ at time instant $t_2 \in T$ or park at node $i \in N_A^p$ from time instant $t$ to $t+1$. Let $P$ denote the set of SAV providers, then, $N_A^p \subseteq N$ denotes the set of nodes allowing parking for provider $p \in P$.

Several physical attributes related to the road links are defined. The length of road link $(i, j) \in L$ is denoted by $l_{ij}$, and the fixed travel time of link $(i, j) \in L$ is denoted by $t_{ij}$.

Fig. 2 provides an illustrative example of vehicle movements in a time-space network. Passenger-carrying movements are represented by solid lines, relocation movements without passengers are represented by dashed lines, and parking states are represented by dotted lines. In this small physical network, a uniform travel time of one time step is assumed for each road link.

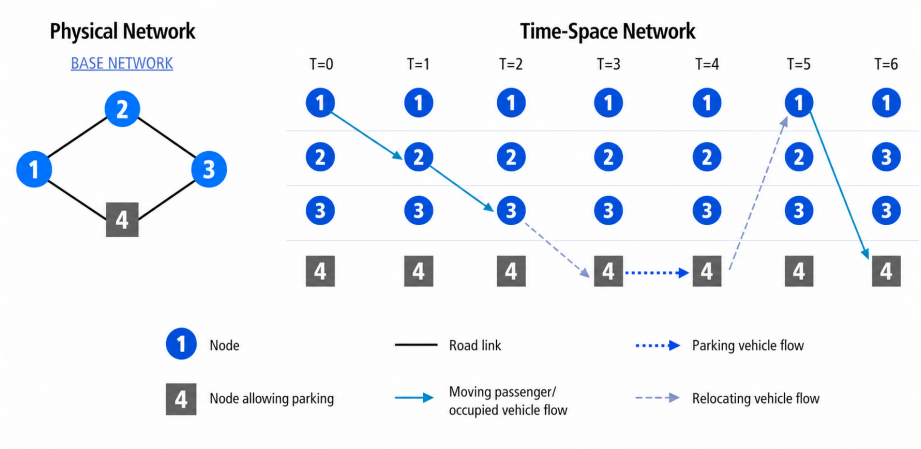



Figure 2: Illustration of the physical network, the time-space network, and vehicle movements over the time-space network.

## 3.3 Demand representation and choice modelling

Instead of tracking each trip individually, demand that shares the same travel information and income level is aggregated into a group. We introduce set $R$ as the set of groups of trips, where each group of trips $r \in R$ is characterised by the same origin $o^r \in N$, destination $d^r \in N$, trip request time $a^r \in T$, shortest travel distance $sd^r$ in kilometres, shortest travel time $st^r$, and income level. For

each SAV provider $p \in P$, the pickup time of group $r$ must lie between $a^r$ and $a^r + \bar{w}_p$, where $\bar{w}_p$ denotes the maximum waiting time allowed by provider $p$. Passengers in each travel group can choose among different providers. Let $n^r$ denote the total number of trips in travel group $r \in R$.

The SAV provider choices for travellers in each group are analysed using discrete choice modelling within the framework of random utility maximisation theory. To measure the probability of selecting a specific SAV provider, the utility of each provider is calculated. Moreover, three income levels for travellers are considered: high income, middle income, and low income. The income level determines the value of travel time (VOTT) of travellers, which directly affects their perceived utility for each SAV provider and consequently influences their provider choice. The VOTT for travellers in the same group $r \in R$ is assumed to be the same, denoted by $VOT^r$ in euros per time step. These income classes are not explicitly defined in the model but are implicitly included in the travel information of each group of trips.

However, the utility is not known with certainty due to factors such as unobserved heterogeneity among travellers, unobserved attributes of the alternatives, and perception errors (Ben-Akiva and Lerman, 1985). Therefore, the utility of SAV provider $p \in P$ for trips in group $r \in R$, denoted by $U_p^r$, is treated as a random variable. It consists of a deterministic systematic component $V_p^r$, which is the observable part of the utility, and a random component $\varepsilon_p^r$, which captures the unobservable effects:

$$U_p^r = V_p^r + \varepsilon_p^r, \quad \forall r \in R,\ p \in P. \tag{1}$$

The deterministic utility $V_p^r$ of choosing an SAV provider for group $r \in R$ depends on the generalised cost of the trip and the travellers' satisfaction with the provider's service. Specifically, the generalised cost is calculated as the sum of the fare $OM_p^r$ and the travel-time-related cost $VOT^r T_p^r$. The fare depends on the provider's pricing strategy and the travel distance. The perceived travel time $T_p^r$ consists of the shortest travel time $st^r$ and the maximum waiting time $\bar{w}_p$. The shortest travel time $st^r$ is fixed for each group of trips and is assumed to be the same for all SAV providers. Thus, the perceived travel time $T_p^r$ is given by:

$$T_p^r = st^r + \bar{w}_p, \quad \forall r \in R,\ p \in P. \tag{2}$$

Parameter $\beta_0$ converts generalised costs into utility (in utility/euro), reflecting the sensitivity of travellers to changes in monetary cost. Travellers' satisfaction is modelled via the trip service rate $\alpha_p$. The parameter $\beta_1$ captures the disutility caused by unsatisfied demand. Thus, the deterministic utility $V_p^r$ is given by:

$$V_p^r = -\beta_0 \left(OM_p^r + VOT^r T_p^r\right) - \beta_1 \left(1 - \alpha_p\right), \qquad \forall r \in R,\ p \in P. \tag{3}$$

In Eq. (1), $\varepsilon_p^r$ represents the unobserved utility component (i.e., the error term) between the actual and deterministic utility of selecting SAV provider

$p \in P$ for trips in group $r \in R$. This component captures the variation unknown to the analyst.

Assuming that these error terms are independently and identically Gumbel distributed, the probability that group of trips $r \in R$ chooses SAV provider $p \in P$, denoted by the continuous variable $\mathcal{P}_p^r$, is given by the multinomial logit model:

$$\mathcal{P}_p^r = \frac{\exp\left(V_p^r\right)}{\sum_{j \in P} \exp\left(V_j^r\right) + \exp\left(V_B^r\right)}, \qquad \forall r \in R,\ p \in P. \tag{4}$$

where $V_p^r$ is the deterministic utility of choosing SAV provider $p \in P$ for group of trips $r \in R$. The deterministic term $V_B^r$ of the utility for using a bicycle for group $r \in R$ is calculated based on the monetary cost of using bicycles, as shown in Eq. (5).

$$V_B^r = -\beta_0 \left(OM_B^r + VOT_B^r T_B^r\right), \quad \forall r \in R. \tag{5}$$

The monetary cost $OM_B^r$ for group $r \in R$ is the bicycle's depreciation cost, which is calculated by dividing the bicycle's purchase price by its service life. $VOT_B^r$ represents the passengers' value of travel time for group $r \in R$ when using the bike. We assume that the bike travel time $T_B^r$ for group $r \in R$ is constant.

We use integer variables $D_p^r$ to represent the demand assigned to SAV provider $p \in P$ from group $r \in R$. The demand for provider $p \in P$ is computed by multiplying the total number of trips $n^r$ in group $r \in R$ by the probability of choosing provider $p \in P$. Then, we round this value to the nearest integer using a floor function as shown in Eqs. (6).

$$D_p^r = \left\lfloor n^r \mathcal{P}_p^r + 0.5 \right\rfloor, \qquad \forall r \in R,\ p \in P. \tag{6}$$

To avoid creating additional demand after rounding, Constraints (7) ensures that the total demand allocated to all SAV providers and the bicycle alternative equals the original demand of group of trips $r \in R$.

$$\sum_{p \in P} D_p^r + D_{\mathrm{B}}^r = n^r, \qquad \forall r \in R. \tag{7}$$

Eqs.(1) - (6) and Constraints (7) describe the demand calculation and multinomial logit choice model, which is calculated before the fleet planning and operational optimisation model. Therefore, the demand values $D_p^r$ are computed beforehand and are passed to the optimisation model as input parameters.

### 3.4 Service quality

Since congestion is not considered, the vehicle flows generated by one provider do not affect the travel times of the other providers. Consequently, they do not affect passengers' utilities or demand for the other providers. Once demand is determined, fleet planning and operational optimisation can be performed independently for each provider. The following formulation presents the model for a given provider $p$.

For each group $r \in R$, the total number of trips served by SAV provider $p$ is specified by integer variables $S_p^r$. In this study, the service rate $\alpha_p$ is determined outside the optimisation model and is treated as an exogenous input. The relationship between the total number of served trips, the total demand and the service rate can be described by Constraint (8).

$$\alpha_p \sum_{r \in R} D_p^r \leq \sum_{r \in R} S_p^r \tag{8}$$

In addition, the number of served trips $S_p^r$ for each group of trips $r \in R$ should not exceed the demand $D_p^r$ assigned to provider $p$.

$$S_p^r \leq D_p^r, \quad \forall r \in R \tag{9}$$

### 3.5 Passenger and vehicle flow modelling

In this section, we develop the base formulation for SAV provider $p$ to manage its SAV fleet. Two tiers of decisions are made in this model: (1) at the strategic level, given the service rate, the model determines the fleet size used by provider $p$ and the initial fleet distribution at the beginning of the operation period; and (2) at the operational level, the model determines the assignment of passengers, parking, and relocation decisions.

We introduce integer variables $PF_{i_{t_1} j_{t_2}}^{r,p}$ to represent the passenger flow in group of trips $r \in R$ served by SAV provider $p$ on road link $(i, j)$, from time instant $t_1$ to $t_2$. The variables $PF_{i_{t_1} j_{t_2}}^{r,p}$ are only defined for $(i_{t_1}, j_{t_2}) \in G$ with $t_1 < t_2$. Constraints (10) ensure that passengers in group of trips $r \in R$ are picked up by the SAVs at the origin node $o^r$ at pickup time $\tau \in T$, with $a^r \leq \tau \leq a^r + \bar{w}_p$, where $a^r$ denotes the request time of group of trips $r \in R$.

$$S_p^r = \sum_{\tau \in T | a^r \leq \tau \leq a^r + \bar{w}_p} \sum_{j_t | (o_\tau^r, j_t) \in G} PF_{o_\tau^r j_t}^{r,p}, \quad \forall r \in R. \tag{10}$$

Since passengers are allowed to wait before pickup and each provider has its own maximum waiting time $\bar{w}_p$, passengers in the same group may arrive at destination $d^r$ at different times, but no later than the provider-specified latest arrival time $a^r + \bar{w}_p + st^r$. The integer variables $E_t^{r,p}$ represent the total number of passengers in group of trips $r \in R$ served by provider $p$ and arriving at destination $d^r$ at time $t \in T$ with $a^r + st^r \leq t \leq a^r + \bar{w}_p + st^r$. Constraints (11) and (12) ensure that the number of trips in group $r \in R$ served by provider $p$ equals the total number of trips arriving at destination $d^r$.

$$S_p^r = \sum_{t \in T | a^r + st^r \leq t \leq a^r + \bar{w}_p + st^r} E_t^{r,p}, \qquad \forall r \in R, \tag{11}$$

$$E_t^{r,p} = \sum_{i_{t_1} | (i_{t_1}, d_t^r) \in G} PF_{i_{t_1} d_t^r}^{r,p}, \qquad \forall r \in R, \quad a^r + st^r \leq t \leq a^r + \bar{w}_p + st^r. \tag{12}$$

Constraints (13) and (14) ensure that each group of trips $r \in R$ visits its destination $d^r$ and origin $o^r$ only once:

$$\sum_{j_{t_1}|(d_t^r, j_{t_1})\in G} PF^{r,p}_{d_t^r j_{t_1}} = 0, \quad \forall r \in R,\ a^r + st^r \le t \le a^r + \bar{w}_p + st^r. \tag{13}$$

$$\sum_{i_{t_1}|(i_{t_1}, o_t^r)\in G} PF^{r,p}_{i_{t_1} o_t^r} = 0, \quad \forall r \in R,\ a^r \le t \le a^r + \bar{w}_p + st^r. \tag{14}$$

Constraints (15) guarantee passenger flow conservation at each node $i \in N \setminus \{o^r, d^r\}$ for each group of trips $r \in R$ and provider $p$.

$$\sum_{j_{t_1}|(j_{t_1}, i_t)\in G} PF^{r,p}_{j_{t_1} i_t} = \sum_{j_{t_2}|(i_t, j_{t_2})\in G} PF^{r,p}_{i_t j_{t_2}}, \quad \forall r \in R,\ a^r < t < a^r + \bar{w}_p + st^r,\ i \in N \setminus \{o^r, d^r\} \tag{15}$$

The SAVs operated by provider $p$ are in one of three statuses:(1) transporting a passenger; (2) driving empty to pick up the next passenger or driving to a parking depot; (3) being parked at a depot. We introduce integer variables $F^p_{i_{t_1} j_{t_2}}$ to denote vehicle flow over link $(i, j)$ from $t_1$ to $t_2$, where $(i_{t_1}, j_{t_2}) \in G$. Provider $p$ controls its own set of parking nodes $N^p_A$. If $t_1 = 0$, only links from parking nodes $i \in N^p_A$ are defined, meaning that SAVs have to depart from the parking nodes at the beginning of the day. Integer variables $W^p_{i_t}$ represent the number of SAVs of provider $p$ parked at node $i \in N^p_A$ between time $t$ and $t+1$, with $t \in T$.

No matter in which status, the vehicle flow $F^p_{i_{t_1} j_{t_2}}$ of provider $p$ on each road link should always be greater than or equal to the total passenger flow $\sum_{r\in R} PF^{r,p}_{i_{t_1} j_{t_2}}$ on that link to satisfy the mobility need. This is ensured by Constraints (16):

$$\sum_{r\in R} PF^{r,p}_{i_{t_1} j_{t_2}} \le F^p_{i_{t_1} j_{t_2}} \quad \forall (i_{t_1}, j_{t_2}) \in G. \tag{16}$$

The following constraints describe the flow conservation rules for the circulation of SAVs operated by provider $p$ at both normal and parking nodes.

$$\sum_{j_{t_1}|(j_{t_1}, i_t)\in G, t_1<t} F^p_{j_{t_1} i_t} = \sum_{j_{t_2}|(i_t, j_{t_2})\in G, t<t_2} F^p_{i_t j_{t_2}}, \quad \forall i \in N \setminus N^p_A, 0 < t < \mathcal{T} \tag{17}$$

$$\sum_{j_{t_1}|(j_{t_1}, i_t)\in G,\ t_1<t} F^p_{j_{t_1} i_t} + W^p_{i_{t-1}} = \sum_{j_{t_2}|(i_t, j_{t_2})\in G,\ t<t_2} F^p_{i_t j_{t_2}} + W^p_{i_t}, \quad \forall\, i \in N^p_A,\ 0 < t < \mathcal{T} \tag{18}$$

The initial distribution of SAVs operated by provider $p$ at parking node $i \in N_A^p$ at the beginning of a day is denoted by integer variables $M_i^p$. At the beginning of the optimisation period, SAVs of provider $p$ either depart from the parking depots to pick up passengers or park at the parking node waiting for dispatch instructions from the SAV provider, as described in Constraints (19). Given that both $F_{i_0,j_t}^p$ and $M_i^p$ are integer variables, the parking variable $W_{i_0}^p$ will also take integer values.

$$\sum_{j_t|(i_0,j_t)\in G} F_{i_0,j_t}^p + W_{i_0}^p = M_i^p, \forall i \in N_A^p \tag{19}$$

In addition, the total fleet size of provider $p$, denoted by the integer variable $M_p$, is equal to the sum of the initial fleet distributed at the parking nodes, as shown in Constraint (20).

$$\sum_{i\in N_A^p} M_i^p = M_p \tag{20}$$

### 3.6 Objective function

To maximise the total profit of SAV provider $p$, the profit function defined in Eq. (21) comprises the total fare revenue collected from service users and the total costs incurred in operating the SAV fleet. Service users pay two types of fares for using an SAV: a fixed base fare $f_p^0$, measured in euros per trip, and a distance-related fare $f_p$, measured in euros per kilometre. For each trip in group $r \in R$, the distance-related fare is calculated based on the shortest travel distance $sd^r$.

The total costs consist of two components: (1) the total depreciation costs of the SAV fleet in the system and (2) the total operational costs. The total depreciation cost is calculated by multiplying the unit depreciation cost $cf$, measured in euros per vehicle, by the total fleet size. The unit depreciation cost $cf$ is calculated by dividing the vehicle's purchase price by its service life span. The total operational costs include fuel, maintenance and insurance costs, which are calculated as the unit operational cost $co$, measured in euros per kilometre, multiplied by the total distance travelled by all SAVs. The total travel distance includes both the distance travelled while carrying passengers and the relocation distance travelled without passengers.

$$\pi_p = \sum_{r\in R} OM_p^r \cdot S_p^r - cf \cdot M_p - co \cdot \sum_{(i_{t_1},j_{t_2})\in G} l_{ij} \cdot F_{i_{t_1}j_{t_2}}^p \tag{21}$$

where

$$OM_p^r = f_p^0 + sd^r \cdot f_p, \quad \forall r \in R. \tag{22}$$

The objective of provider $p$ is then to maximise its own profit:

$$\max\ \pi_p. \tag{23}$$

# 4 Learning-based solution

This section presents a learning framework based on SAC, in which SAV providers learn pricing and service planning strategies that seek to improve their profits under competition. The fleet planning and operational model introduced in Section 3 is used as the learning environment. The rest of this section is organised as follows. Section 4.1 defines the reinforcement learning setting. Section 4.2 presents the SAC algorithm. Section 4.3 describes the overall training workflow of the proposed multi-agent SAC framework. Table 2 summarises the notation used in the reinforcement learning model.

