Can Dynamic Pricing Algorithm Facilitate Tacit Collusion? An Experimental Study Using Deep Reinforcement Learning in Airline Revenue Management

Chengyan Gu*

January 2023†

Abstract

This study conducts a series of simulated experiments in which airlines use deep Q-learning (DQL) algorithms to dynamically control the class of quantity and price pairs offered to the market through the booking horizon to better understand the algorithmic collusion problem. We show that, in a monopoly market, DQL algorithm can learn stochastic demand without any prior knowledge and achieve optimal monopoly revenue. In a duopoly market, with limited information requirements, DQL algorithms can contribute to the same knowledge pool, coordinate airlines' behaviors, learn to collude and share the monopoly profit equally. Compared with the expected marginal seat revenue (EMSR)-b heuristics, DQL algorithms are more adaptive for learning new demand stochasticity and are more likely to sustain collusive outcomes.

Keywords: Airlines, Collusion, Artificial Intelligence, Deep Q-Learning

JEL Codes: L13, L41, D43, C63

* Columbia University. Contact: chengyan.gu@columbia.edu.

† The views expressed in this paper are those of the author and do not reflect those of the author's affiliations. All the opinions and errors remain the sole responsibility of the author.

1 Introduction

The rapid growth of artificial intelligence (AI) pricing algorithms and software has brought many new challenges to the competition law landscape. In particular, the risk of tacit collusion through algorithms has raised concerns that algorithms can learn to collude automatically without any communication and weaken competition (e.g., reduce outputs, increase prices, and lower consumer welfare) in an anonymous way. This would confront the current antitrust policy which typically targets only explicit collusion among firms (e.g., the evidence of an overt act of agreement communication).[1]

On the one hand, in the airline industry, a number of collusion-facilitating practices such as multimarket contact and code-share alliances have been extensively studied in the literature (e.g., Evans and Kesside, 1994; Singal, 1996; Bilotkach, 2011; Ciliberto and Williams, 2014; Gayle, 2008; Ciliberto et al., 2019). On the other hand, while the current rule-based revenue management system (e.g., expected marginal seat revenue (EMSR)) still works well in practice, there is a growing interest in the industry to build its next generation of revenue management systems using AI techniques, such as reinforcement learning (RL) algorithms (Vinod, 2021).

Using the airline industry as an example[2], this study investigates whether a multi-agent deep RL based dynamic pricing algorithm can lead to collusion. Specifically, based on a series of simulated experiments on a single leg route, in which airlines dynamically control the class of quantity and price pairs offered to the market through the booking period, it shows that the deep Q-learning algorithm (DQL) (i.e., Deep Q-learning Network, DQN) can learn the stochastic demand through trial and error, without any prior knowledge of the demand distribution, customer mix, cancellation possibility, and arrival timing uncertainty, slightly overbook the flight and achieve optimal monopoly revenue. In a duopoly market,

[1] Harrington (2018) and Calvano et al. (2020) survey the discussions from both the academic and antitrust policy standpoints.

[2] While this study models a revenue management system in the airline industry, the framework and setting can be generalized and extended to other industries with common features (e.g., hotel, cruise and car rental).

with capacity commitments, DQN algorithms can learn from all airlines' experiences, coordinate them to behave consistently and cooperatively, achieve collusive outcomes with limited information requirements, and equally share the monopoly profit.

This study contributes to our understanding in several ways. First, based on simulated experiments that mimic industry practices, it shows that dynamic pricing algorithms only require very limited information to achieve collusive outcomes, without the need for any explicit communication. The collusive algorithms seek only public information, such as total number of bookings by time, each airline's capacity, fare classes, prices, and quantities offered on the market, which are normally available or can be inferred by industry practitioners, given the fact that airlines play tons of repeated games on each route day-to-day. Contributing to the same knowledge pool (e.g., historical databases or outsourcing companies[3]), DQN algorithms can autonomously learn from each other's strategies, coordinate, and behave consistently as a monopoly.

Second, this study compares the performances of DQL algorithms and EMSR-b heuristics in either a monopoly or duopoly. In a monopoly market, when the demand forecasting used in EMSR-b rules is precise, the EMSR heuristics can perform as well as the DQL algorithm. However, EMSR-b heuristics underperform DQL algorithm when demand forecasting is biased: an 10% underestimation in the number of business customers leads to approximately 5% revenue loss in the EMSR-b system. In a duopoly market, stochastic demand could bias EMSR's forecasting (which lacks quick learning ability) and hinder two EMSR-b airlines from achieving collusive outcomes, whereas DQL algorithms are more adaptive, can learn new uncertainty autonomously, and sustain collusive behaviors.

In addition, this study contributes to the literature on algorithmic collusion problem. Compared to previous studies (Waltman and Kaymark, 2008; Calvno et al., 2020; Klein, 2021) on this topic, which typically adopt a Q-learning framework (which works well in a

[3]Such as DB1B (Data Bank 1B), T100, MIDT (Marketing Information Data Tapes), OAG (Official Airline Guide), PODS (Passenger Origin/Destination Simulator), etc. Harrington (2022) discusses the anti-competitive effect of outsourcing pricing algorithms to a third party developing company.

low-dimensional state-action space), this study models the airline collusion problem in a high-dimensional space using DQN which is more suitable for complex decision-making tasks (e.g., when there are heterogeneities related to non-linearities and interactions among features). Instead of modeling collusion in a Bertrand or sequential pricing game, this study models the airline collusion using a cooperative multi-agent reinforcement learning (MARL) framework and follows the industry practices where airlines simultaneously and jointly determine the class of quantity and price pairs offered to the market over the booking period.