Table 2: Notation for the reinforcement learning model.

| **Notation** | **Description** |
|---|---|
| $d$ | Decision round index, where $d \in \{1, \dots, \mathcal{D}\}$. |
| $e$ | Episode index, where $e \in \{1, \dots, \mathcal{E}\}$. |
| $p$ | Index of an SAV provider. |
| $s_d$ | Shared state observed by all SAV providers in decision round $d$. |
| $a_{p,d}$ | Action selected by SAV provider $p$ for interaction with the environment in decision round $d$. |
| $a_d$ | Joint action of all providers in decision round $d$, where $a_d = (a_{1,d}, \dots, a_{P,d})$. |
| $r_{p,d}$ | Reward received by SAV provider $p$ in decision round $d$. |
| $r_d$ | Joint reward vector in decision round $d$, where $r_d = (r_{1,d}, \dots, r_{P,d})$. |
| $\underline{a}_p$ | Lower bounds of the action vector of provider $p$. |
| $\overline{a}_p$ | Upper bounds of the action vector of provider $p$. |
| $\delta_d$ | Terminal indicator in decision round $d$. |
| $\mathcal{B}$ | Shared replay buffer used by all providers. |
| $\mathcal{B}_p^{\text{inf}}$ | Informative pool for provider $p$. |
| $\mathcal{B}^{\text{std}}$ | Standard pool. |
| *Critic-related notation* | |
| $\theta_{p,i}$ | Parameter vector of Q-network $i$ of provider $p$, where $i \in \{1, 2\}$. |
| $\bar{\theta}_{p,i}$ | Parameter vector of target Q-network $i$ of provider $p$, where $i \in \{1, 2\}$. |
| $\tilde{a}_{d+1}$ | Auxiliary joint action sampled from the current policies at state $s_{d+1}$ for computing the critic target. |
| $Q_{\theta_{p,i}}(s_d, a_d)$ | Q-value estimated by trainable Q-network $i \in \{1, 2\}$ of provider $p$ for the sampled state and joint action. |
| $Q_{\bar{\theta}_{p,i}}(s_{d+1}, \tilde{a}_{d+1})$ | Q-value estimated by target Q-network $i \in \{1, 2\}$ of provider $p$ for the next state and auxiliary joint action. |

Continued on next page

| Notation | Description |
|---|---|
| $\gamma$ | Discount factor applied to the next state soft Q-value in the critic target. |
| $y^{\delta}_{p,d}$ | Target Q-value of provider $p$ in decision round $d$. |
| $J_Q(\theta_{p,i})$ | Loss function of Q-network $i \in \{1,2\}$ of provider $p$. |
| $\widehat{\nabla}_{\theta_{p,i}} J_Q(\theta_{p,i})$ | Estimated gradient used to update Q-network $i \in \{1,2\}$ of provider $p$. |
| $\eta_Q$ | Learning rate for the Q-networks. |
| $\rho$ | Soft update coefficient for the target Q-networks. |
| *Actor-related notation* | |
| $\phi_p$ | Parameter vector of the policy network of provider $p$. |
| $\omega_{\phi_p}(a_{p,d} \mid s_d)$ | Probability density of selecting action $a_{p,d}$ given state $s_d$ for SAV provider $p$ in decision round $d$, parameterised by $\phi_p$. |
| $\epsilon_{p,d}$ | Gaussian noise vector sampled for provider $p$ in decision round $d$. |
| $h_{\phi_p}(\epsilon_{p,d}; s_d)$ | Deterministic transformation used to generate an auxiliary action for provider $p$ from state $s_d$ and noise $\epsilon_{p,d}$. |
| $\tilde{a}_{p,d}$ | Auxiliary action generated for provider $p$ at state $s_d$ for the actor update. |
| $\tilde{a}_d$ | Auxiliary joint action for the actor update, where $\tilde{a}_d = (\tilde{a}_{1,d}, \ldots, \tilde{a}_{P,d})$. |
| $J_\omega(\phi_p)$ | Loss function of the policy network of provider $p$. |
| $\widehat{\nabla}_{\phi_p} J_\omega(\phi_p)$ | Estimated gradient used to update the policy network of provider $p$. |
| $\eta_\omega$ | Learning rate for the policy network. |
| *Entropy and temperature related notation* | |
| $\psi_p$ | Temperature parameter of provider $p$, which controls the weight of the entropy term. |
| $\mathcal{H}\left(\omega_{\phi_p}(\cdot \mid s_d)\right)$ | Differential entropy of the policy of SAV provider $p$ at state $s_d$ in decision round $d$. |
| $\bar{\mathcal{H}}$ | Target entropy used for automatic temperature adjustment. |
| $J(\psi_p)$ | Temperature loss expressed in terms of the temperature parameter of provider $p$. |
| $J(\log \psi_p)$ | Temperature loss expressed in terms of the logarithmic temperature parameter and used in the implementation. |
| $\widehat{\nabla}_{\log \psi_p} J(\log \psi_p)$ | Estimated gradient used to update the logarithmic temperature parameter of provider $p$. |
| $\eta_\psi$ | Learning rate for the logarithmic temperature parameter. |

## 4.1 Reinforcement learning setting

This section introduces the reinforcement learning setting used in this study. The main components, including agents, episodes, decision rounds, states, actions, learning environment, and the state transition are defined in sequence.

**Agent** Each SAV provider $p$ is treated as an agent.

**Episodes and decision rounds** Training is organised into episodes, and each episode consists of multiple decision rounds. At the beginning of each episode, the state (defined below) is reset to an initial state. Each episode then proceeds for a predefined number of decision rounds. In each decision round, each provider first observes the shared state and selects an action. After all providers have selected their actions, the environment evaluates the joint actions, returns a reward to each provider, and updates the state. The resulting transition is then stored in the replay buffer for network training.

**State** At the beginning of each episode, the state of every provider is initialised as $s_{p,1} = a_0$, where $a_0$ is a predefined initial joint action. For each subsequent decision round $d$, the state observed by provider $p$ is defined as $s_{p,d} = a_{d-1}$, where $a_{d-1} = (a_{1,d-1}, \ldots, a_{P,d-1})$ denotes the joint action of all providers in decision round $d-1$. Because all providers observe the same joint action from the previous decision round, their states are identical and can be expressed as $s_{1,d} = s_{2,d} = \cdots = s_{P,d} = a_{d-1}$. For notational simplicity, this shared state is denoted by $s_d$ hereafter.

**Action** The action variable of provider $p$ represents its pricing and service decisions. Formally, the action of provider $p$ in decision round $d$ is defined as $a_{p,d} = \left(f^0_{p,d}, f_{p,d}, \bar{w}_{p,d}, \alpha_{p,d}\right)$, where $f^0_{p,d}$ is the base fare, $f_{p,d}$ is the distance-based fare, $\bar{w}_{p,d}$ is the maximum waiting time, and $\alpha_{p,d}$ is the service rate. All providers select their actions simultaneously under a parallel update scheme. The individual actions are then combined into the joint action $a_d = (a_{1,d}, \ldots, a_{P,d})$, which is passed to the environment.

In the implementation, the SAC actor first outputs four normalised values between 0 and 1. These values are then mapped to the actual ranges of the base fare, distance-based fare, maximum waiting time, and service rate before being passed to the environment.

**Environment** The learning environment of provider $p$ is the market feedback environment introduced in Sections 3.2 to 3.6.

The environment receives the joint action $a_d$ of all providers in decision round $d$. Based on this joint action, the passenger choice model allocates demand among the providers. A provider-specific fleet planning and operational optimisation model is then solved for each provider, and the resulting profit is returned as the reward.

**Reward** The reward $r_{p,d}$ for provider $p$ in decision round $d$ is defined as the total profit $\pi_{p,d}$ obtained by provider $p$. The joint reward vector is defined as $r_d = (r_{1,d}, \ldots, r_{P,d})$. Following Haarnoja et al. (2018), we scale the rewards in the implementation because SAC is sensitive to the scale of the reward signal. A sensitivity analysis of different reward scales is presented in Section 6.2.1.

**State transition** At the end of decision round $d$, the next state for decision

round $d+1$ is constructed from the joint action of all providers in decision round $d$, denoted by $s_{d+1} = a_d$. The resulting transition is represented by $(s_d, a_d, r_d, s_{d+1})$.

## 4.2 Soft Actor-Critic algorithm

This section presents the Soft Actor-Critic (SAC) algorithm proposed by Haarnoja et al. (2018), which is used to train the agents representing SAV providers in the proposed learning-based framework. SAC is selected because its off-policy design improves sample efficiency by reusing previously collected experience, which is important since each new interaction requires solving the optimisation model. Its entropy term promotes exploration and may help reduce the risk of convergence to a suboptimal strategy.

For each SAV provider, the SAC agent comprises five feedforward neural networks: one stochastic actor network, two centralised online critic networks (online soft Q-networks), and two target critic networks (target Q-networks). The actor network represents the stochastic policy and generates actions for each provider. The soft Q-networks estimate the value of each state-action pair. The term "soft" indicates that these values include both the reward and the entropy term that encourages exploration. The target Q-networks are slowly updated from the soft Q-networks and are used to construct more stable target Q-values during critic training. During critic training, SAC uses the smaller output of the two target critic networks to construct the target Q-value. During actor training, it uses the smaller output of the two online critic networks to evaluate the sampled action. This use of twin critics helps reduce overestimation bias.

Each feedforward neural network consists of an input layer, one or more hidden layers, and an output layer. During forward propagation, the input values pass sequentially through the hidden layers to the output layer. Each neuron in a hidden layer computes a weighted sum of the outputs from the previous layer, adds a bias term, and applies a nonlinear activation function. The resulting value is then passed to the next layer. These nonlinear transformations enable the network to represent complex relationships between its inputs and outputs. After the loss functions of the actor and the two centralised critic networks have been computed, backpropagation calculates the gradients with respect to the network weights and biases (LeCun et al., 2015). In the implementation, each actor and critic network contains two hidden layers, each with 256 neurons, and applies the ReLU activation function after each hidden layer. The network weights and biases are iteratively updated using the Adaptive Moment Estimation (Adam) optimiser (Kingma and Ba, 2014), which computes adaptive parameter updates from the gradients. The parameters of the two target critic networks are updated separately through soft updates from their corresponding centralised critic networks.

**Replay buffer** To improve sample efficiency, we adapt the highlight experience replay approach proposed by Horváth et al. (2024). The shared replay buffer is organised into one informative pool for each provider $p \in P$ and one

standard pool: $\mathcal{B} = \{\mathcal{B}_1^{\text{inf}}, \ldots, \mathcal{B}_P^{\text{inf}}, \mathcal{B}^{\text{std}}\}$. Each pool stores complete joint transitions.

At the beginning of training, an empty shared replay buffer $\mathcal{B}$ is initialised. An initial random exploration phase is then conducted, during which actions are sampled uniformly from the feasible action ranges and the resulting transitions are stored in the replay buffer. Network updates begin only when sufficient transitions are available to construct a complete minibatch under the specified sampling rule. Each interaction with the environment generates a joint transition $(s_d, a_d, r_d, s_{d+1})$, where $r_d = (r_{1,d}, \ldots, r_{P,d})$ contains the rewards of all providers. Every joint transition is stored in the standard pool $\mathcal{B}^{\text{std}}$. Given the reward definition in this thesis, a nonzero reward provides an informative learning signal for the corresponding provider. To account for numerical precision, the reward $r_{p,d}$ is treated as nonzero when $|r_{p,d}| > 10^{-6}$. A transition satisfying this condition is therefore considered informative for provider $p$. In this case, the complete joint transition $(s_d, a_d, r_d, s_{d+1})$ is additionally stored in $\mathcal{B}_p^{\text{inf}}$. If the rewards of multiple providers satisfy the informative criterion, the transition is stored in the informative pool of each corresponding provider. If no provider's reward satisfies the criterion, the transition remains only in the standard pool.

At the end of each decision round, the current transition is first stored in the relevant pools. A shared minibatch is then constructed for network training. In the implementation, the sampling ratio is chosen to increase the representation of informative transitions while distributing informative samples equally across the providers' informative pools and retaining samples from the standard pool. Specifically, in the experiments involving two providers, 25% of each minibatch is sampled from the informative pool of provider 1, 25% from the informative pool of provider 2, and the remaining 50% from the standard pool.

This sampling strategy can improve learning efficiency because many transitions collected during random exploration and the early stages of training may yield zero rewards. Under uniform sampling, a minibatch may therefore consist mainly of transitions with zero rewards, making it difficult for the critic networks to learn the relationship between provider actions and profits and providing the actor networks with weak learning signals.

**Critic** For each transition in the minibatch, a target Q-value is constructed using the soft Bellman equation. The first step is to sample an auxiliary next action from the current policy conditioned on the next state, which is written as $\tilde{a}_{p,d+1} \sim \omega_{\phi_p}(\cdot \mid s_{d+1})$, where $\omega_{\phi_p}$ denotes the stochastic policy of provider $p$ parameterised by $\phi_p$. The sampled actions are combined into the auxiliary joint next-action vector $\tilde{a}_{d+1}$, which is used to compute the target Q-values. Let $Q_{\bar{\theta}_{p,1}}$ and $Q_{\bar{\theta}_{p,2}}$ denote the two target Q-networks of provider $p$.

A centralised critic architecture is used, under which each critic observes the shared state and the joint actions of all providers. Accordingly, the two target Q-networks evaluate the next state-action pair $(s_{d+1}, \tilde{a}_{d+1})$. If the critic observed only the state and its own provider's action, the same input could correspond to different outcomes when the other providers select different actions, thereby making the learning signal unclear. We introduce $y_{p,d}$ to denote the target Q-

value for provider $p$ in decision round $d$. The target Q-value is defined as the current reward $r_{p,d}$ plus the discounted target value of the next state-action pair after the entropy adjustment:

$$y_{p,d} = r_{p,d} + \gamma \left[ \min_{i=1,2} Q_{\bar{\theta}_{p,i}}(s_{d+1}, \tilde{a}_{d+1}) - \psi_p \log \omega_{\phi_p}(\tilde{a}_{p,d+1} \mid s_{d+1}) \right], \tag{24}$$

where $\gamma$ is the discount factor and $\psi_p$ is the temperature parameter for provider $p$.

The two online critic networks, $Q_{\theta_{p,1}}$ and $Q_{\theta_{p,2}}$, use the sampled state-action pair $(s_d, a_d)$ as input and output the corresponding Q-value estimates: $Q_{\theta_{p,i}}(s_d, a_d), \quad i \in \{1, 2\}$. The online critic networks are trained by minimising the mean squared error between the estimated Q-value $Q_{\theta_{p,i}}(s_d, a_d)$ and the target Q-value $y_{p,d}$. For each $i \in \{1, 2\}$, the critic loss is defined as

$$J_Q(\theta_{p,i}) = \mathbb{E}_{(s_d, a_d, r_d, s_{d+1}) \sim \mathcal{B}} \left[ \frac{1}{2} \left( Q_{\theta_{p,i}}(s_d, a_d) - y_{p,d} \right)^2 \right]. \tag{25}$$

During the critic update, the smaller output of the two target critic networks is used to construct $y_{p,d}$. Once $y_{p,d}$ has been computed and treated as fixed, each online critic network independently minimises the squared difference between its own Q-value estimate and $y_{p,d}$.

Following the SAC implementation provided by the Robotic AI & Learning Lab at UC Berkeley,[1] the target Q-value includes a terminal indicator $\delta_d$, which indicates whether decision round $d$ is the final round of an episode. In this thesis, since all agents interact within the same episode and terminate simultaneously, a single terminal indicator is shared by all critics. If $d$ is the final decision round of an episode, then $\delta_d = 1$, and the next-state value is excluded from the critic target. Otherwise, $\delta_d = 0$, and the target Q-value includes the discounted value of the next state. Eq. (26) and Eq. (27) define the target Q-value with the terminal indicator and the corresponding stochastic critic gradient.

$$y^{\delta}_{p,d} = r_{p,d} + (1 - \delta_d)\gamma \left[ \min_{i=1,2} Q_{\bar{\theta}_{p,i}}(s_{d+1}, \tilde{a}_{d+1}) - \psi_p \log \omega_{\phi_p}(\tilde{a}_{p,d+1} \mid s_{d+1}) \right]. \tag{26}$$

$$\widehat{\nabla}_{\theta_{p,i}} J_Q(\theta_{p,i}) = \nabla_{\theta_{p,i}} Q_{\theta_{p,i}}(s_d, a_d) \left( Q_{\theta_{p,i}}(s_d, a_d) - y^{\delta}_{p,d} \right), \quad i \in \{1, 2\}. \tag{27}$$

**Actor** After the critic networks have been updated, the actor network is updated using the Q-value estimates provided by the online critics. The actor network of provider $p$ represents a stochastic policy network $\omega_{\phi_p}(a_{p,d} \mid s_d)$, where $\phi_p$ denotes the trainable parameters. The term $\omega_{\phi_p}(a_{p,d} \mid s_d)$ denotes the probability density assigned to action $a_{p,d}$ given the observed state $s_d$ in decision round $d$. This decision-making process is stochastic because the policy network

[1] https://github.com/rail-berkeley/softlearning/blob/master/softlearning/algorithms/sac.py#L29-L34

maps the state to a probability distribution over the action space instead of directly outputting a single deterministic action.