The remainder of this paper is organized as follows. The next section reviews related works. Section 3 provides a description of the DQL algorithms used in the simulation. Section 4 describes the airline market in which the simulations are performed. Section 5 discusses the results and Section 6 reports the robustness check. Section 7 concludes the paper.

2 Related Work

Collusion among oligopolistic firms can achieve monopoly outcomes, such as reduced output and increased price, and has been extensively studied in both literature and policy practice (e.g., Harrington (2008); Asker and Nocke (2022)). Theoretically, in an infinitely repeated game, even without any explicit collusion, the threat of a vigorous price war or other credible retaliations would be sufficient to deter oligopolists' temptation to cut prices and drive them to coordinate as a monopoly (e.g., Friedman, 1971; Abreu, 1986). While various factors can facilitate tacit collusion, in the airline industry, it is found that average fares in a market increase with the degree of multimarket contact among firms (Evans and Kessides, 1994; Singal, 1996; Bilotkach, 2011; Ciliberto and Williams, 2014; Ciliberto et al., 2019), but there is no evidence that average prices increase as a result of the code-share agreement (Gayle, 2008). Aryal et al. (2022) examine whether airlines communicate via earnings calls to coordinate capacity reduction and therefore increase fares.

Another strand of research focuses on dynamic inventory and pricing controls in the airline industry. In practice, airlines rely on rule-based policies to control the class of (quantity, price) pair available to the market dynamically. Littlewood rule accepts a discount passenger if the discount fare is greater than the expected revenue of future full-price passengers. EMSR a/b rules (Belobaba, 1987, 1992; Talluri and Van Gyrzin, 2004) set the joint seat protection level by applying Littlewood rule to all class pairs. While these heuristics still work well (Vinod, 2021)[4], there is growing interest in leveraging RL algorithms to achieve autonomous inventory and price control. Gosavi et al. (2002) model this control problem as a Markov-decision process, approximate the state-action value function as a temporal-difference Bellman equation, and show that the optimal policy can be found through the Q-learning. Shihab and Wei (2022) use a deep neural network to determine the optimal policy, avoiding the need to specify the value function for all possible state-action combinations.

Those approaches often assume a single player case. In reality, airlines usually operate in an oligopolistic market, which requires a MARL framework. Previously, the transition from single agent RL to MARL was challenging because of known obstacles, such as nonstationarity, partial observability, and curse of dimensionality. The breakthrough of applying deep learning methods in single agent RL settings (e.g., game playing, Mnih et al., 2015; Silver et al., 2016; robotics, Levine et al., 2016) has enabled many critical challenges to be addressed. Typically, based on the designed reward structure, MARL can be modeled in three settings (Busoniu et al., 2008; Zhang et al., 2021): in a cooperative setting, agents work together to maximize a joint or equally shared reward; in a competitive setting, agents play a zero-sum game to maximize their own rewards while minimizing rivals' rewards; in a mixed general-sum game setting, there is no restriction on agents' goals.[5]

Limited work has focused on the algorithmic collusion problem in a multi-agent setting.

[4]For example, the discount allocation model in American Airlines' DINAMO (Dynamic Inventory and Maintenance Optimizer) system is an expected marginal seat revenue model based on the logit approximation of the normal distributed demand.

[5]These frameworks have been applied to solve tasks with real-world complexity and achieve human-level performance, such as strategy games (Dota 2, OpenAI, 2019; StarCraft, Vinyals et al., 2019), bidding optimization (Jin et al., 2018), autonomous driving (Sallab et al., 2017), trading (Bao and Liu, 2019), etc.

Some studies have shown that Q-learning agents can learn to collude (Waltman and Kaymark, 2008; Calvno et al., 2020; Klein, 2021), but full collusion does not always emerge. This is not surprising as traditional Q-learning and policy gradient approaches suffer from the nonstationary problem[6]: Because each agent's policy is changing, the environment becomes nonstationary from the perspective of any independent player (Tan, 1993). Multiple equilibria exist, and there is no guarantee that the game converges to any Markov perfect equilibrium. In addition, these approaches are often poorly suited when the state-space exponentially increases with the number of agents and other state features. Leveraging a deep RL framework,[7] Kastius and Schlosser (2020) model agents competing in a sequential price-cutting game and show that a bounded price cutting strategy can lead to a collusion price. Bondoux et.al (2020) model airline algorithm competition as a pure price game and show that if both duopoly players adopt deep RL pricing algorithms, it can lead to more competitive prices and reduced revenues.