SAC is a maximum entropy reinforcement learning framework that combines the expected return with an entropy term to encourage exploration. The entropy term reduces the risk of the policy converging too early to a suboptimal solution. Following Shannon (1948), the differential entropy of a continuous random variable $X$ with probability density function $h(x)$ is defined as

$$\mathcal{H}(X) = \mathbb{E}_{x\sim h}\left[-\log h(x)\right]. \tag{28}$$

Accordingly, the entropy of the policy $\omega_{\phi_p}$ at state $s_d$ is expressed as

$$\mathcal{H}\left(\omega_{\phi_p}(\cdot \mid s_d)\right) = \mathbb{E}_{a_{p,d}\sim\omega_{\phi_p}(\cdot|s_d)}\left[-\log \omega_{\phi_p}(a_{p,d} \mid s_d)\right]. \tag{29}$$

For provider $p$, the maximum entropy objective is to maximise the expected cumulative return while encouraging policy entropy

$$\phi_p^* = \arg\max_{\phi_p} \mathbb{E}_{s_d, a_{p,d}\sim\omega_{\phi_p}}\left[\sum_d \left(r_{p,d} + \psi_p \mathcal{H}\left(\omega_{\phi_p}(\cdot \mid s_d)\right)\right)\right], \tag{30}$$

where $\psi_p$ is the temperature parameter controlling the relative importance of the entropy term. A larger $\psi_p$ places greater weight on the entropy term and promotes stronger exploration, whereas a smaller $\psi_p$ reduces the influence of the entropy term and makes the policy focus more on actions with high expected returns.

In practice, the long-term objective in Eq. (30) is not optimised directly. For each state $s_d$, every actor network samples an auxiliary action from its current policy. These actions are combined into the auxiliary joint action $\tilde{a}_d$, where $\tilde{a}_d \sim \boldsymbol{\omega}(\cdot \mid s_d)$ and $\boldsymbol{\omega} = (\omega_{\phi_1}, \ldots, \omega_{\phi_P})$ denotes the joint policy of all providers. The policy parameters $\phi_p$ are updated by minimising an actor loss. This loss encourages the policy to assign higher probability to actions with larger soft Q-values while maintaining sufficient entropy for exploration. The actor loss function is defined as

$$J_\omega(\phi_p) = \mathbb{E}_{s_d\sim\mathcal{B}}\left[\mathbb{E}_{\tilde{a}_d\sim\boldsymbol{\omega}(\cdot|s_d)}\left[\psi_p \log \omega_{\phi_p}(\tilde{a}_{p,d} \mid s_d) - \min_{i=1,2} Q_{\theta_{p,i}}(s_d, \tilde{a}_d)\right]\right]. \tag{31}$$

Both the actor and the Q-network are represented by differentiable neural networks. Since the action is sampled from a stochastic policy, directly propagating the gradient of the actor objective with respect to $\phi_p$ through the sampling process is difficult. Therefore, the reparameterisation trick is applied to obtain a lower variance estimator. For this purpose, the policy is reparameterised using a neural network transformation

$$\tilde{a}_{p,d} = h_{\phi_p}(\epsilon_{p,d}; s_d), \qquad \epsilon_{p,d} \sim \mathcal{N}(0, \mathbf{I}). \tag{32}$$

where $h_{\phi_p}$ is a deterministic transformation parameterised by the actor network, and $\epsilon_{p,d}$ is an independent noise vector. For each sampled state $s_d$, the auxiliary

joint action is generated as

$$\tilde{a}_d = (\tilde{a}_{1,d}, \ldots, \tilde{a}_{P,d}), \qquad \tilde{a}_{j,d} = h_{\phi_j}(\epsilon_{j,d}; s_d). \tag{33}$$

The corresponding joint noise vector is denoted by $\epsilon_d = (\epsilon_{1,d}, \ldots, \epsilon_{P,d})$. When updating the actor network of provider $p$, the auxiliary actions $\tilde{a}_{j,d}$ for $j \neq p$ are treated as constants.

With this reparameterisation, the loss function in Eq. (31) is rewritten as

$$\begin{aligned} J_\omega(\phi_p) = \mathbb{E}_{s_d \sim \mathcal{B},\, \boldsymbol{\epsilon}_d \sim \mathcal{N}} \Big[ & \psi_p \log \omega_{\phi_p}\big(h_{\phi_p}(\epsilon_{p,d}; s_d) \mid s_d\big) \\ & - \min_{i=1,2} Q_{\theta_{p,i}}\big(s_d, \tilde{a}_d\big) \Big]. \end{aligned} \tag{34}$$

By applying the chain rule to the reparameterised objective, the stochastic gradient of $J_\omega(\phi_p)$ with respect to the policy parameters $\phi_p$ can be written as

$$\begin{aligned} \widehat{\nabla}_{\phi_p} J_\omega(\phi_p) = \mathbb{E}_{s_d \sim \mathcal{B},\, \boldsymbol{\epsilon}_d \sim \mathcal{N}} \Big[ & \nabla_{\phi_p} \psi_p \log \omega_{\phi_p}(\tilde{a}_{p,d} \mid s_d) \\ & + \Big( \nabla_{\tilde{a}_{p,d}} \psi_p \log \omega_{\phi_p}(\tilde{a}_{p,d} \mid s_d) - \nabla_{\tilde{a}_{p,d}} Q_{\theta_{p,i^*}}(s_d, \tilde{a}_d) \Big) \\ & \times \nabla_{\phi_p} h_{\phi_p}(\epsilon_{p,d}; s_d) \Big], \end{aligned} \tag{35}$$

where

$$i^* = \arg \min_{i \in \{1,2\}} Q_{\theta_{p,i}}(s_d, \tilde{a}_d)$$

denotes the index of the critic network with the smaller Q-value.

**Temperature parameter** After the actor update, the temperature parameter is updated. Instead of fixing the temperature parameter manually, SAC can automatically adjust it by treating the policy entropy as a constraint (Haarnoja et al., 2018). Based on the maximum entropy objective in Eq. (30), the constrained objective for provider $p$ is written as

$$\begin{aligned} \max_{\phi_p} \quad & \mathbb{E}_{\boldsymbol{\omega}} \left[ \sum_d r_{p,d} \right] \\ \text{s.t.} \quad & \mathbb{E}_{s_d \sim \mathcal{B}} \left[ \mathcal{H}\left(\omega_{\phi_p}(\cdot \mid s_d)\right) \right] \geq \bar{\mathcal{H}}. \end{aligned} \tag{36}$$

where $\bar{\mathcal{H}}$ is the target entropy of the policy of provider $p$.

Since the entropy term $\mathcal{H}(\omega_{\phi_p}(\cdot \mid s_d))$ already includes the expectation over actions sampled from $\omega_{\phi_p}(\cdot \mid s_d)$, the outer expectation in Eq. (36) is taken only over states sampled from the replay buffer $\mathcal{B}$. Following the target entropy setting reported in *Appendix D* of Haarnoja et al. (2018), the target entropy in SAC is set to the negative dimension of the action vector. Since each provider's action vector is four-dimensional in this study, the reference target entropy in the original squashed action space is $-4$. *Appendix C* of Haarnoja et al. (2018) shows that, in practice, the probability density of a transformed action usually includes the corresponding Jacobian determinant. In the implementation, the entropy

calculation includes both the tanh transformation and the subsequent rescaling from $(-1, 1)$ to $(0, 1)$ with respect to the normalised action range. Therefore, a Jacobian correction is also applied to the entropy. Based on the formula in *Appendix C*, the target entropy is adjusted to $\bar{\mathcal{H}} = -4 - 4\log 2 \approx -6.773$.

The temperature parameter $\psi_p$ is then updated by minimising the following loss:

$$J(\psi_p) = \mathbb{E}_{s_d \sim \mathcal{B}} \left[ \psi_p \left( \mathcal{H} \left( \omega_{\phi_p}(\cdot \mid s_d) \right) - \bar{\mathcal{H}} \right) \right]. \tag{37}$$

In the implementation, the logarithmic form $\log \psi_p$ is optimised instead of $\psi_p$ to ensure that the temperature parameter remains positive. Consequently, the temperature loss function in Eq. (37) can be written as

$$J(\log \psi_p) = \mathbb{E}_{s_d \sim \mathcal{B}} \left[ \log \psi_p \left( \mathcal{H} \left( \omega_{\phi_p}(\cdot \mid s_d) \right) - \bar{\mathcal{H}} \right) \right]. \tag{38}$$

The gradient used to update $\log \psi_p$ is

$$\widehat{\nabla}_{\log \psi_p} J(\log \psi_p) = \mathbb{E}_{s_d \sim \mathcal{B}} \left[ \mathcal{H} \left( \omega_{\phi_p}(\cdot \mid s_d) \right) - \bar{\mathcal{H}} \right]. \tag{39}$$

**Target critic network** After the online critic networks, actor networks, and temperature parameters have been updated, the target critic networks are updated to gradually track the online critic networks. For each provider $p$, the target critic parameters are updated using

$$\bar{\theta}_{p,i} \leftarrow \rho \theta_{p,i} + (1 - \rho) \bar{\theta}_{p,i}, \quad i \in \{1, 2\}. \tag{40}$$

where $\rho \in (0, 1)$ is the soft update coefficient.

**Training procedure** The training procedure consists of two stages that are repeated throughout training. First, the agents interact with the environment and collect transition data, as shown in Figure 3. In decision round $d$, provider $p$ observes its state $s_d$ and selects an action $a_{p,d}$. During the initial exploration phase, this action is sampled randomly. After the initial phase, the action is sampled from the current policy according to $a_{p,d} \sim \omega_{\phi_p}(\cdot \mid s_d)$.

After all providers have selected their actions in decision round $d$, the environment evaluates the joint actions and returns the next state $s_{d+1}$, the reward $r_{p,d}$, and the shared terminal indicator $\delta_d$. The resulting joint transition is $(s_d, a_d, r_d, s_{d+1}, \delta_d)$. The joint transition is then stored in the replay buffer:

$$\mathcal{B} \leftarrow \mathcal{B} \cup \{(s_d, a_d, r_d, s_{d+1}, \delta_d)\}.$$

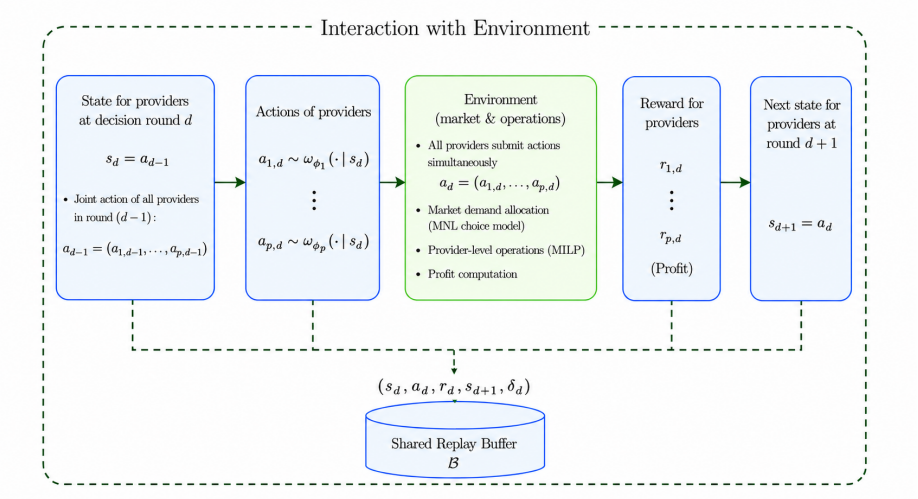



Figure 3: Environment interaction process within one decision round.

Second, the networks are updated using data from the replay buffer. Once the replay buffer contains enough transitions, minibatches are sampled from $\mathcal{B}$ for agent learning network updates. For each next state $s_{d+1}$ in the minibatch, every provider samples an auxiliary action from its current policy. These actions are combined into the auxiliary joint action $\tilde{a}_{d+1}$. Each provider then calculates its target Q-value $y^{\delta}_{p,d}$ according to Eq. (26) and then updates its online critic networks using the gradient in Eq. (27).

After the critic networks have been updated, each provider samples another auxiliary action from its current policy for each state $s_d$ in the same minibatch. These actions are combined into the auxiliary joint action $\tilde{a}_d$ for the actor updates. Each provider then calculates its actor gradient according to Eq. (35), using its updated online critic networks. When computing the gradient for provider $p$, the auxiliary actions of the other providers are treated as constants. All actor losses and gradients are calculated before any actor parameters are updated. This prevents one provider from using another provider's updated policy during the same network update. Each provider then updates its actor parameters using the corresponding gradient.

Finally, after the actor parameters have been updated, each provider adjusts its temperature parameter according to Eq. (39) and then updates its target critic networks using the soft update in Eq. (40). Figure 4 illustrates the synchronous network update process performed at the end of a decision round using two providers as an example.

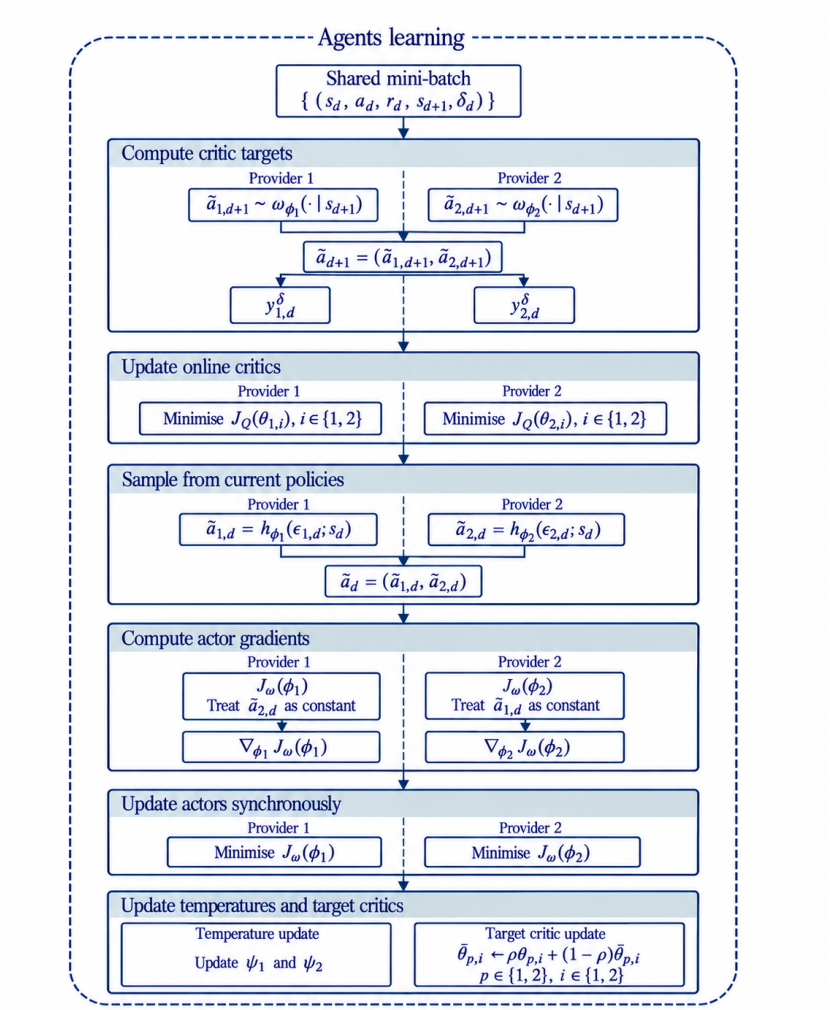



Figure 4: Synchronous network update process for two SAV providers.

## 4.3 Overall workflow

Algorithm 1 summarises the training procedure following the initial random exploration phase and assumes that the replay buffer contains sufficient transitions to sample a complete minibatch. Let $e \in \{1, \ldots, \mathcal{E}\}$ denote the episode index, where $\mathcal{E}$ is the total number of training episodes, and let $d \in \{1, \ldots, \mathcal{D}\}$ denote the decision-round index, where $\mathcal{D}$ is the number of decision rounds in each episode. At the beginning of each episode, an initial action vector $a_{p,0}$ is randomly generated for each provider $p \in P$. It is sampled componentwise from the uniform distribution $a_{p,0} \sim U\left(\underline{a}_p, \overline{a}_p\right)$, where $\underline{a}_p$ and $\overline{a}_p$ denote the lower and upper bounds of the action vector of provider $p$, respectively. The initial

shared state is then constructed as $s_1 = (a_{1,0}, \ldots, a_{P,0})$.