Unlike the work described above, which models MARL in a competitive setting, this study formulates the algorithmic collusion problem in a cooperative setting. Specifically, it shows that two airlines can cooperate through algorithms to dynamically control their seat quantity and price to achieve monopoly outcomes (i.e., equally sharing monopoly profits). Instead of modeling airline interaction as a Bertrand or sequential pricing competition, this study follows the industry practice assuming that airlines can jointly decide the class of (quantity, price) pairs in the market under certain capacity constraints and commitments. In addition, by applying a DQN in a cooperative setting, this study overcomes some limitations in the traditional Q-learning framework, such as the nonstationary issue.

[6]Value-based methods like Q-learning learn the value function of each state-action pair and derive the optimal policy choosing the action with the highest value. Policy-based approaches like the score function method (policy gradient) directly learn the probability distribution over actions for each state and search the optimal policy selecting the action with the highest probability. Policy-based approaches also tend to suffer from the high variance problem when coordination of multiple agents is required (Lowe et al., 2017).

[7]Actor-Critic methods combine value-based and policy-based method and consist of an actor learning a policy and a critic learning a value function to evaluate the state-action pair. Popular methods include advantage actor-critic (A2C), asynchronous advantage actor-critic (A3C), soft actor-critic (SAC) and proximal policy optimization (PPO). SAC is an off-policy method built on actor-critic framework and uses the cross-entropy maximization (based on softmax function) to encourage exploration.

3 Methods

3.1 Single-Agent RL

Markov Decision Process (MDP): Typically a RL agent is modeled to perform sequential decision-making by interacting with unknown environments. Such environments are often formalized as MDP, defined by a tuple (S, A, P, R, γ). At each timestep t, an agent observes a state $s_t \in S$ and chooses an action $a_t \in A$ that determines the reward $r_t \sim R(s_t, a_t)$ and next state $s_{t+1} \sim P(s_t, a_t)$. P is the transition probability function and $\gamma \in [0, 1)$ is the discount factor. The goal is to find a policy $\pi : S \to \Delta(A)$, a mapping from the state space S to the distribution over action space A (i.e., $a_t \sim \pi(\cdot|s_t)$), to maximize the expected discounted sum of rewards, $E_\pi[\sum_{t=0}^{T} \gamma^t r_t]$.

Q-Learning: Q-Learning is a model-free off-policy algorithm for estimating the long-term expected return of executing an action from a given state. The estimated returns of the state-action pairs are Q-values that can be learned iteratively by updating the current Q-value estimate towards the observed reward plus the maximum Q-value over all actions $a^{'}$ in the next state $s^{'}$:

$$Q(s, a) = Q(s, a) + \alpha(r + \gamma \max_{a^{'}} Q(s^{'}, a^{'}) - Q(s, a))$$

where $\alpha \in [0, 1)$ is the learning rate that determines the extent to which newly acquired information overrides old information. A common way of deriving a new policy during the iterations is the ϵ-greedy strategy: at each state s_t, with probability ϵ a random action is selected and with probability $1 - \epsilon$ the best action maximizing Q_t is selected. This gives the agent exploration ability, enabling it to discover new regions in the state-action space and correct current estimates when appropriate.

Deep Q-Learning (DQL): While this method works well when the $S \times A$ tabular space is small, it is poorly suited to questions with a huge state-action space such as Atari

games. As such, instead of maintaining an estimate for each state-action pair, DQL uses a neural network $Q(s, a|\theta)$ to approximate the Q-values, where θ are the weights of the neural network that parameterize the Q-values. The agent only needs to learn and update the neural network weights by minimizing the loss function:

$$L(s, a|\theta_i) = (r + \gamma \max_{a'} Q(s', a'|\theta_i) - Q(s, a|\theta_i))^2$$

The back-propagation algorithm can be used to update the weights at iteration $i + 1$ using $\theta_{i+1} = \theta_i + \alpha \nabla_\theta L(\theta_i)$. This can significantly lower the learning space when $|\theta| \ll |S \times A|$ and generalize to unseen state-action pairs.

Because the same network is used to generate the next state target Q-values, which are then used to update its current Q-values, such an update can diverge. In practice, two techniques are often used to stabilize the training process: (1) experience replay and (2) target networks. First, past experiences (s, a, r, s') are stored in a replay buffer and sampled uniformly during training. This not only reduces the correlation between consecutive samples but also helps the network remember previous experiences. Second, instead of updating the target network (i.e., $r + \gamma \max_{a'} Q(s', a'|\theta)$) at every iteration, θ is fixed as θ^- and updated every τ steps. This slowly updated target network ensures that the targets come from a relatively stationary distribution and hence stabilizes learning.

3.2 Multi-Agent RL

Markov Games: In MARL, a set of autonomous agents interact with unknown environments as well as other agents. When agents have full observability of the state, the problem can be presented by a Markov game (MG), defined by a tuple (N, S, A, P, R, γ) where N is the set of N agents, S denotes the state space observed by all agents, $A = A^1 \times A^2 \times \cdots \times A^N$ is the joint action space of N agents, $P : S \times A \to \Delta(S)$ denotes the transition probability from any state $s_t \in S$ to a new state $s_{t+1} \in S$ for any joint action A, and $R = (r^1, r^2, ..., r^N)$

where r^i is each agent's immediate reward and $\gamma \in [0, 1)$ is the discount factor. The goal of each agent i is to find a policy $\pi^i : S \to \Delta(A^i)$ such that $a_t^i \sim \pi^i(\cdot|s_t)$ to maximize its expected discounted sum of rewards: $E_{\pi^i, \pi^{-i}}[\sum_{t=0}^{T} \gamma^t r_t^i]$, where $-i$ represents the indices of all agents in N, except agent i.

Nash Equilibrium: In this MG, the optimal performance of each agent is controlled not only by its own policy but also by the choices of all other agents. Given this, the Nash Equilibrium (NE) of the MG can be defined as a joint policy $\pi^* = (\pi^{1,*}, ..., \pi^{N,*})$ such that for any $s \in S$ and $i \in N$:

$$V^i_{\pi^{i,*}, \pi^{-i,*}}(s) \geq V^i_{\pi^i, \pi^{-i,*}}(s), for\ any\ \pi^i.$$

where $V^i(s)$ denotes the value function for agent i in state s. In a NE, there exists an optimal policy set π^* in which no agent has an incentive to deviate from it. For any agent i, policy $\pi^{i,*}$ is the best response of all the other agents' policies $\pi^{-i,*}$. Note that, while learning NE is the standard goal of MARL, such a NE might not be unique.

Cooperative Setting: To model the collusion in a cooperative setting, several simplifying assumptions are made. First, agents have complete information and can fully observe others' states and actions (e.g., the number of tickets booked and prices offered on the market). This waives the need of modeling information-sharing and communication processes and potential mixed strategies, which can lead to nonstationary issues. Second, symmetric agents share a common reward function $R^1 = R^2 = ... = R^N = R$. As such, the value function and Q-function are identical for all agents, and these enable agents to contribute and learn from a common knowledge pool.

In addition, the agents are assumed to be controlled by a centralized controller. During the training process, the central controller has complete information about the entire environment and learn from the observations and policies of all agents. During the execution stage, all agents simultaneously send their states to the central controller, which then decides which action to take for each agent. Although this centralized approach is computationally expensive in large environments, it enables the single-agent RL algorithm to be applied, and

all agents can coordinate behaviors as one decision maker.[8]

4 Airline Market

This study considers the optimization problem in a single leg market: Stochastic demands with overbooking and cancellations are assumed for two types of customers (i.e., business and leisure), and then airline(s) dynamically decide what fare classes are available through the booking period.

4.1 Customer Arrival Model

To model the relative number and timing of potential customer arrivals, business (B) and leisure (L) customers are assumed to arrive according to a well-studied (McGill and Van Ryzin, 1999), nonhomogeneous Poisson process (NHPP) with a mean rate $\lambda_i(t)A_i$ where $t \in [T, 0)$ is the booking period, and $i \in \{B, L\}$. A_i is the total number of expected arrivals for the customer class i:

$$\lambda_i(t) = x^{\alpha_i - 1}(1 - x)^{\beta_i - 1}/B(\alpha_i, \beta_i)$$

where $x = t/T$ distributes customers on the booking horizon and α_i and β_i are the shape parameters of beta distribution B (i.e., determining the arrival pattern for customer class i). While booking can be done up to 180 to 360 days prior to departure, airlines often cut such a booking horizon into discrete data collection points (DCPs) or decision control points. Without loss of generality,[9] here $T = 10$ DCPs are assumed over a 180 days horizon.

We assume and test three mean arrival settings to represent different market conditions: $(A_B, A_L) \in \{(40, 90), (60, 70), (20, 110)\}$. Market setting 1 reflects a normal market in which

[8] Another popular method uses a structure called centralized training and decentralized execution. For example, in the actor-critic architecture (Lowe et al., 2017), a central critic is assumed to have all agents' information and used to learn the value function, while an actor only has access to local information and use it to guide the policy (i.e., each agent has their own actor).

[9] Examples are shown in Walczak et al. (2012) and Vinod (2021). Gu (2022) empirically tests different pricing strategies airlines used across different booking horizons.

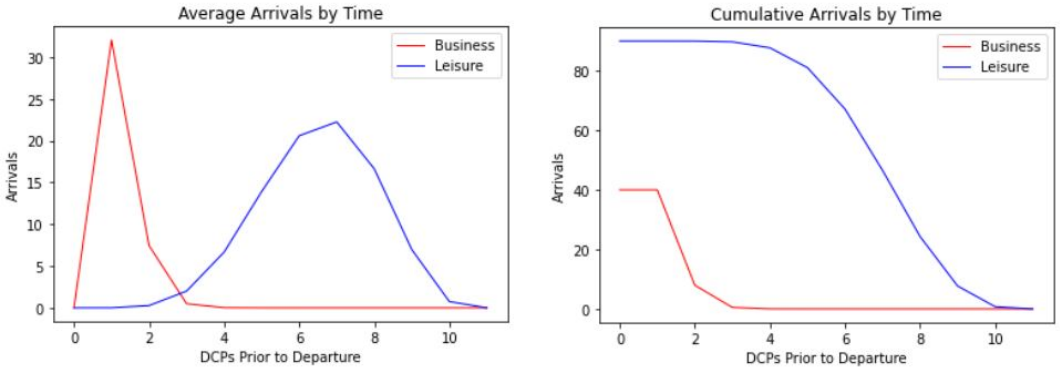

Figure 1: Average and Cumulative Arrivals by Customer Class

the number of business travelers is about half the number of leisure travelers, market 2 represents a business route, and market 3 represents a pleasure route. In general, business customers arrive later than leisure customers do, and their beta parameters are assumed to be $(\alpha_B, \beta_B) = (28, 2)$ and $(\alpha_L, \beta_L) = (4, 6)$.[10]