**Algorithm 1** Multi-Agent SAC Algorithm

Initialise $\mathcal{B}$ ▷ Initialise an empty replay buffer
**for** each SAV provider $p$ **do**
  $\theta_{p,1}, \theta_{p,2}, \phi_p, \psi_p$ ▷ Initialise parameters
  $\bar{\theta}_{p,1} \leftarrow \theta_{p,1}$, $\bar{\theta}_{p,2} \leftarrow \theta_{p,2}$ ▷ Initialise target network weights
**end for**
**for** episode $e = 1, \ldots, \mathcal{E}$ **do**
  **for** each SAV provider $p \in P$ **do**
    $a_{p,0} \sim U(\underline{a}_p, \overline{a}_p)$
  **end for**
  $s_1 = (a_{1,0}, \ldots, a_{P,0})$ ▷ Construct the initial state from random actions
  **for** decision round $d = 1, \ldots, \mathcal{D}$ **do**
    **for** each SAV provider $p \in P$ in parallel **do**
      $a_{p,d} \sim \omega_{\phi_p}(a_{p,d} \mid s_d)$ ▷ Sample action from the policy
    **end for**
    $a_d = (a_{1,d}, \ldots, a_{P,d})$ ▷ Construct joint action vector
    Calculate $D_{p,d}$ based on $a_d$
    **for** each SAV provider $p \in P$ in parallel **do**
      Solve the provider-specific optimisation model
      $r_{p,d} = \pi_{p,d}$ ▷ Obtain reward from the optimisation model
    **end for**
    $r_d = (r_{1,d}, \ldots, r_{P,d})$ ▷ Construct joint reward vector
    $s_{d+1} = a_d$ ▷ Construct next shared state
    $\mathcal{B} \leftarrow \mathcal{B} \cup \{(s_d, a_d, r_d, s_{d+1}, \delta_d)\}$ ▷ Store the transition in the replay buffer
    Sample a minibatch from $\mathcal{B}$
    **for** each SAV provider $p \in P$ in parallel **do**
      $\theta_{p,i} \leftarrow \theta_{p,i} - \eta_Q \widehat{\nabla}_{\theta_{p,i}} J_Q(\theta_{p,i}), \quad i \in \{1, 2\}$ ▷ Update the critic parameters
      $\phi_p \leftarrow \phi_p - \eta_\omega \widehat{\nabla}_{\phi_p} J_\omega(\phi_p)$ ▷ Update policy weights
      $\log \psi_p \leftarrow \log \psi_p - \eta_\psi \widehat{\nabla}_{\log \psi_p} J(\log \psi_p)$ ▷ Adjust temperature
      $\bar{\theta}_{p,i} \leftarrow \rho \theta_{p,i} + (1 - \rho) \bar{\theta}_{p,i}, \quad i \in \{1, 2\}$ ▷ Update target network weights
    **end for**
  **end for**
  **if** early stopping is enabled and the convergence criterion is satisfied **then**
    **break**
  **end if**
**end for**

To evaluate convergence, the deterministic evaluation actions are used as the main indicator. During training, SAC samples actions from a stochastic policy, so the training rewards may fluctuate because of exploration. During

deterministic evaluation, each provider selects the action generated from the mean of its current policy distribution. If the deterministic evaluation actions of all providers remain stable during training, the learned policies are considered to have stabilised. Under fixed environment settings, stable deterministic actions also produce stable rewards.

# 5 Diagonalisation method

For comparison with the learning-based framework, this study considers the diagonalisation method proposed by Yang et al. (2022). In this approach, the optimisation problem is formulated as a nonlinear mixed integer programming model. This section first introduces the diagonalisation method. The maximum waiting time is then defined, followed by the linearisation of the nonlinear components in the model. We have summarised the mathematical notations in Table 3 for easy reference.

Table 3: Notation for the diagonalisation method.

| Notation | Description |
|---|---|
| *Diagonalisation method* | |
| $g$ | Iteration index of the diagonalisation method. |
| $\varepsilon_{tol}$ | Convergence tolerance. |
| $a_p^g$ | Decision vector of SAV provider $p$ at iteration $g$. |
| $a_{-p}^g$ | Decision vectors of all SAV providers except provider $p$ at iteration $g$. |
| $a^g$ | Joint decision vector of all SAV providers at iteration $g$. |
| $\hat{a}_p$ | Best-response decision of SAV provider $p$. |
| $\|a_p^g - a_p^{g-1}\|_2$ | Euclidean distance between the decision vectors of SAV provider $p$ in two consecutive iterations. |
| *Maximum waiting time* | |
| $b^r$ | Latest arrival time for group of trips $r \in R$. |
| $E_t^{r,p}$ | Total number of passengers in group of trips $r \in R$ served by provider $p \in P$ arriving at time $t \in T$. |
| $A_t^{r,p}$ | Binary variable equal to 1 if at least one passenger in group $r \in R$ served by provider $p \in P$ arrives at time $t \in T$, and 0 otherwise. |
| *Binary logit linearisation* | |
| $q$ | Representative competing alternative for SAV provider $p \in P$. |
| $V_q^r$ | Deterministic utility of the representative competing alternative $q$ for group of trips $r \in R$. |
| $LN_p^r$ | Auxiliary continuous variable, where $p \in P$, $r \in R$. |
| $LN_q^r$ | Auxiliary continuous variable, where $r \in R$. |
| $K$ | $= \{1, 2, \ldots, k, \ldots, \mathcal{K}\}$. Index set of predetermined breakpoints. |
| $(u^k, \ln u^k)$ | Coordinate of the $k$th breakpoint. |
| $\mu_{r,p}^k$ | Convex combination coefficient for breakpoint $k$ for group $r \in R$, and SAV provider $p \in P$. |

Continued on next page

| Notation | Description |
|---|---|
| $\lambda_{r,p}^{k}$ | Binary variable indicating whether an interval $[u^k, u^{k+1}]$ is active or not, where $k \in \{1, 2, \ldots, k, \ldots, \mathcal{K}\}$, $r \in R$ and $p \in P$. |
| *Service rate linearisation* | |
| $\bar{D}_c^p$ | Binary variable utilised for describing integer variable $\sum_{r \in R} D_p^r$ where $c \in \{0, 1, \ldots, \mathcal{C}\}$ and $p \in P$. |
| $Y_c^p$ | Continuous variable utilised for describing the value of the integer variable $\sum_{r \in R} S_p^r$ where $c \in \{0, 1, \ldots, \mathcal{C}\}$ and $p \in P$. |
| *Objective function linearisation* | |
| $\bar{S}_c^{r,p}$ | Binary variable used to represent the integer variable $S_p^r$, where $c \in \{0, 1, \ldots, \mathcal{C}_r\}$, $r \in R$, and $p \in P$. |
| $\overline{OM}_p^r$ | Upper bound on the monetary cost $OM_p^r$, where $r \in R$ and $p \in P$. |
| $Z_c^{r,p}$ | Continuous variable representing $OM_p^r \bar{S}_c^{r,p}$, where $c \in \{0, 1, \ldots, \mathcal{C}_r\}$, $r \in R$, and $p \in P$. |

## 5.1 Diagonalisation method

In the diagonalisation method, the fare, service rate, and maximum waiting time are provider-level decision variables jointly determined to maximise each provider's profit. The main constraints and functions involved in this optimisation problem are as follows. The longest travel time and deterministic utility are defined in Eq. (2) and Eq. (3), respectively. The corresponding choice probability is given in Eq. (4), and the demand is given in Eq. (6). The service quality requirements are imposed by Constraints (8)–(9), the passenger flow constraints are given by Constraints (10)–(15), and the vehicle flow constraints are given by Constraints (16)–(20). The resulting profit is calculated by the provider profit function in Eq. (21)–(22), and the objective is given by the provider objective function in Eq. (23).

For a given provider, the decisions of the other providers are fixed and treated as competitors' decision inputs. The optimisation problem is then solved for the selected provider under these fixed competitors' decisions. Once the model is solved, the selected provider's updated decisions replace its previous decisions and are used in the optimisation model of the next provider. This sequential update process is repeated until the providers' decisions become stable.

The procedure of the diagonalisation approach is summarised in Algorithm 2, and the optimisation approach is illustrated in Fig. 5.

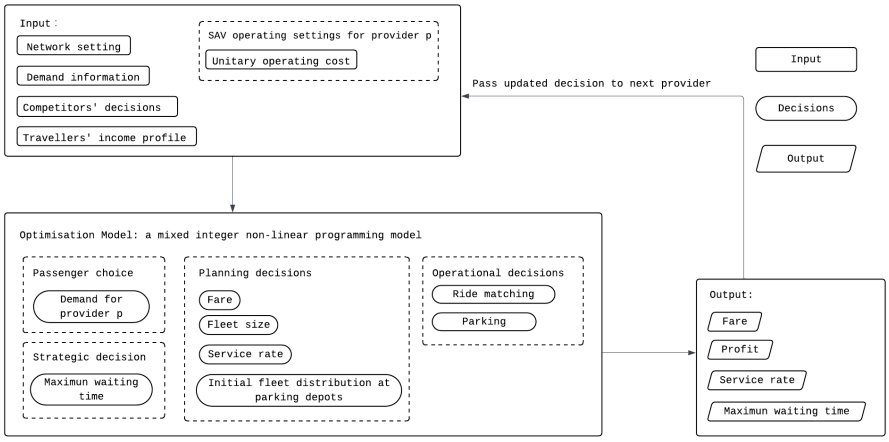



Figure 5: Structure of the diagonalisation-based optimisation method.

**Algorithm 2** Diagonalisation Method

Choose initial decisions $a_p^0$ for all $p \in P$
Set $a^0 = [a_1^0, \ldots, a_p^0, \ldots, a_{|P|}^0]$
Set $g \leftarrow 1$
**while** true **do**
    Set $a^g \leftarrow a^{g-1}$
    **for** each provider $p \in P$ **do**
        Set $a_{-p}^g = (a_1^g, \ldots, a_{p-1}^g, a_{p+1}^{g-1}, \ldots, a_{|P|}^{g-1})$ ▷ Fix competitors' decisions
        Compute $a_p^g \in \arg\max_{a_p \in \mathcal{A}_p} \pi_p(a_p, a_{-p}^g)$ ▷ Solve provider $p$'s optimisation problem
    **end for**
    **if** $\|a_p^g - a_p^{g-1}\|_2 \leq \varepsilon_{tol}$ for all $p \in P$ **then**
        **break**
    **end if**
    Set $g \leftarrow g + 1$
**end while**

## 5.2 Maximum waiting time

In the SAC method, the maximum waiting time $\bar{w}_p$ is fixed before the operational optimisation problem is solved. Therefore, the corresponding latest arrival time $a^r + \bar{w}_p + st^r$ is known in advance for provider $p$. In the diagonalisation method, however, $\bar{w}_p$ is a decision variable and cannot be used to define a fixed time range before the model is solved. The arrival time $t$ is therefore restricted to $a^r + st^r \leq t \leq b^r$, where $b^r$ is the latest arrival time specified by the user. Accordingly, the time ranges in the passenger flow constraints are

modified as follows. Constraints (10) becomes

$$S_p^r = \sum_{\tau \in T | a^r \leq \tau \leq b^r - st^r} \sum_{j_t | (o_\tau^r, j_t) \in G} PF_{o_\tau^r j_t}^{r,p}, \quad \forall r \in R. \tag{41}$$

Constraints (11) and Constraints (12) are reformulated as

$$S_p^r = \sum_{t \in T | a^r + st^r \leq t \leq b^r} E_t^{r,p}, \qquad \forall r \in R, \tag{42}$$

$$E_t^{r,p} = \sum_{i_{t_1} | (i_{t_1}, d_t^r) \in G} PF_{i_{t_1} d_t^r}^{r,p}, \qquad \forall r \in R,\ a^r + st^r \leq t \leq b^r. \tag{43}$$

Constraints (13) and Constraints (14) are changed to

$$\sum_{j_{t_1} | (d_t^r, j_{t_1}) \in G} PF_{d_t^r j_{t_1}}^{r,p} = 0, \quad \forall r \in R,\ a^r + st^r \leq t \leq b^r, \tag{44}$$

$$\sum_{i_{t_1} | (i_{t_1}, o_t^r) \in G} PF_{i_{t_1} o_t^r}^{r,p} = 0, \quad \forall r \in R,\ a^r \leq t \leq b^r. \tag{45}$$

Constraints (15) are therefore reformulated as follows

$$\sum_{j_{t_1} | (j_{t_1}, i_t) \in G} PF_{j_{t_1} i_t}^{r,p} = \sum_{j_{t_2} | (i_t, j_{t_2}) \in G} PF_{i_t j_{t_2}}^{r,p}, \quad \forall r \in R,\ a^r < t < b^r,\ i \in N \setminus \{o^r, d^r\}. \tag{46}$$

To specify the perceived travel time $T_p^r$ defined in Eq. (2), we introduce binary variables $A_t^{r,p}$, where $A_t^{r,p}$ is 1 if at least one passenger in group $r \in R$ served by provider $p \in P$ arrives at destination $d^r$ at time $t \in T$, and 0 otherwise. Constraints (47) specify the arrival times of trips in group $r \in R$.

$$\frac{E_t^{r,p}}{n^r} \leq A_t^{r,p} \leq E_t^{r,p}, \qquad \forall r \in R,\ a^r + st^r \leq t \leq b^r. \tag{47}$$

Constraints (48) impose a lower bound on the perceived travel time of trips in group $r \in R$ served by provider $p \in P$, meaning that it has to be greater than or equal to all the different travel times experienced by passengers.

$$T_p^r \geq A_t^{r,p}(t - a^r), \qquad \forall r \in R,\ a^r + st^r \leq t \leq b^r. \tag{48}$$

### 5.3 Linearisation of the binary logit model

In the learning-based framework, the fare and service settings are selected before the fleet planning and operational model is solved. Hence, the choice probabilities and the demand can be calculated outside the fleet planning and operational model and used as fixed inputs. In the diagonalisation method,

these fare and service settings are treated as decision variables. As a result, the utility, choice probability, and demand of provider $p$ depend on the optimisation variables, which makes the demand constraints derived from the multinomial logit model nonlinear.

To simplify the choice formulation, provider $p \in P$ is optimised against a representative competing alternative, denoted by $q$. Here, $q$ represents all alternatives other than provider $p$, including the bike alternative and all other SAV providers. Since the decisions of the other SAV providers and the attributes of the bike alternative are fixed in the best-response problem of provider $p \in P$, the utility of the representative competing alternative can be treated as fixed. It is defined as

$$V_q^r = \ln\left(e^{V_B^r} + \sum_{j \in P,\ j \neq p} e^{V_j^r}\right), \qquad \forall r \in R. \tag{49}$$

Therefore, the traveller choice problem in the best-response problem of provider $p \in P$ is formulated as a binary choice between provider $p \in P$ and the representative competing alternative $q$.

We rewrite Eq. (4) as:

$$\frac{\mathcal{P}_p^r}{1 - \mathcal{P}_p^r} = \frac{e^{V_p^r}}{e^{V_q^r}}, \forall r \in R,\ p \in P. \tag{50}$$

Then, we take the logarithm of both sides to obtain:

$$\ln \mathcal{P}_p^r - \ln(1 - \mathcal{P}_p^r) = V_p^r - V_q^r, \forall r \in R,\ p \in P. \tag{51}$$

By defining new variables $LN_p^r$ and $LN_q^r$, the equation can be further simplified as follows:

$$LN_p^r = \ln \mathcal{P}_p^r, \forall r \in R,\ p \in P, \tag{52}$$

$$LN_q^r = \ln(1 - \mathcal{P}_p^r), \forall r \in R,\ p \in P, \tag{53}$$

$$LN_p^r - LN_q^r = V_p^r - V_q^r, \forall r \in R,\ p \in P. \tag{54}$$

Following Wang et al. (2015) and Guo et al. (2022), we apply an outer-inner approximation to linearise the logarithmic terms in Eqs. (52) and (53). The logarithmic function is approximated by a set of linear constraints, which provide upper and lower bounds for the original logarithmic function.

The original logarithmic function is divided into $\mathcal{K} - 1$ segments by $\mathcal{K}$ pre-determined breakpoints, where $\mathcal{K}$ denotes the number of breakpoints. For each segment, the tangent lines at the two breakpoints are used as upper bounds, and the secant line between the two breakpoints is used as the lower bound of the logarithmic function. We introduce the breakpoint index set $K = \{1, 2, \ldots, k, \ldots, \mathcal{K}\}$. Each breakpoint $k \in K$ has coordinates $(u^k, \ln u^k)$, where $u^k \in [0, 1]$. The segment between two adjacent breakpoints $k$ and $k + 1$

is denoted by $[u^k, u^{k+1}]$. If the value of $\mathcal{P}_p^r$ falls within the interval $[u^k, u^{k+1}]$, this interval is called the active interval. A binary variable $\lambda_{r,p}^k$ is introduced to indicate whether interval $[u^k, u^{k+1}]$ is active, where $k \in \{1, 2, \ldots, \mathcal{K}-1\}$ for group $r \in R$.

A set of constraints is introduced to describe the outer-inner approximation. Constraints (55) describe the tangent lines at each breakpoint, which serve as the outer approximation of the logarithmic function.

$$LN_p^r \leq \frac{1}{u^k}\mathcal{P}_p^r + \ln u^k - 1, \qquad \forall r \in R,\ p \in P,\ k \in K. \tag{55}$$

Constraints (56)–(62) describe the inner approximation of the logarithmic function. The values of $LN_p^r$ and $\mathcal{P}_p^r$ can be represented by the convex combination of the coordinates of two consecutive breakpoints. The continuous variable $\mu_{r,p}^k$ represents the convex combination coefficient for breakpoint $k \in K$ and traveller group $r \in R$ as shown in Constraints (56) and (57).

$$LN_p^r \geq \sum_{k=1}^{\mathcal{K}} \mu_{r,p}^k \ln u^k, \qquad \forall r \in R,\ p \in P. \tag{56}$$

$$\mathcal{P}_p^r = \sum_{k=1}^{\mathcal{K}} \mu_{r,p}^k u^k, \qquad \forall r \in R,\ p \in P. \tag{57}$$

The summation of the convex combination coefficients has to be one, as shown in Constraint (58).

$$\sum_{k=1}^{\mathcal{K}} \mu_{r,p}^k = 1, \qquad \forall r \in R,\ p \in P. \tag{58}$$

There exists only one active interval for each traveller group $r$. This means that the values of $\mathcal{P}_p^r$ and $LN_p^r$ can only fall into one line segment, as ensured by Constraint (59).

$$\sum_{k=1}^{\mathcal{K}-1} \lambda_{r,p}^k = 1, \qquad \forall r \in R,\ p \in P. \tag{59}$$

The following constraints describe the relationship between two consecutive breakpoints and the active interval between them.

$$\mu_{r,p}^1 \leq \lambda_{r,p}^1, \qquad \forall r \in R,\ p \in P. \tag{60}$$

$$\mu_{r,p}^k \leq \lambda_{r,p}^{k-1} + \lambda_{r,p}^k, \qquad \forall r \in R,\ p \in P,\ k \in \{2, \ldots, \mathcal{K}-1\}. \tag{61}$$

$$\mu_{r,p}^{\mathcal{K}} \leq \lambda_{r,p}^{\mathcal{K}-1}, \qquad \forall r \in R,\ p \in P. \tag{62}$$

To use the outer-inner approximation, a set of breakpoints needs to be specified before solving the optimisation model. In this study, these breakpoints

are generated using the breakpoint determination method proposed by Fan et al. (2023).