Figure 1 shows the average and cumulative arrivals over time by fare class, for a market setting of 1. This process generated from the NHPP mimics the reality and has several stochastic characteristics (Weatherford et al., 1993), such as uncertain total number of customers and an uncertain mix of customers for each simulated episode. Moreover, the NHPP produces an uncertain order of arrival. This allows the RL algorithm to learn stochastic policies rather than deterministic ones.

Overbooking: In a monopoly market the airline's capacity is assumed to be $K = 100$, and in the duopoly market each airline's capacity is assumed to be $K = 50$ (i.e., the airline makes a capacity commitment) to consider the overbooking. In either case, the total number of expected arrivals is larger than the airlines' total capacity. However, when airlines overbook, they need to buy leisure customers first and then business customers. Following the U.S. Department of Transportation's rules, the bumping cost of a leisure customer is assumed to be the minimum of 200% of the fare and \$775, and for business customers, the bumping cost is assumed to be \$775. This provides the airline with an incentive to avoid

[10]Different beta parameters are experimented and the results are robust to different settings.

overbooking too many customers.

Cancellation: Without loss of generality, three cancellation rates are considered during the simulation: 0%, 5% and 10%. Specifically, when a customer arrives, the retention rate is simulated through a normal distribution and then compared with the assumed cancellation rate. If this customer's retention rate is smaller than the cancellation rate, then the customer randomly chooses one day prior to departure to cancel the booking.

4.2 Customer Choice Model

When a customer arrives, they first discard fare offers that are more expensive than their willingness-to-pay (WTP). In a monopoly market, it is assumed that airlines can build up fences to segment the business and leisure market (Garrow, 2012). In other words, the customer only buys the ticket according to their class and there is no buy-down or buy-up.[11] In a duopoly market, when symmetric airlines offer the same fares to the customer, the customer chooses the airline according to a multinomial logit/softmax distribution:[12]

$$P^i = \frac{\exp(V^i(s^i))}{\sum_{j \in C} \exp(V^j(s^j))}$$

where $V^i(s)$ is the value function for airline i at state s from the neural network and C is the choice set. This allows allies to obtain different rewards and break ambiguities. If a customer does not see their preferred fare, they choose not to buy and leave the market.

4.3 Monopoly Market

In the monopoly market, a state s can be defined as a tuple $s = (n_B, n_L, t)$ where n_B is the total number of bookings made by business class customers until time t, n_L is the total

[11] This is often referred to the spiral-down effect. Our study tests such effect and finds that it generally leads to 5% revenue loss in current settings.

[12] When actual data are available, the customer demand (i.e., market shares of alternative itineraries) can be estimated through multinomial probit/nested logit models (Garrow, 2012; Vinod, 2021). Coldren et al.(2003) show an aggregate air-travel itinerary share model estimated at the city-pair level using the data from United Airlines.

number of bookings made by leisure customers, and $t \in [T, 0)$ is the time to departure. At each time step t, the airline must choose between two actions $a \in A$: $a_1 = (Open_B, Close_L)$ and $a_2 = (Open_B, Open_L)$. In Action 1, the business class is opened and the leisure class is closed, and in Action 2, both classes are opened.

As such, an airline with capacity $K = 100$ must make a sequence of $T = 10$ decisions. At every time step t, based on the airline's action, the number of newly booked customers is added to n_B or n_L separately according to their classes. Then, the resulting new state can be represented as $s^{'} = (n_B^{'}, n_L^{'}, t-1)$. On departure day $t = 0$, the airline needs to bump overbooked customers out and pay the bumping cost following the logic described in the overbooking model (e.g., bumping leisure customers first, then business customers).

Without loss of generality, the fares for leisure and business classes are assumed to be \$80 and \$240[13], and leisure and business class customers' WTPs are assumed to be \$100 and \$300, respectively. Though it is not a perfect price discrimination, this assumption simplifies the question by excluding an option that a customer has a WTP lower than any fare on the market and has to drop their travel plan.

4.4 Duopoly Market

In a duopoly market, the state space doubles $S = s^1 \times s^2$ where $s^1 = (n_B^1, n_L^1, t)$ and $s^2 = (n_B^2, n_L^2, t)$ represent the states of airline 1 and 2. In a complete information game, the central controller can observe two airlines' actions and states, and the state s can be compressed as $s = (n_B^i, n_L^i, n_B^j, n_L^j, t)$ where $i, j \in \{1, 2\}$ indicates the airlines. Each airline still has two action options $a \in A$: $a_1 = (Open_B, Close_L)$ and $a_2 = (Open_B, Open_L)$, but now they have to consider the possible actions adopted by their rivals as the next state is determined by the airlines' joint action choice. As a result, a joint action space can be defined as the 2×2 matrix shown below.

Airlines are assumed to be symmetric, and customers choose airlines randomly from a

[13]The price of a typical one-way flight ticket between Boston (BOS) to New York (LGA) or Chicago (ORD) to New York (LGA) is about $\$80 - 90$.

		Airline 2	
		a_1	a_2
Airline 1	a_1	(a_1, a_1)	(a_1, a_2)
	a_2	(a_2, a_1)	(a_2, a_2)

Table 1: Duopoly Action Space

multinomial logit distribution. Therefore, at each time step, the airlines may end up with different booking numbers. Airline 1's state is $s^1 = (n_B^1, n_L^1, n_B^2, n_L^2, t)$ and airline 2's state is $s^2 = (n_B^2, n_L^2, n_B^1, n_L^1, t)$. During training, the central controller learns from both airlines' experiences, and during execution, each airline supplies its own state to the central controller and obtains the optimal action guide. This process enables airlines to contribute to a common knowledge pool and learn from each other's experiences.

In this market, each airline's capacity is assumed to be half of the monopoly capacity (i.e., $K = 50$) and the prices are kept the same as the monopoly prices: \$80 for leisure class and \$240 for business class. The difference is that, for example, a leisure customer might see two airlines offer the \$80 ticket, and they have to choose one airline from them. In another case, if only one airline offers a leisure class ticket on the market, they can only book from that airline.

5 Experiments

5.1 Network Architecture

In either the monopoly or duopoly case, a dense feed-forward neural network with one input layer, three hidden layers and one output layer is trained. For the input layer, the input of the monopoly case is the three-dimensional state tuple $s = (n_B, n_L, t)$ and that of the duopoly case is the five-dimensional state tuple $s = (n_B^i, n_L^i, n_B^j, n_L^j, t)$ where $i, j \in \{1, 2\}$ indicates airlines. The three hidden layers include 128 ReLU-activated neurons (i.e., including the bias term). The output layer has two linearly activated nodes (one for each action), which output the Q-Values for a state-action pair. Figure 2 shows the neural network structure of

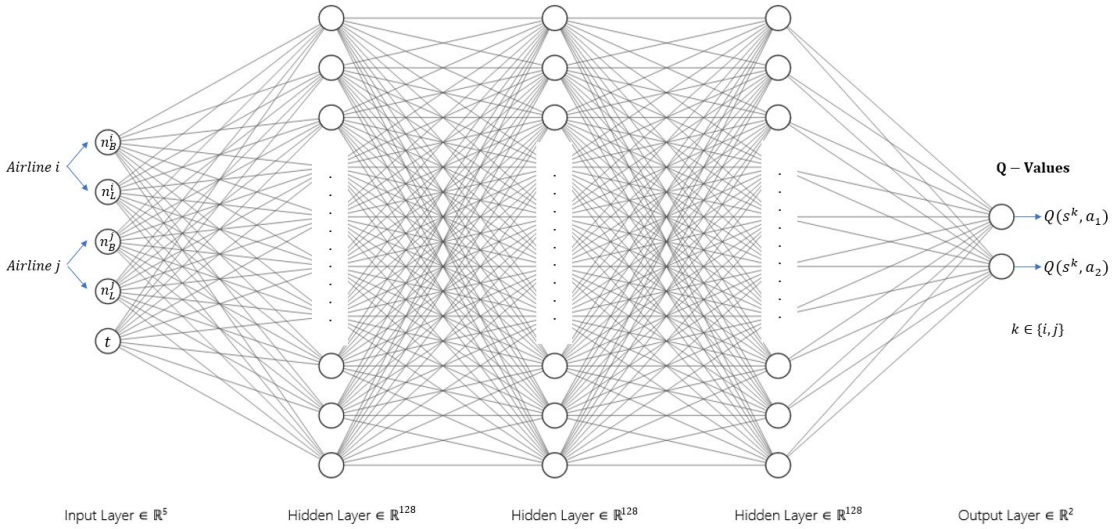

Figure 2: The Multi-agent Deep Q-Network Architecture

the central controller in a duopoly market.

To train the neural network, episodes with stochastic numbers of customer arrivals and cancellations are simulated. Experience replay and target network techniques are used during the training process. Specifically, each data sample includes an experience tuple of the form $(s, a^i, r^i, s^{'})$ where s is the starting state, such as $s = (n_B^i, n_L^i, n_B^j, n_L^j, t)$, a^i is the action airline i taken, r^i is the reward received and $s^{'}$ is the new state (i.e., as a result of new arrivals, cancellations, and the joint-action of the airlines). In addition, in the duopoly market, for each step, two experiences (i.e., one from airline 1 and one from airline 2) are recorded in the replay buffer and then sampled and learned by the central controller.

5.2 Monopoly Results

In the monopoly market, the airline knows the expected mean number of arrivals for the market in general, and it does not know the exact numbers of arrivals for each flight (i.e., each simulated episode). However, in an ideal scenario, if the exact numbers and timing of arrivals and cancellations are known, fare classes offered to the market can be controlled dynamically without any overbooking. The optimal revenue can be calculated given each

simulated episode and used as the benchmark.