### 5.4 Linearisation of the floor function

The demand calculation function in Eq. (6) is non-linear. Therefore, we replace Eq. (6) with the following constraints:

$$n^r \mathcal{P}_p^r - 0.5 < D_p^r \leq n^r \mathcal{P}_p^r + 0.5, \qquad \forall r \in R,\ p \in P. \tag{63}$$

### 5.5 Linearisation of the service rate constraint

In this model, the service rate $\alpha_p$ is defined as a continuous variable with $\alpha_p \in [0, 1]$. As a result, Constraint (8) becomes nonlinear because it contains the product of the continuous variable $\alpha_p$ and the integer demand variable $D_p^r$. To linearise this constraint, additional binary variables $\bar{D}_c^p$ are introduced to discretise the integer term $\sum_{r \in R} D_p^r$, and continuous variables $Y_c^p \in [0, 1]$ are introduced to describe the value of the continuous term $\alpha_p \bar{D}_c^p$, where $c \in \{0, 1, \ldots, \mathcal{C}\}$. Here, $\mathcal{C}$ should be chosen large enough so that the constraints remain valid when all traveller demand is assigned to provider $p \in P$. This product can be represented by the following linear constraints:

$$\sum_{r \in R} D_p^r = \sum_{c=0}^{\mathcal{C}} 2^c \bar{D}_c^p, \qquad \forall\, p \in P. \tag{64}$$

$$Y_c^p \leq \alpha_p, \qquad \forall\, p \in P,\ c \in \{0, 1, \ldots, \mathcal{C}\}. \tag{65}$$

$$Y_c^p \leq \bar{D}_c^p, \qquad \forall\, p \in P, c \in \{0, 1, \ldots, \mathcal{C}\}. \tag{66}$$

$$Y_c^p \geq \alpha_p + \bar{D}_c^p - 1, \qquad \forall\, p \in P, c \in \{0, 1, \ldots, \mathcal{C}\}. \tag{67}$$

$$\sum_{c=0}^{\mathcal{C}} 2^c Y_c^p \leq \sum_{r \in R} S_p^r, \qquad \forall\, p \in P. \tag{68}$$

### 5.6 Linearisation of the objective function

In Eq. (21), both the monetary cost $OM_p^r$ and the total number of trips $S_p^r$ in group $r \in R$ served by SAV provider $p \in P$ depend on decision variables. Consequently, the product of $OM_p^r S_p^r$ is bilinear. We introduce binary variables $\bar{S}_c^{r,p}$ for each group of trips $r \in R$ to represent the integer variable $S_p^r$, where $c \in \{0, 1, \ldots, \mathcal{C}_r\}$, and $\mathcal{C}_r$ should be chosen large enough to ensure that the formulation remains valid when all traveller demand in group of trips $r \in R$ is served by provider $p \in P$. Let $\overline{OM}_p^r$ denote an upper bound on $OM_p^r$, such that $0 \leq OM_p^r \leq \overline{OM}_p^r$. Using this upper bound, nonnegative continuous variables

$Z_c^{r,p}$ are introduced to represent the continuous term $OM_p^r \bar{S}_c^{r,p}$. This product can be linearised using the following constraints.

$$S_p^r = \sum_{c=0}^{\mathcal{C}_r} 2^c \bar{S}_c^{r,p}, \qquad \forall r \in R,\ p \in P. \tag{69}$$

$$Z_c^{r,p} \leq \overline{OM}_p^r \bar{S}_c^{r,p}, \qquad \forall r \in R,\ p \in P,\ c \in \{0, 1, \ldots, \mathcal{C}_r\}. \tag{70}$$

$$Z_c^{r,p} \leq OM_p^r, \qquad \forall r \in R,\ p \in P,\ c \in \{0, 1, \ldots, \mathcal{C}_r\}. \tag{71}$$

$$Z_c^{r,p} \geq OM_p^r - \overline{OM}_p^r \left(1 - \bar{S}_c^{r,p}\right), \qquad \forall r \in R,\ p \in P,\ c \in \{0, 1, \ldots, \mathcal{C}_r\}. \tag{72}$$

$$\sum_{r \in R} OM_p^r S_p^r = \sum_{r \in R} \sum_{c=0}^{\mathcal{C}_r} 2^c Z_c^{r,p}, \qquad \forall p \in P. \tag{73}$$

# 6 Numerical experiments

The numerical experiments are implemented in Python 3.9.6 and PyTorch 2.8.0, and the MILP operational model is solved using the Gurobi optimiser version 10.0.1. All computational experiments are performed on a MacBook Pro equipped with an Apple M5 Pro chip and 24 GB of memory, running macOS Tahoe 26.5.1.

## 6.1 Parameter settings

The SAC agents are trained on two test networks. The smaller network contains 4 nodes and 10 links, while the larger network contains 16 nodes and 48 links. Both networks use a time step of 2.5 minutes and a planning horizon of 85 minutes. Nodes 3 and 4 are the possible parking locations in the smaller network, while nodes 4, 6, 9, and 16 are the possible parking locations in the larger network. The smaller network includes two OD trip requests, while the larger network includes three. Each request is divided into three income classes, resulting in 6 and 9 group of trips, respectively. Figure 6 shows the two test networks, and Table 4 summarises the information for the group of trips.

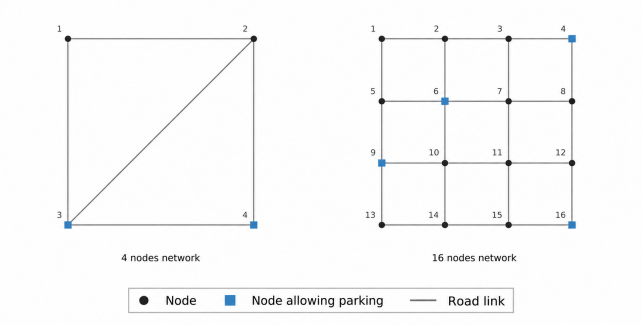



Figure 6: Illustration of grid networks.

**4-node network**

| Group $r$ | $o^r$ | $d^r$ | $a^r$ | $b^r$ | Income level | $n^r$ | $sd^r$ | $st^r$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 4 | 2 | 28 | High | 67 | 4 | 2 |
| 2 | 1 | 4 | 2 | 28 | Middle | 67 | 4 | 2 |
| 3 | 1 | 4 | 2 | 28 | Low | 66 | 4 | 2 |
| 4 | 2 | 4 | 2 | 27 | High | 67 | 2 | 1 |
| 5 | 2 | 4 | 2 | 27 | Middle | 67 | 2 | 1 |
| 6 | 2 | 4 | 2 | 27 | Low | 66 | 2 | 1 |

**16-node network**

| Group $r$ | $o^r$ | $d^r$ | $a^r$ | $b^r$ | Income level | $n^r$ | $sd^r$ | $st^r$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 16 | 4 | 34 | High | 67 | 12 | 6 |
| 2 | 1 | 16 | 4 | 34 | Middle | 67 | 12 | 6 |
| 3 | 1 | 16 | 4 | 34 | Low | 66 | 12 | 6 |
| 4 | 2 | 11 | 3 | 30 | High | 33 | 6 | 3 |
| 5 | 2 | 11 | 3 | 30 | Middle | 33 | 6 | 3 |
| 6 | 2 | 11 | 3 | 30 | Low | 34 | 6 | 3 |
| 7 | 4 | 10 | 0 | 28 | High | 33 | 8 | 4 |
| 8 | 4 | 10 | 0 | 28 | Middle | 33 | 8 | 4 |
| 9 | 4 | 10 | 0 | 28 | Low | 34 | 8 | 4 |

Table 4: Information on the trip requests in the two networks.

The experiments consider two SAV providers, denoted by P1 and P2. The action bounds and model parameters used in the experiments are summarised in Table 5.

| **Parameter** | **Value** |
|---|---|
| Base fare bound | $[0, 50]$ euros |
| Per-kilometre price bound | $[0, 50]$ euros/km |
| Maximum waiting time bound | $[0, 60]$ min |
| Service rate bound | $[0, 1]$ |
| $cf$ (P1) | 1.2 euros/vehicle |
| $cf$ (P2) | 2.5 euros/vehicle |
| $co$ (P1) | 0.32 euros/km |
| $co$ (P2) | 0.60 euros/km |
| Parking nodes (4-node & P1) | $\{3, 4\}$ |
| Parking nodes (4-node & P2) | $\{3, 4\}$ |
| Parking nodes (16-node & P1) | $\{4, 6\}$ |
| Parking nodes (16-node & P2) | $\{4, 6, 9, 16\}$ |
| $\beta_0$ | 0.1 (Fan et al., 2023) |
| $\beta_1$ | 1.0 (Fan et al., 2023) |
| $VOT^r$ (high/middle/low income) | 6.6, 4.6, and 3.8 euros/h (Kolarova et al., 2019) |
| $VOT_B^r$ (high/middle/low income) | 24.9, 17.3, and 14.1 euros/h (Kolarova et al., 2019) |
| $OM_B^r$ | $350/(6 \times 365) \approx 0.160$ euros (Fan et al., 2023) |
| $T_B^r$ | Travel time of bicycles which is calculated by dividing the length of the shortest path by the average speed of the Dutch on a pedal bicycle, 12.4 km/h (Bicycle Dutch, 2018) |

Table 5: Operational and action settings.

Table 6 lists the SAC hyperparameters used in the evaluation, following *Appendix D* of Haarnoja et al. (2018). As part of the training configuration, each episode consists of 10 decision rounds, and the initial random exploration phase lasts for 4,000 decision rounds, equivalent to 400 episodes.

The maximum waiting time produced by the actor network is a continuous variable. For example, the actor may output 4.56 time steps. However, the MILP model in the learning environment can only execute an integer number of time steps. Therefore, the maximum waiting time is rounded during implementation, and 5 time steps are executed in this example. To maintain consistency, the replay buffer and the next state also store the executed integer value. During actor training, changing the output from 4.56 to 4.66 does not change the rounded result. The gradient of the rounding operation is zero in this interval, which prevents the actor from updating through backpropagation. Therefore, following Kim et al. (2021), a straight-through estimator (STE) is used to approximate the derivative of the rounding operation as one during

backpropagation, allowing the actor to continue learning.

| Parameter | Value |
|---|---|
| Optimiser | Adam (Kingma and Ba, 2014) |
| Actor learning rate | $3 \times 10^{-4}$ |
| Critic learning rate | $3 \times 10^{-4}$ |
| Temperature learning rate | $3 \times 10^{-4}$ |
| Discount factor ($\gamma$) | 0.99 |
| Replay buffer size | $10^6$ |
| Number of hidden layers | 2 |
| Number of hidden units per layer | 256 |
| Minibatch size | 256 |
| Entropy target | $-\dim(\mathcal{A}) - \dim(\mathcal{A}) \log 2$ |
| Activation function | ReLU |
| Soft update coefficient ($\rho$) | 0.005 |
| Target update interval | 1 |
| Gradient steps | 1 |

Table 6: SAC hyperparameters.

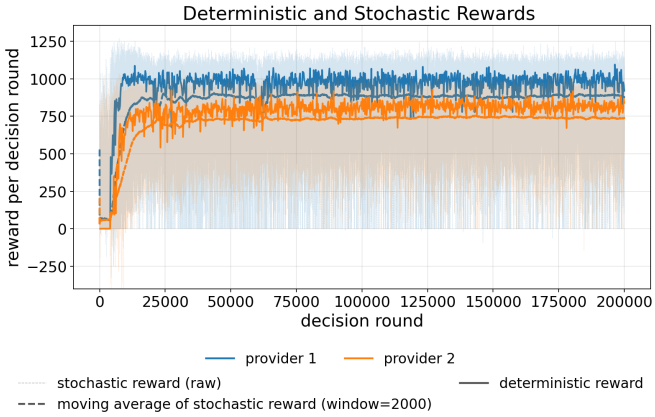
## 6.2 Results

The results are presented in the following order. First, a sensitivity analysis examines the effect of reward scaling on SAC training. Next, a representative SAC run is analysed in detail, followed by a comparison across different random seeds. Finally, the results of the diagonalisation method are presented and compared with the SAC results.

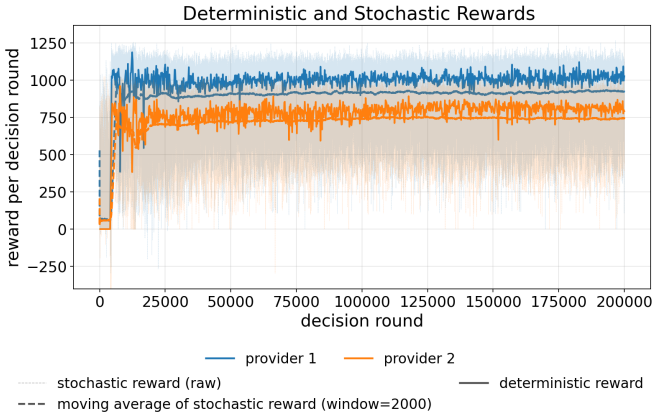
### 6.2.1 Sensitivity analysis on reward scaling

We tested four reward scaling factors: 1, 0.1, 0.01, and 0.001. Figures 7 and 8 present the reward trajectories for the four scaling factors in the 4-node and 16-node cases, respectively. Each experiment was conducted over 20,000 episodes, corresponding to 200,000 decision rounds.

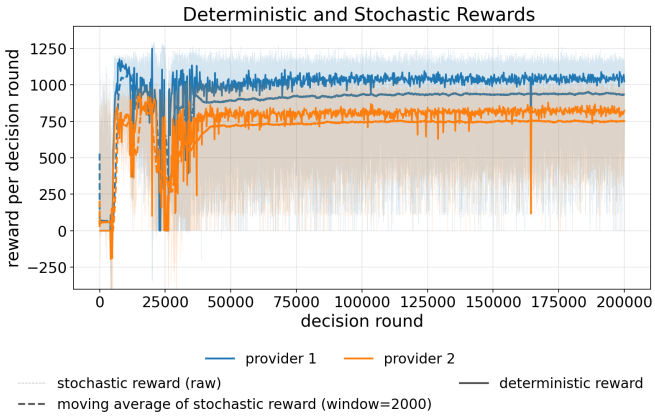
The stochastic reward is the reward returned by the environment for the joint action executed during training. During the first 4,000 decision rounds, the training actions are sampled uniformly from the feasible action ranges. Thereafter, they are sampled from the current stochastic policies. The stochastic reward is recorded after every successfully completed decision round and is represented by the light dashed lines. The thick dashed lines represent the corresponding moving averages over 2,000 decision rounds and show the overall training trends. The deterministic reward is obtained by evaluating the joint actions generated from the means of the current policy distributions without stochastic exploration. This evaluation is performed every 200 decision rounds, and the resulting rewards are represented by the solid lines.




(a) 4-node network with reward scaling 0.001.

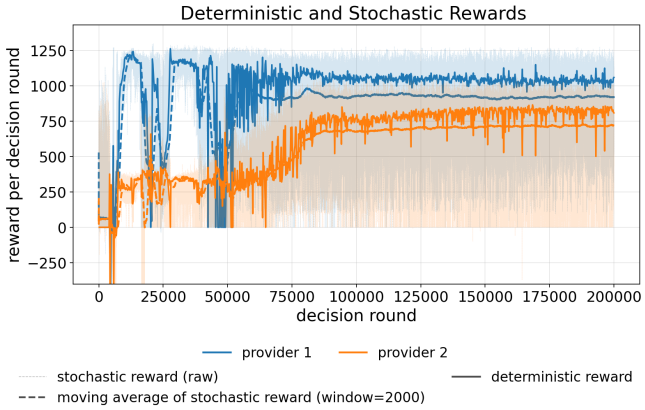



(b) 4-node network with reward scaling 0.01.



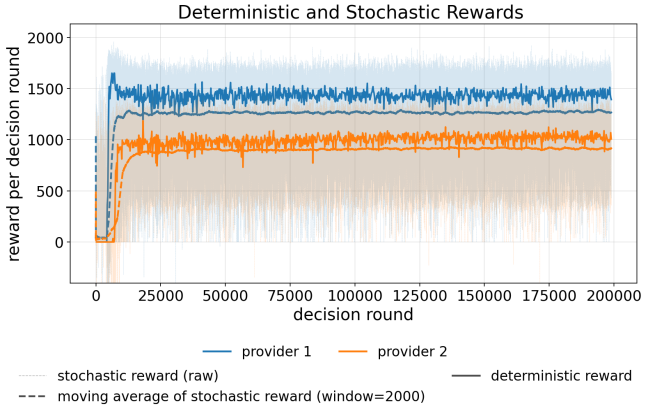

(c) 4-node network with reward scaling 0.1.

Figure 7: Sensitivity analysis of reward scaling for the 4-node network.

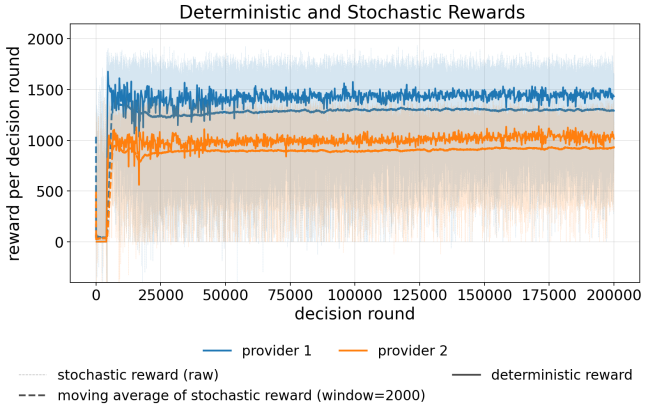



(d) 4-node network with reward scaling 1.