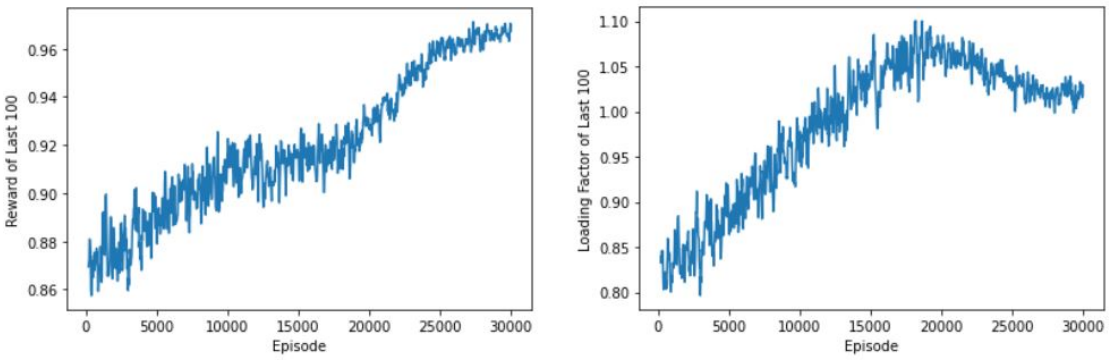

Figure 3: Rewards and Loading Factors: Mean Arrivals = $(40, 90)$, Cancellation = 5%

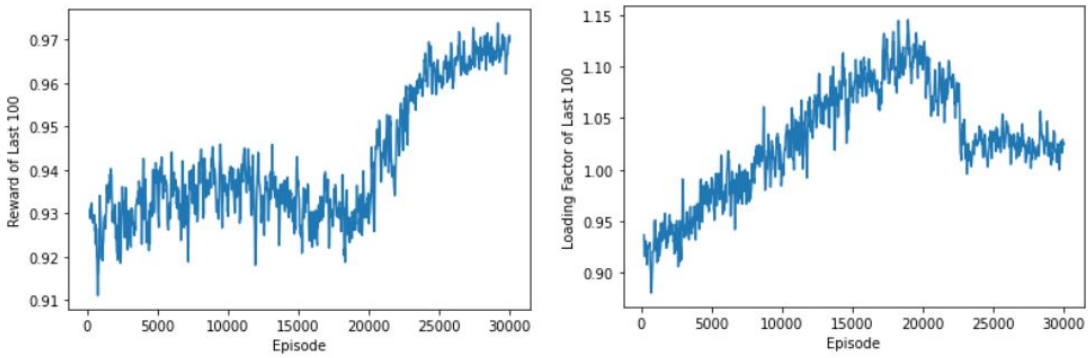

Figure 4: Rewards and Loading Factors: Mean Arrivals = $(60, 70)$, Cancellation = 5%

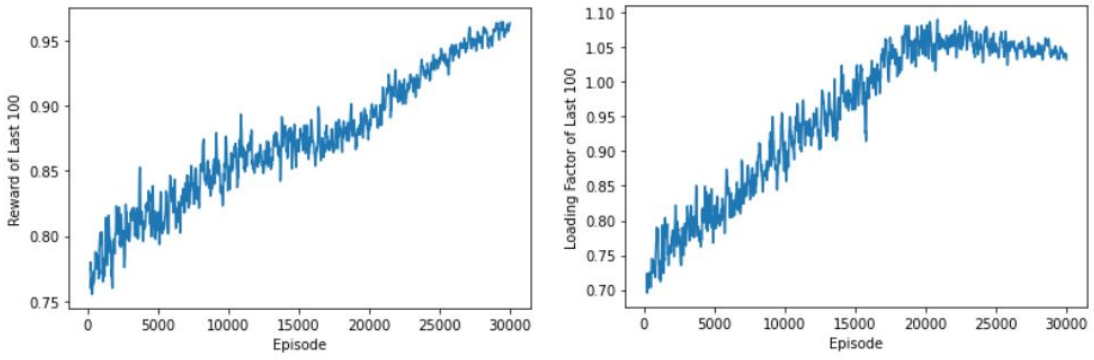

Figure 5: Rewards and Loading Factors: Mean Arrivals = $(20, 110)$, Cancellation = 5%

Figure 3-5 show the training results under different market mean arrival assumptions. Several findings can be observed. First, through the agent has no prior knowledge of customer

arrival distribution, customer mix and cancellation, it is able to learn them after a sufficient number of episodes. It can achieve approximately $96\% - 98\%$ of the theoretically optimal revenue. Second, during the learning process, the airline first learns to accept more customers to increase its loading factor (i.e., from 70% to 115%) and then learn to avoid overbooking and lower its loading rate to about $102 - 105\%$ (i.e., after considering the assumed 5% cancellation rate). Additionally, different market settings (e.g., business or pleasure routes) with different customer class mixes can affect the learning process. However, the agent can be sufficiently adaptive in different markets and converges to the optimal strategy. For example, the agent adopts a more aggressive overbooking strategy on the business route initially (i.e., comparing to pleasure route) and then gradually learns that it should reserve enough seats for business customers during the early booking period.