Figure 7: Sensitivity analysis of reward scaling for the 4-node network (continued).

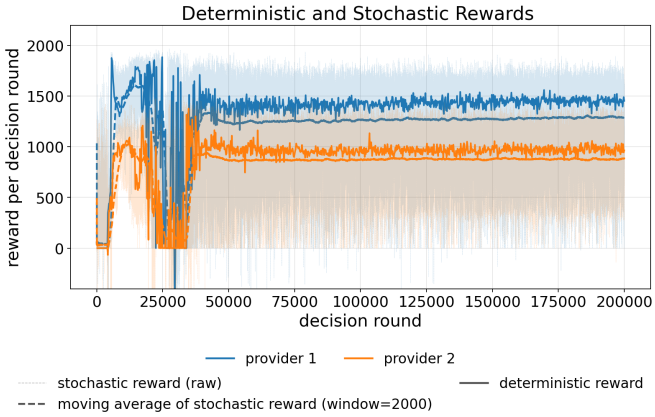



(a) 16-node network with reward scaling 0.001.

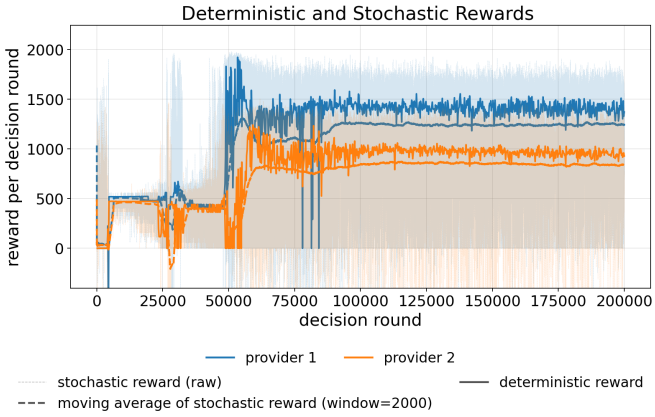



(b) 16-node network with reward scaling 0.01.

Figure 8: Sensitivity analysis of reward scaling for the 16-node network.




(c) 16-node network with reward scaling 0.1.




(d) 16-node network with reward scaling 1.

Figure 8: Sensitivity analysis of reward scaling for the 16-node network (continued).

Across both the 4-node and 16-node cases, reward scales of 0.01 and 0.001 show relatively stable training performance. This may be because the profit obtained by each provider is usually several hundred. Using the original profit directly as the reward therefore produces large target Q-values. Particularly during the early stages of training, this can increase the differences between the critic estimates and their targets, leading to larger and less stable critic updates. The resulting fluctuations in the Q-value estimates may then provide unstable learning signals to the actor network.

Table 7 reports provider-specific training statistics for the final 40,000 decision rounds of the 16-node case. For each network update, the target Q statistic was calculated as the mean absolute target Q-value across the minibatch, while the critic statistic was calculated by averaging the losses of the two critic networks. The table reports the median of each statistic across all updates during these decision rounds.

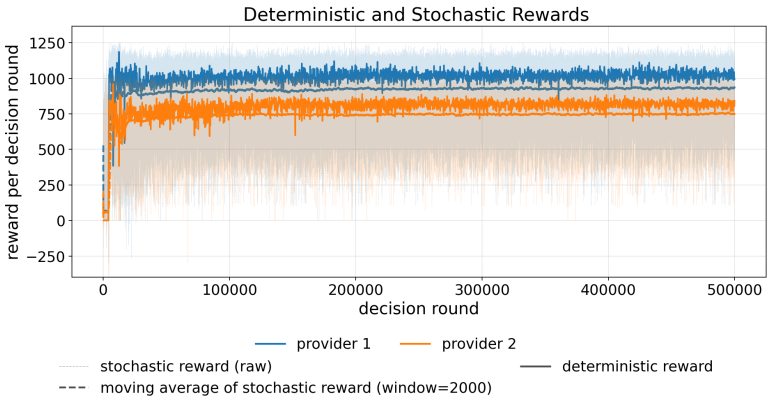
The results show that reward scales of 1 and 0.1 produced large target Q-values and critic losses. Large target Q-values may increase the differences between the critic estimates and their targets, potentially making critic updates less stable. As shown in the figures, reward scales of 0.001 and 0.01 produced relatively stable reward trajectories. A reward scale of 0.01 was used in the subsequent experiments.

| | | Reward scale | | | |
|---|---|---|---|---|---|
| Metric | Provider | 0.001 | 0.01 | 0.1 | 1 |
| Median absolute target Q-value | Provider 1 | 4.39 | 57.23 | 551.48 | 5,295.78 |
| | Provider 2 | 3.09 | 33.86 | 312.84 | 3,010.65 |
| Median critic loss | Provider 1 | 0.73 | 128.77 | 16,519.36 | 2,338,789.06 |
| | Provider 2 | 0.37 | 43.29 | 5,001.31 | 804,458.22 |

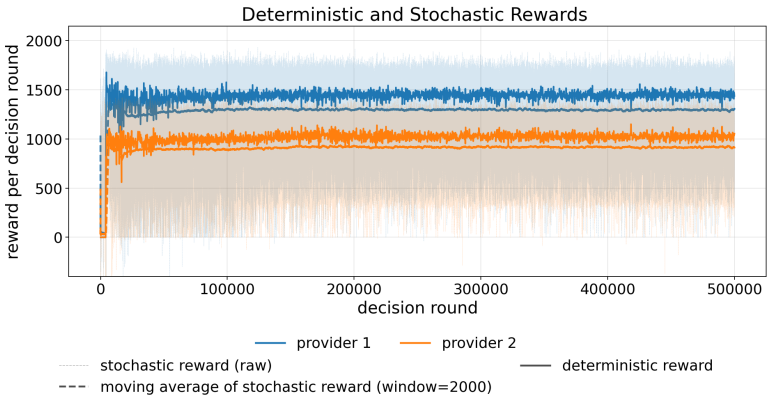
Table 7: Median absolute target Q-values and critic losses during the final 40,000 decision rounds for the 16-node network.

### 6.2.2 SAC results

Figures 9a and 9b show the rewards obtained from stochastic training actions and deterministic evaluation actions for both providers in the 4-node and 16-node settings, respectively. Each experiment was run for 50,000 episodes, corresponding to 500,000 decision rounds. The stochastic rewards were recorded in every decision round, while the deterministic rewards were evaluated and recorded every 200 decision rounds. The raw stochastic rewards fluctuate substantially because the actions are sampled from the current stochastic policies. For each setting, the moving average is calculated separately for each provider using its stochastic rewards over the most recent 2,000 decision rounds. This smoothing reduces short-term fluctuations and reveals the overall reward trends more clearly. The deterministic evaluation rewards are generally higher than the moving averages of the stochastic rewards. Although the overall reward trends become stable, the stochastic rewards continue to fluctuate during the later stage of training. This may be because SAC maintains stochastic policies throughout training while the target entropy remains fixed.



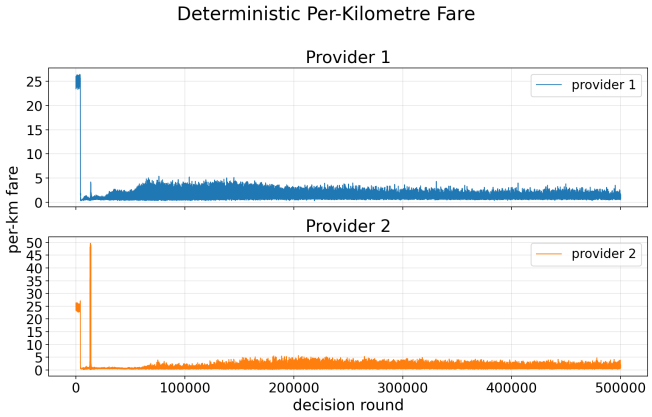

(a) Deterministic and stochastic rewards for the 4-node network.



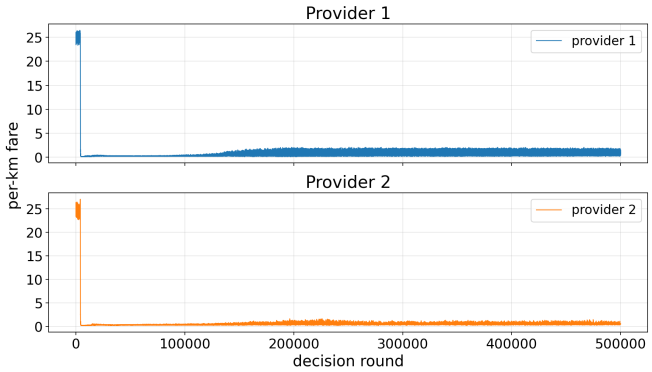

(b) Deterministic and stochastic rewards for the 16-node network.

Figure 9: Deterministic and stochastic rewards during training.

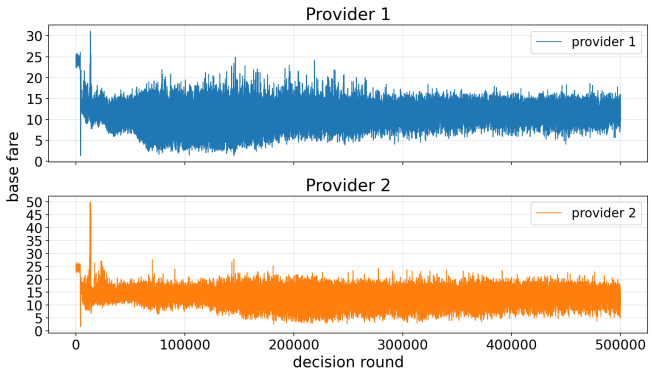
Throughout the following analysis, the deterministic actions were generated from the current policies and recorded every 10 decision rounds. Figure 10 shows the deterministic pricing actions during training under the 4-node and 16-node settings. The distance-based fares remain within narrower ranges after approximately 150,000 to 200,000 decision rounds. The deterministic base fares also settle within relatively stable ranges, but these ranges are wider than those of the distance-based fares. Since the base fares continue to change within these ranges and the training actions are sampled from stochastic policies, the sampled pricing actions remain variable. This pattern is consistent with the fluctuations that remain in the stochastic rewards even during the later stage of training.




(a) Deterministic per-kilometre (distance-based) fare under the 4-node setting.

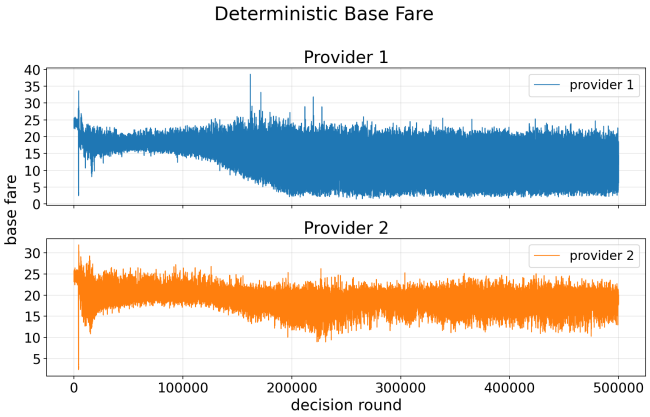



(b) Deterministic per-kilometre (distance-based) fare under the 16-node setting.




(c) Deterministic base fare under the 4-node setting.

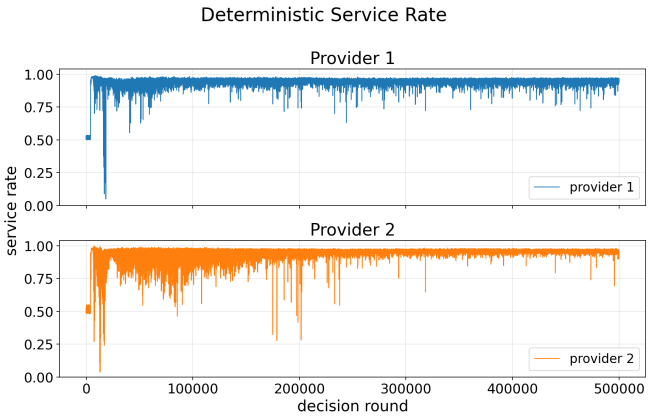
Figure 10: Deterministic pricing actions during training.




(d) Deterministic base fare under the 16-node setting.

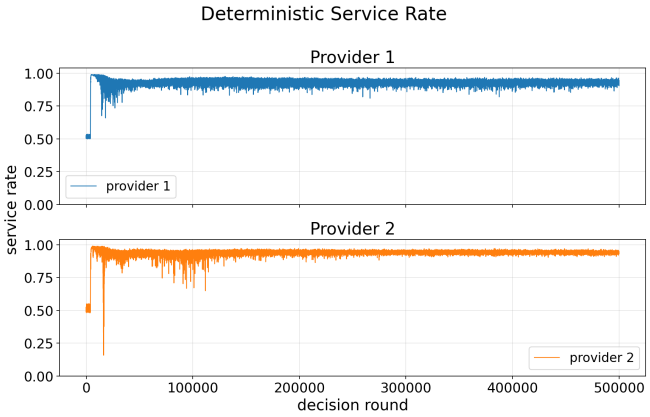
Figure 10: Deterministic pricing actions during training (continued).

Figures 11 and 12 show the deterministic service rates and maximum waiting times, respectively, for both providers under the two network settings. Overall, the service rates produced by the learned policies remain close to 1 with relatively small fluctuations. By comparison, the deterministic maximum waiting times show larger fluctuations. These fluctuations may result from rounding. The maximum waiting time is rounded to an integer before execution. Small changes in the continuous output may cross a rounding threshold, causing the waiting time to switch between adjacent integer values.



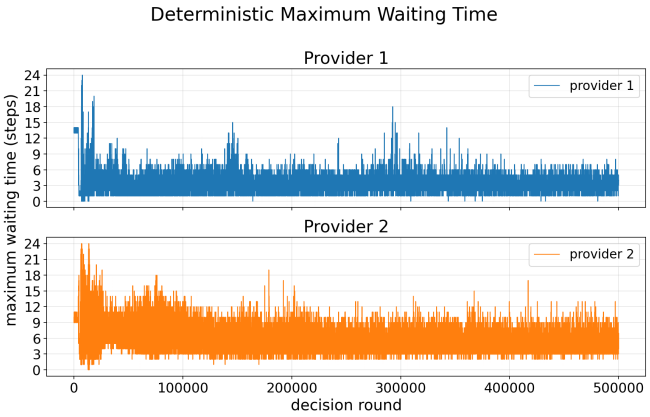

(a) Deterministic service rate under the 4-node setting.

Figure 11: Deterministic service rates during training.

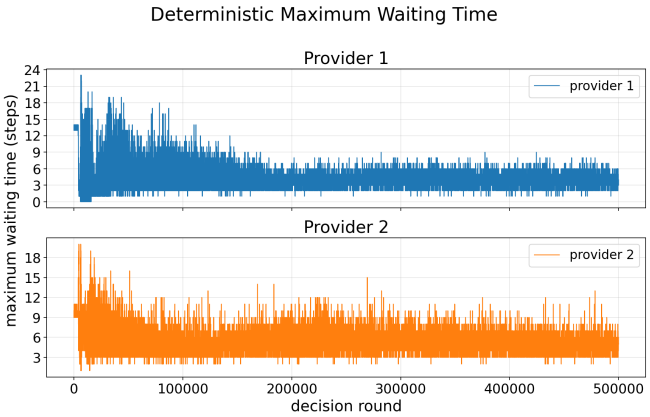



(b) Deterministic service rate under the 16-node setting.

Figure 11: Deterministic service rates during training (continued).



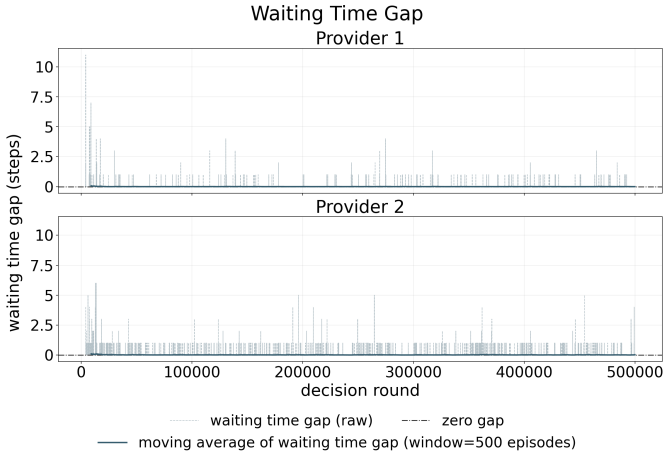

(a) Deterministic maximum waiting time under the 4-node setting.



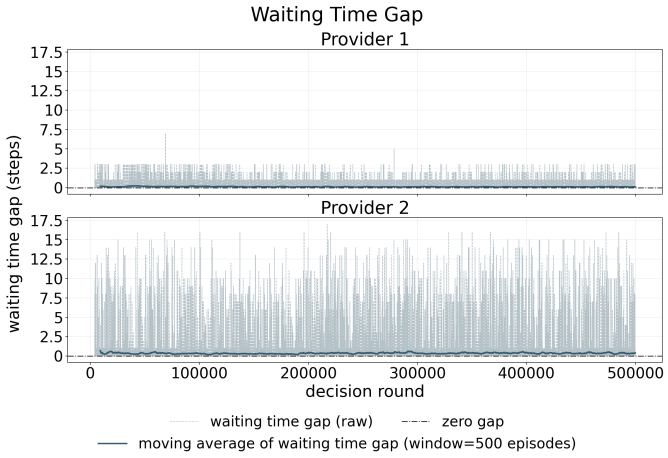

(b) Deterministic maximum waiting time under the 16-node setting.

Figure 12: Deterministic maximum waiting times during training.

In addition to rewards and actions, we examine the difference between the maximum waiting time selected by the stochastic policy and the longest waiting time observed during training. For each served trip group, the realised waiting time is calculated as the longest travel time minus the shortest in-vehicle travel time for the corresponding OD pair. The maximum value across all served groups of trips is then taken as the longest realised waiting time. In Figure 13, for each episode, one raw waiting time gap is recorded by subtracting the longest realised waiting time from the maximum waiting time selected by the provider. The rolling mean is calculated over the most recent 500 episodes to make the overall trend clearer.




(a) Waiting time comparison under the 4-node setting.




(b) Waiting time comparison under the 16-node setting.

Figure 13: Differences between the selected maximum waiting times and the longest realised waiting times during training.

Although the dense raw lines make the gaps appear to be frequently positive, they are zero in most episodes. Under the 4-node setting, the gaps are zero in 99.7% of episodes for Provider 1 and 99.0% for Provider 2. Under the 16-node setting, the corresponding percentages are 89.8% and 84.6%. Positive waiting-time gaps may occur because the maximum waiting time is sampled stochastically and acts as an upper bound in the operational optimisation model. Consequently, the sampled maximum waiting time may exceed the realised waiting time produced by the operational solution.

The deterministic action plots show that some actions, particularly the maximum waiting time and base fare, continued to fluctuate. To quantify these changes, the deterministic joint actions recorded for the final 200 episodes were analysed. One deterministic joint action was recorded every 10 completed decision rounds, corresponding to one record per episode. The base fare and distance-based fare were considered stable when the absolute difference between the medians of the first 100 and last 100 records did not exceed 0.1, while the corresponding threshold for the service rate was 0.05. The maximum waiting time was considered stable when the modal integer value occurred in at least 190 of the 200 records (95%) and the difference between the maximum and minimum values did not exceed one time step.

Table 8 presents the stability assessment results. Of the four action variables, only the service rate met the stability thresholds for both providers under both network settings. Future work could apply entropy annealing to examine whether gradually reducing the target entropy leads to more stable deterministic actions and stochastic rewards.

| | | 4-node network | | 16-node network | |
|---|---|---|---|---|---|
| Action variable | Stability criterion | P1 | P2 | P1 | P2 |
| Base fare | Median difference $\leq 0.1$ | 0.421 (Fail) | 0.402 (Fail) | 2.451 (Fail) | 0.296 (Fail) |
| Distance-based fare | Median difference $\leq 0.1$ | 0.005 (Pass) | 0.117 (Fail) | 0.269 (Fail) | 0.012 (Pass) |
| Service rate | Median difference $\leq 0.05$ | 0.0018 (Pass) | 0.0006 (Pass) | 0.0027 (Pass) | 0.0015 (Pass) |
| Maximum waiting time | Modal frequency $\geq 190/200$ | 83 (Fail) | 69 (Fail) | 108 (Fail) | 96 (Fail) |
| Maximum waiting time | Difference $\leq 1$ time step | 5 (Fail) | 8 (Fail) | 4 (Fail) | 5 (Fail) |

Table 8: Stability criteria and results.

### 6.2.3 Sensitivity analysis on random seeds

The random seed affects several stochastic components of the training process. In our experiments, these components include (1) neural network initialisation, which results in different initial weights and biases for each provider's actor and critic networks; (2) the initial shared state, which is constructed from a different initial joint action at the beginning of each episode; (3) early-stage random exploration, which generates different fare, maximum waiting time, and service rate decisions; (4) the Gaussian noise used to sample actions from the stochastic SAC policies; and (5) mini-batch sampling from the replay buffer during network updates. Consequently, different random seeds may lead to variations in training stability, convergence speed, and policy performance (Henderson et al., 2018). We therefore repeat the experiment over 20,000 episodes on both networks using three different random seeds to examine whether the results remain consistent across different random seeds.

After 20,000 training episodes, the learned policies are evaluated using deterministic actions to obtain representative final actions and rewards. At the beginning of the evaluation, a joint action is randomly generated and used as the initial state. Both providers then follow deterministic policies, where each agent directly outputs the mean of its policy distribution without random exploration. This process continues for 10 evaluation rounds. The deterministic policy evaluation is continued for 10 rounds because the state is defined as the joint action from the previous round. Consequently, the action and reward in the first evaluation round may be affected by the randomly generated initial state. Repeated evaluation allows this initial-state effect to decrease as each deterministic joint action becomes the state of the following round.

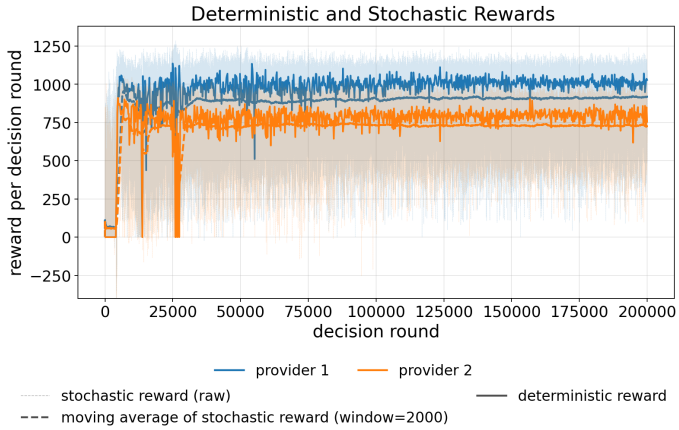
The deterministic policy evaluation results from the final round of each of the three independent runs are reported in Table 9. Different random seeds produced some variation in the final action profiles, but the resulting profits were broadly comparable across the three runs. Thus, the learned solutions remain sensitive to the random seed while achieving similar overall performance.

The service rate values reported in Table 9 represent the minimum service rates selected by the providers rather than the realised proportions of demand served. Although the selected service rates range from 0.92 to 0.96, the operational model serves all assigned demand in every evaluated solution, resulting in a realised service rate of 1.0. This difference arises because Constraint (8) requires the realised service rate to be greater than or equal to the service rate selected by the provider. When serving additional trips is operationally feasible and profitable, the model may serve more demand than required by the selected service rate.

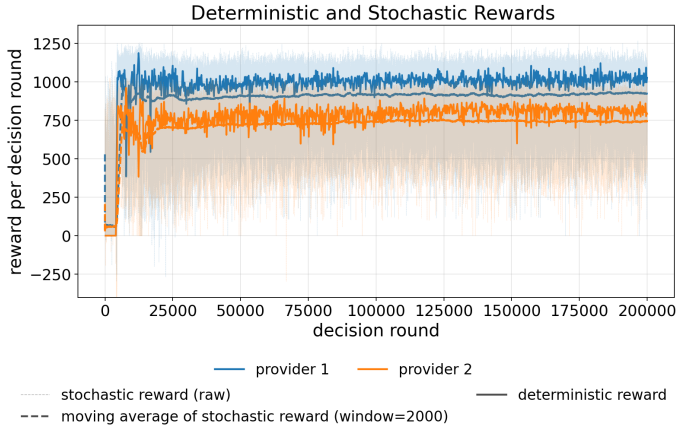
The stochastic and deterministic policy rewards during training are shown in Figures 14 and 15 for the two network studies, respectively. For the 4-node network, the learning curves differ across random seeds during the early stages of training but gradually converge towards similar reward levels. For the 16-node network, however, the learning curve for Seed 0 exhibits greater variability than those for Seeds 1 and 2.

| Network | Seed | Provider | Base fare | Per-km fare | Max waiting time | Service rate | Profit | Served/ Demand |
|---|---|---|---|---|---|---|---|---|
| 4-node | 0 | P1 | 13.51 | 0.84 | 3 | 0.95 | 1052.10 | 76/76 |
| 4-node | 0 | P2 | 14.34 | 0.92 | 7 | 0.96 | 824.48 | 63/63 |
| 4-node | 1 | P1 | 12.37 | 1.07 | 2 | 0.95 | 1032.52 | 79/79 |
| 4-node | 1 | P2 | 17.67 | 0.69 | 2 | 0.96 | 798.88 | 54/54 |
| 4-node | 2 | P1 | 9.62 | 1.58 | 6 | 0.96 | 990.98 | 81/81 |
| 4-node | 2 | P2 | 14.95 | 0.93 | 3 | 0.94 | 819.73 | 61/61 |
| 16-node | 0 | P1 | 16.00 | 0.29 | 9 | 0.93 | 1354.15 | 108/108 |
| 16-node | 0 | P2 | 3.45 | 1.95 | 7 | 0.92 | 948.61 | 79/79 |
| 16-node | 1 | P1 | 17.37 | 0.47 | 5 | 0.92 | 1384.26 | 89/89 |
| 16-node | 1 | P2 | 19.09 | 0.46 | 5 | 0.94 | 1058.90 | 79/79 |
| 16-node | 2 | P1 | 16.33 | 0.41 | 5 | 0.94 | 1445.32 | 104/104 |
| 16-node | 2 | P2 | 16.59 | 0.76 | 5 | 0.94 | 983.94 | 71/71 |

Table 9: Deterministic policy evaluation results across random seeds for the 4-node and 16-node networks.

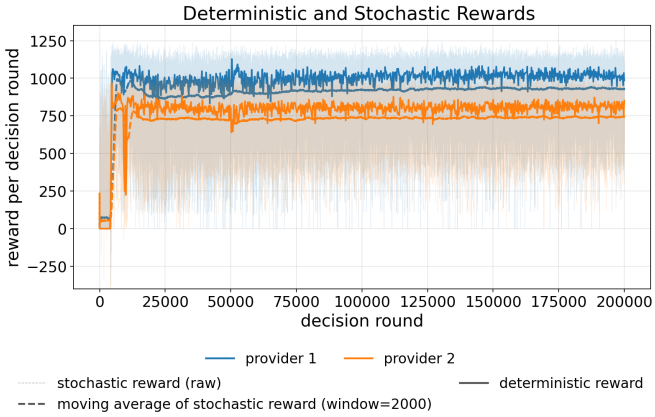



(a) 4-node network, Seed 0.



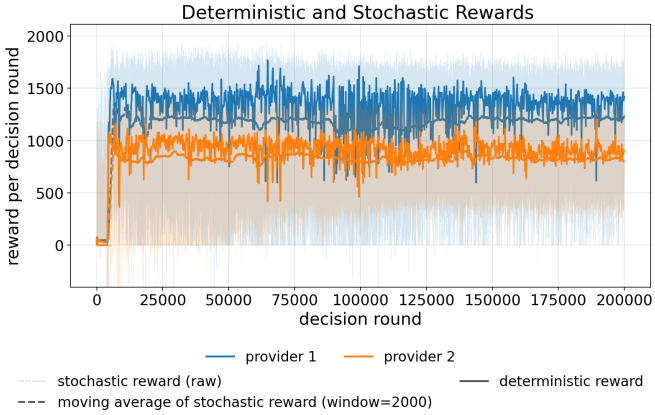

(b) 4-node network, Seed 1.

Figure 14: Sensitivity analysis on random seeds for the 4-node network.

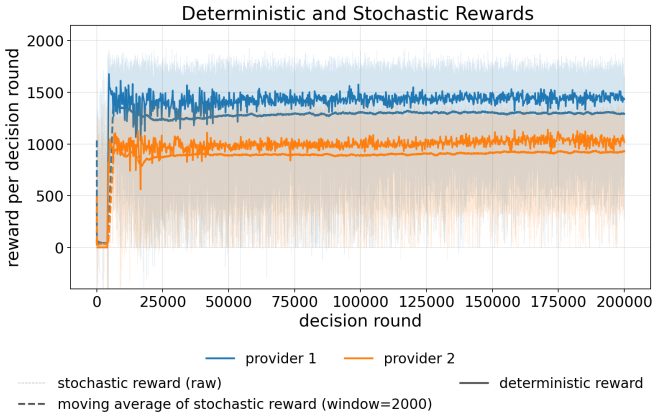



(c) 4-node network, Seed 2.

Figure 14: Sensitivity analysis on random seeds for the 4-node network (continued).

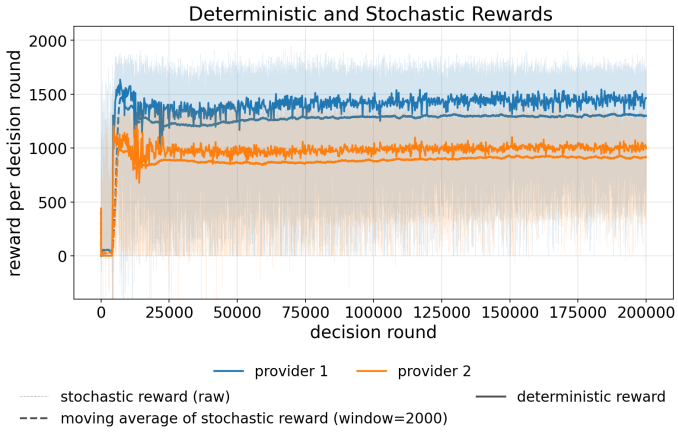



(a) 16-node network, Seed 0.




(b) 16-node network, Seed 1.

Figure 15: Sensitivity analysis on random seeds for the 16-node network.




(c) 16-node network, Seed 2.

Figure 15: Sensitivity analysis on random seeds for the 16-node network (continued).

### 6.2.4 Diagonalisation results

The diagonalisation method is initialised by solving the problem for Provider 1 under the assumption that no other SAV provider is operating in the market. Therefore, Provider 1 initially competes only with the bicycle alternative. The resulting decisions of Provider 1 are then treated as fixed competitor decisions when solving the problem for Provider 2. After this initialisation, the providers update their decisions sequentially through successive best responses until the convergence conditions are satisfied. To examine the effect of the initialisation order, the order of the two providers was reversed. The resulting actions, demand allocation, and profits were nearly identical, indicating that the diagonalisation solution is not sensitive to the initialisation order in this case.

The diagonalisation method converged in the fourth iteration for both network settings. Action convergence was assessed using the Euclidean distance between each provider's action vectors in two successive iterations, with a tolerance of $10^{-2}$. Table 10 summarises the detailed results for each iteration of the diagonalisation method. Table 11 summarises the operational results at the converged solution. The MIP gap represents the relative difference between the best feasible objective value and the best objective bound when the solver terminates. Gurobi's default relative MIP gap tolerance of $10^{-4}$, equivalent to 0.01%, was used (Gurobi Optimization, LLC, 2026). All reported MIP gaps were below this tolerance, indicating that the final solutions were close to optimal for the corresponding best-response problems. The total method time denotes the total running time required to complete the entire diagonalisation procedure for each network.

| Iteration | Provider | Base fare | Distance fare | Service rate | Maximum waiting time | MIP gap | Profit (euros) |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 12.658786 | 0.919534 | 1.000 | 0 | 0.0053% | 1112.728258 |
| 1 | 2 | 14.215092 | 0.921628 | 1.000 | 3 | 0.0073% | 889.241646 |
| 2 | 1 | 12.044689 | 1.217210 | 1.000 | 0 | 0.0000 | 1097.084315 |
| 2 | 2 | 14.062216 | 0.995628 | 1.000 | 3 | 0.0062% | 894.228347 |
| 3 | 1 | 12.014207 | 1.231965 | 1.000 | 0 | 0.0048% | 1098.270662 |
| 3 | 2 | 14.054787 | 0.999217 | 1.000 | 3 | 0.0067% | 894.468994 |
| 4 | 1 | 12.012718 | 1.232679 | 1.000 | 0 | 0.0095% | 1098.326887 |
| 4 | 2 | 14.054424 | 0.999390 | 1.000 | 3 | 0.0086% | 894.480381 |

(a) Results of the diagonalisation method for the 4-node network.

| Iteration | Provider | Base fare | Distance fare | Service rate | Maximum waiting time | MIP gap | Profit (euros) |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 15.304532 | 0.668838 | 1.000 | 0 | 0.0000 | 1594.928540 |
| 1 | 2 | 16.377351 | 0.748399 | 1.000 | 3 | 0.0099% | 1124.926136 |
| 2 | 1 | 14.856194 | 0.715270 | 1.000 | 0 | 0.0000 | 1610.506249 |
| 2 | 2 | 16.243005 | 0.762914 | 1.000 | 3 | 0.0091% | 1125.503839 |
| 3 | 1 | 14.821205 | 0.719616 | 1.000 | 0 | 0.0095% | 1611.290496 |
| 3 | 2 | 16.233673 | 0.764074 | 1.000 | 3 | 0.0085% | 1125.666748 |
| 4 | 1 | 14.818334 | 0.720020 | 1.000 | 0 | 0.0094% | 1611.403684 |
| 4 | 2 | 16.232915 | 0.764179 | 1.000 | 3 | 0.0080% | 1125.688656 |

(b) Results of the diagonalisation method for the 16-node network.

Table 10: Results of the diagonalisation method: (a) the 4-node network and (b) the 16-node network.

**4-node network**

| Result | Provider 1 | Provider 2 |
|---|---|---|
| Total demand | 85 | 70 |
| Total served demand | 85 | 70 |
| Fleet size | 85 | 36 |
| Total revenue | 1336.647 | 1195.680 |
| Depreciation cost | 102.000 | 90.000 |
| Operational cost | 136.320 | 211.200 |
| Profit | 1098.327 | 894.480 |
| MIP gap (%) | 0.0095294 | 0.0086236 |
| Choice probability | 0.205582 | 0.168510 |
| Total method time (min) | 24.336 | |

**16-node network**

| Result | Provider 1 | Provider 2 |
|---|---|---|
| Total demand | 106 | 83 |
| Total served demand | 106 | 83 |
| Fleet size | 106 | 63 |
| Total revenue | 2315.244 | 1964.789 |
| Depreciation cost | 127.200 | 157.500 |
| Operational cost | 576.640 | 681.600 |
| Profit | 1611.404 | 1125.689 |
| MIP gap (%) | 0.0093920 | 0.0080218 |
| Choice probability | 0.255487 | 0.199037 |
| Total method time (min) | 73.526 | |

Table 11: Operational results and computation times at the converged diagonalisation solutions.

### 6.2.5 Comparison of the SAC and diagonalisation methods

Before comparing the performance of the two methods, we first examine whether their evaluation environments are consistent. For this purpose, the joint action profile consisting of Provider 1's strategy from iteration 4 and Provider 2's strategy from iteration 3, obtained using the diagonalisation method, was evaluated directly in the SAC environment. The results show slight differences between the profits calculated in the two evaluation environments, as the exact and approximated choice probabilities may result in different integer allocations of passenger demand. To illustrate this more clearly, Table 12 uses Provider 1 at iteration 4 of the 16-node case as an example and presents the choice probabilities and integer demand allocations produced by the two methods. Specifically, the table reports the exact MNL probabilities and corresponding integer demands computed in the SAC environment, alongside the piecewise-linear probability approximations and integer demands obtained from the diagonalisation model. For trip groups 1, 6, and 8, the exact expected MNL demand lies below the rounding threshold, whereas the approximated expected demand reaches the threshold and is rounded upward. Consequently, the SAC environment assigns one fewer trip to Provider 1 in each of these three request groups. Provider 1's total demand is therefore 103 trips in the SAC environment, compared with 106 trips in the diagonalisation model.

Apart from the differences in choice probabilities and demand allocation, when the same action profile and demand allocation used in the diagonalisation model were directly provided as inputs to the SAC environment, the resulting profits were identical. This confirms that the two methods use consistent operational models and profit calculations.

| **Group** | $n^r$ | **Exact MNL probability** | **Exact** $n_p^r$ | **SAC demand** | **Approximated probability** | **Approximated** $n_p^r$ | **Diagonalisation demand** |
|---|---|---|---|---|---|---|---|
| 1 | 67 | 0.361928074 | 24.249181 | 24 | 0.365671642 | 24.500000 | 25 |
| 2 | 67 | 0.261696826 | 17.533687 | 18 | 0.261194030 | 17.500000 | 18 |
| 3 | 66 | 0.220064857 | 14.524281 | 15 | 0.219696970 | 14.500000 | 15 |
| 4 | 33 | 0.258753136 | 8.538853 | 9 | 0.257988940 | 8.513635 | 9 |
| 5 | 33 | 0.209489493 | 6.913153 | 7 | 0.207224484 | 6.838408 | 7 |
| 6 | 34 | 0.190004075 | 6.460139 | 6 | 0.191176471 | 6.500000 | 7 |
| 7 | 33 | 0.293515964 | 9.686027 | 10 | 0.291033039 | 9.604090 | 10 |
| 8 | 33 | 0.226426191 | 7.472064 | 7 | 0.227272727 | 7.500000 | 8 |
| 9 | 34 | 0.199774131 | 6.792320 | 7 | 0.198193464 | 6.738578 | 7 |
| **Total** | **400** | – | – | **103** | – | – | **106** |

Table 12: Comparison of exact and approximated demand for Provider 1 in iteration 4 of the 16-node case.

Following this consistency check, the SAC and diagonalisation methods are compared from three perspectives: (1) profit performance, by comparing the profits of the deterministic solutions obtained from the three SAC training seeds with the converged profits produced by the diagonalisation method; (2) market-

equilibrium analysis, by using the deterministic joint action obtained from the deterministic evaluation of each SAC training seed as the initial solution for the diagonalisation method and determining whether either provider can increase its profit through a best-response update while the other provider's action is held fixed, as the diagonalisation method can be regarded as a procedure for searching for a stable market solution; and (3) computational efficiency, by comparing the computational times of the two methods.

Table 13 presents the deterministic policy evaluation results obtained using different random seeds alongside the converged results of the diagonalisation method. In both network settings, the diagonalisation method produces higher profits for each provider than those obtained from all three deterministic policy evaluations. The main differences are that the diagonalisation method selects shorter maximum waiting times and service rates of 1.00 for both providers.

| Network | Method | Run | Provider | Base fare | Per-km fare | Maximum waiting time | Service rate | Demand | Profit (euros) |
|---|---|---|---|---|---|---|---|---|---|
| 4-node | SAC | Seed 0 | P1 | 13.51 | 0.84 | 3 | 0.95 | 76 | 1052.100 |
| 4-node | SAC | Seed 0 | P2 | 14.34 | 0.92 | 7 | 0.96 | 63 | 824.480 |
| 4-node | SAC | Seed 1 | P1 | 12.37 | 1.07 | 2 | 0.95 | 79 | 1032.520 |
| 4-node | SAC | Seed 1 | P2 | 17.67 | 0.69 | 2 | 0.96 | 54 | 798.880 |
| 4-node | SAC | Seed 2 | P1 | 9.62 | 1.58 | 6 | 0.96 | 81 | 990.980 |
| 4-node | SAC | Seed 2 | P2 | 14.95 | 0.93 | 3 | 0.94 | 61 | 819.730 |
| 4-node | Diagonalisation | Iteration 4 | P1 | 12.01 | 1.23 | 0 | 1.00 | 85 | 1098.327 |
| 4-node | Diagonalisation | Iteration 4 | P2 | 14.05 | 1.00 | 3 | 1.00 | 70 | 894.480 |
| 16-node | SAC | Seed 0 | P1 | 16.00 | 0.29 | 9 | 0.93 | 108 | 1354.150 |
| 16-node | SAC | Seed 0 | P2 | 3.45 | 1.95 | 7 | 0.92 | 79 | 948.610 |
| 16-node | SAC | Seed 1 | P1 | 17.37 | 0.47 | 5 | 0.92 | 89 | 1384.260 |
| 16-node | SAC | Seed 1 | P2 | 19.09 | 0.46 | 5 | 0.94 | 79 | 1058.900 |
| 16-node | SAC | Seed 2 | P1 | 16.33 | 0.41 | 5 | 0.94 | 104 | 1445.320 |
| 16-node | SAC | Seed 2 | P2 | 16.59 | 0.76 | 5 | 0.94 | 71 | 983.940 |
| 16-node | Diagonalisation | Iteration 4 | P1 | 14.82 | 0.72 | 0 | 1.00 | 106 | 1611.404 |
| 16-node | Diagonalisation | Iteration 4 | P2 | 16.23 | 0.76 | 3 | 1.00 | 83 | 1125.689 |

Table 13: Comparison of the deterministic policy evaluation results obtained from three SAC training seeds with the converged diagonalisation solutions.

One possible explanation for why the learned policies do not produce solutions with higher profits is that deterministic policy evaluation uses the means of the learned policy distributions rather than explicitly searching for actions associated with higher profits. As noted by Haarnoja et al. (2018), evaluating a policy using the mean action generally results in a higher return, but the mean action does not necessarily correspond to the optimal action for the maximum-return objective.

Taking the 4-node network with Seed 0 as an example, we conducted a test in which the service rate of each provider was separately increased to 1, while all other actions were held fixed. The profit increased from 1052.104 to

1066.027 for Provider 1 and from 824.477 to 838.266 for Provider 2. In addition, an examination of the sampled actions showed that the stochastic policies had explored this higher service rate range. Specifically, during the final 10,000 decision rounds, approximately 13.7% and 12.2% of the service rate actions sampled by Providers 1 and 2, respectively, were between 0.98 and 0.99. This shows that increasing the service rate can lead to higher profits and that the agents explored this range during training. However, the stochastic policies continued to sample a range of service rate actions, including lower values. Consequently, the mean service rates used for deterministic evaluation remained below the higher service rates explored during training. Thus, the actions generated during deterministic policy evaluation did not necessarily produce higher realised profits.

The deterministic solutions obtained from the three SAC training seeds were subsequently used separately to initialise the diagonalisation method. For both the 4-node and 16-node networks, the best-response iterations converged to the corresponding diagonalisation solutions, regardless of which seed solution was used for initialisation. This shows that the diagonalisation method can further improve the strategies of both providers and identify the corresponding stable market solutions.

For the computation time comparison, we examine the time required by the two methods to obtain stable results. For the diagonalisation method, the computation time is measured from the start of the method until the best-response iterations satisfy the predefined convergence criteria. Although SAC did not satisfy the previously defined stability criteria for deterministic actions within 500,000 decision rounds, Figure 9 shows that the deterministic rewards had largely stabilised after approximately 50,000 decision rounds. Moreover, the deterministic policy evaluations conducted after 50,000 and 500,000 decision rounds produced similar results, indicating that the additional training provided only limited improvement in evaluation performance. The computation time for 50,000 decision rounds is therefore used as a practical estimate of the time taken for the reward performance of SAC to stabilise. To reduce the effect of random variation, the SAC computation time was averaged across the three runs with different random seeds for each network. For the network with 4 nodes, SAC took approximately 35 minutes on average to complete 50,000 decision rounds, whereas the diagonalisation method took approximately 24 minutes to converge. For the network with 16 nodes, the corresponding computation times were approximately 290 minutes for SAC and 73 minutes for the diagonalisation method. Under this comparison criterion, the diagonalisation method was more computationally efficient than SAC for both network sizes.

# 7 Conclusion and future research

This thesis studied the pricing and service decisions of SAV providers in a competitive multi-provider market. Each provider determines its base fare, distance-based fare, maximum waiting time, and minimum service rate while considering the decisions of the other providers, with the aim of increasing its own profit. In the proposed framework, a multinomial logit model is used to describe passenger choices among the different SAV providers and the bicycle alternative and to calculate the passenger demand allocated to each provider. Once the passenger demand allocated to each provider has been determined, a mixed-integer linear programming model is used to determine the corresponding fleet size, ride matching decisions, vehicle parking decisions, and resulting profit.

To solve this problem, two approaches were considered: a diagonalisation method and a multi-agent SAC method. The diagonalisation method updates the decisions of the providers through successive best responses, whereas in the learning-based approach, the providers learn their policies through repeated interactions with the market environment. Numerical experiments compared the two methods from three perspectives: profit performance, market stability, and computational efficiency. In the tested cases, the profits obtained using SAC were close to, but generally lower than, those obtained using the diagonalisation method. When the SAC solutions were used to initialise the diagonalisation method, the providers were still able to increase their profits through best-response updates, indicating that the SAC solutions had not reached stable market states. In contrast, the diagonalisation method converged to stable market solutions and required less computation time than SAC in both network settings. The comparison also showed the importance of using consistent demand-allocation and rounding methods, as even small differences in allocated demand may affect operational feasibility and profits. The results indicate that reinforcement learning has potential for joint pricing and service planning, but further improvements are needed to obtain more stable and higher-profit solutions.

For future research, we propose: (1) examining whether the learning process of multiple providers can reach a market equilibrium; (2) applying entropy annealing to reduce policy fluctuations during the later stages of training; and (3) incorporating traffic congestion into the model.

# References


[1] Ben-Akiva, Moshe E. and Lerman, Steven R. *Discrete Choice Analysis: Theory and Application to Travel Demand*. MIT Press, 1985.

[2] Bicycle Dutch. *Dutch Cycling Figures*. https://bicycledutch.wordpress.com/2018/01/02/dutch-cycling-figures/. Accessed: 15 September 2026. 2018.

[3] Bruglieri, Maurizio, Peruzzini, Roberto, and Pisacane, Ornella. "Efficiently routing a fleet of autonomous vehicles in a real-time ride-sharing system". In: *Computers & Operations Research* 168 (2024), p. 106668.

[4] Chen, Chuqiao et al. "Competitive pricing for ride-sourcing platforms with MARL". In: *Transportation Research Part C: Emerging Technologies* 165 (2024), p. 104697.

[5] Chen, Yao and Liu, Yang. "Integrated optimization of planning and operations for shared autonomous electric vehicle systems". In: *Transportation Science* 57.1 (2023), pp. 106–134.

[6] Fan, Qiaochu, Essen, J. Theresia van, and Correia, Gonçalo H. A. "Optimising fleet sizing and management of shared automated vehicle (SAV) services: A mixed-integer programming approach integrating endogenous demand, congestion effects, and accept/reject mechanism impacts". In: *Transportation Research Part C: Emerging Technologies* 157 (2023), p. 104398.

[7] Feng, Chi and Mei, Zhenyu. "Optimization of Shared Autonomous Vehicles Routing Problem: From the View of Parking". In: *Sustainability* 15.16 (2023), p. 12303.

[8] Grondman, Ivo et al. "A survey of actor-critic reinforcement learning: Standard and natural policy gradients". In: *IEEE Transactions on Systems, Man, and Cybernetics, part C (applications and reviews)* 42.6 (2012), pp. 1291–1307.

[9] Guo, Hao, Chen, Yao, and Liu, Yang. "Shared autonomous vehicle management considering competition with human-driven private vehicles". In: *Transportation research part C: emerging technologies* 136 (2022), p. 103547.

[10] Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*. Accessed: 1 October 2026. 2026. URL: https://docs.gurobi.com/projects/optimizer/en/current/reference/parameters.html.

[11] Haarnoja, Tuomas et al. "Soft actor-critic algorithms and applications". In: *arXiv preprint arXiv:1812.05905* (2018).

[12] Haarnoja, Tuomas et al. *Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor*. 2018. arXiv: 1801.01290 [cs.LG]. URL: https://arxiv.org/abs/1801.01290.

[13] Henderson, Peter et al. "Deep reinforcement learning that matters". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 32. 1. 2018.

[14] Horváth, Dániel et al. “HiER: Highlight experience replay for boosting off-policy reinforcement learning agents”. In: *IEEE Access* 12 (2024), pp. 100102–100119.

[15] Kim, Dohyung, Lee, Junghyup, and Ham, Bumsub. “Distance-Aware Quantization”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 5271–5280.

[16] Kingma, Diederik P and Ba, Jimmy. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[17] Kolarova, Viktoriya, Steck, Felix, and Bahamonde-Birke, Francisco J. “Assessing the effect of autonomous driving on value of travel time savings: A comparison between current and future preferences”. In: *Transportation Research Part A: Policy and Practice* 129 (2019), pp. 155–169.

[18] Lai, Zhijie and Li, Sen. “Spatiotemporal pricing and fleet management of autonomous mobility-on-demand networks: A decomposition and dynamic programming approach with bounded optimality gap”. In: *IEEE Transactions on Intelligent Transportation Systems* 25.7 (2023), pp. 7057–7069.

[19] LeCun, Yann, Bengio, Yoshua, and Hinton, Geoffrey. “Deep learning”. In: *nature* 521.7553 (2015), pp. 436–444.

[20] Li, Qing et al. “A duopoly competition problem of shared autonomous vehicles in a multimodal transportation system with government regulation”. In: *Multimodal Transportation* 4.4 (2025), p. 100241.

[21] Li, Qing et al. “Fleet sizing and pricing for hybrid ownership of shared autonomous vehicles in a multimodal transportation system”. In: *Transportation Research Part E: Logistics and Transportation Review* 193 (2025), p. 103871.

[22] Liu, Ao et al. “Joint optimal pricing strategy of shared autonomous vehicles and road congestion pricing: A regional accessibility perspective”. In: *Cities* 146 (2024), p. 104742.

[23] Lu, Ying et al. “Deep reinforcement learning-based charging pricing for autonomous mobility-on-demand system”. In: *IEEE Transactions on Smart Grid* 13.2 (2021), pp. 1412–1426.

[24] Najmi, Ali, Rashidi, Taha H, and Waller, Travis. “A multimodal multi-provider market equilibrium model: A game-theoretic approach”. In: *Transportation Research Part C: Emerging Technologies* 146 (2023), p. 103959.

[25] Shannon, Claude E. “A Mathematical Theory of Communication”. In: *The Bell System Technical Journal* 27.4 (1948), pp. 623–656. DOI: 10.1002/j.1538-7305.1948.tb00917.x.

[26] Sun, Guolin et al. “Dynamic pricing for vehicle dispatching in mobility-as-a-service market via multi-agent deep reinforcement learning”. In: *IEEE Transactions on Vehicular Technology* 73.8 (2024), pp. 12010–12025.

[27] Turan, Berkay, Pedarsani, Ramtin, and Alizadeh, Mahnoosh. "Dynamic pricing and fleet management for electric autonomous mobility on demand systems". In: *Transportation Research Part C: Emerging Technologies* 121 (2020), p. 102829.

[28] Wang, David ZW, Liu, Haoxiang, and Szeto, Wai Y. "A novel discrete network design problem formulation and its global optimization solution algorithm". In: *Transportation Research Part E: Logistics and Transportation Review* 79 (2015), pp. 213–230.

[29] Watkins, Christopher JCH and Dayan, Peter. "Q-learning". In: *Machine learning* 8 (1992), pp. 279–292.

[30] Williams, Ronald J. "Simple statistical gradient-following algorithms for connectionist reinforcement learning". In: *Machine learning* 8 (1992), pp. 229–256.

[31] Yang, Shuang et al. "Integrated optimization of pricing and relocation in the competitive carsharing market: A multi-leader-follower game model". In: *Transportation Research Part C: Emerging Technologies* 138 (2022), p. 103613.