Table 2 compares the DQN agent and rule-based EMSR-b performances under various mean arrival assumptions. First, in an ideal case, the airline can perfectly predict each flight's number of arrivals for either the business or leisure class. The results show that EMSR-b heuristic can produce performance similar to that of the DQN agent (97%) in all market settings. Second, in biased case 1, it is assumed that the airline still correctly predicts the total number of arrivals but incorrectly assigns 10% of business customers to leisure class (i.e., underestimates the number of business customers). In this case, the average benefit of adopting a DQN agent is approximately 5% (i.e., between 3% and 7%). Last, in biased case 2, it is assumed that the airline can only predict the expected mean number of arrivals, and incorrectly assign 10% business customers to leisure class. In this case, the average benefit of employing a DQN agent can be slightly larger, by approximately 6%. This experiment shows the extra benefits of using DQN agent when demand forecasting has some biases.

5.3 Duopoly Results

Figure 6-8 show the training results for different mean arrival settings. Compared with the monopoly optimal results, several observations emerge. First, as duopoly airlines contribute

Table 2: Performance Comparison in the Monopoly Market: DQN vs. EMSR-b

Mean Arrivals	**DQN**		**EMSR - Ideal**		**EMSR-Biased 1**		**EMSR-Biased 2**	
	Reward	LF	Reward	LF	Reward	LF	Reward	LF
[40, 90]	0.967	1.026	0.970	0.946	0.921	1.136	0.920	1.144
	(0.007)	(0.019)	(0.003)	(0.006)	(0.008)	(0.030)	(0.009)	(0.018)
[60, 70]	0.970	1.008	0.969	0.934	0.941	1.126	0.885	1.250
	(0.007)	(0.021)	(0.003)	(0.006)	(0.005)	(0.011)	(0.012)	(0.028)
[20, 110]	0.969	1.010	0.970	0.958	0.894	1.148	0.920	0.892
	(0.008)	(0.018)	(0.004)	(0.005)	(0.007)	(0.012)	(0.012)	(0.015)

[a] Cancellation rate is assumed to be 5%.
[b] Standard deviations are reported in the parentheses.
[c] For each case, the comparisons are based on a common test sample including 500 episodes.

their experiences to the same central controller, they learn from each other's knowledge, and their behaviors converge to a stationary equilibrium. Second, regardless of market arrival settings, the duopoly agents achieve similar performance (i.e., $96\% - 98\%$) as the monopoly case. In other words, they can equally share the monopoly profit and achieve tacit collusion result by coordinating with each other. In addition, given that customer choice introduces another source of stochasticity, the agent takes a bit more time to learn and converge (i.e., 50,000 episodes). Finally, in the monopoly case, the reward resulting from an action is deterministic. Rather, in the duopoly case, airlines know that their actions can receive the reward but with some uncertainty. As a result, the airline adopts an aggressive booking strategy (i.e., overbooking to about 120% level) and then gradually learns to lower their loading factors to $100 - 105\%$ level. This provides direct evidence that duopoly airlines can collude and achieve monopoly profits.

Table 3 shows a scenario in which (1) both airlines adopt the EMSR-b heuristics and (2) customers are allowed to choose one airline randomly if both airlines offer their preferred fare class. In addition, it is assumed that airlines can perfectly predict the total numbers of expected arrivals in this route/episode. The results show that although airlines have perfect predictions on the whole episode, they do not have perfect forecasting on their own flights due to the randomness introduced in the customer choice process. This leads to revenue loss (i.e., $3\% - 7\%$ compared with the monopoly case) and a lower load factor with higher

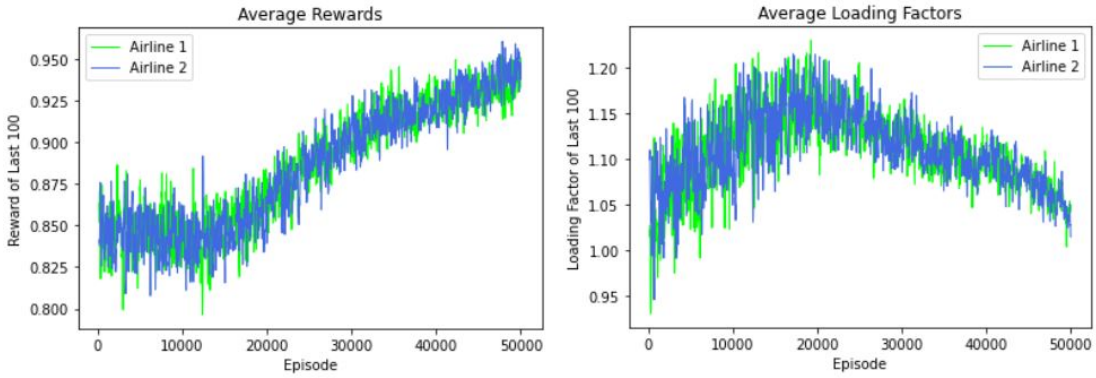

Figure 6: Rewards and Loading Factors: Mean Arrivals = $(40, 90)$, Cancellation = 5%

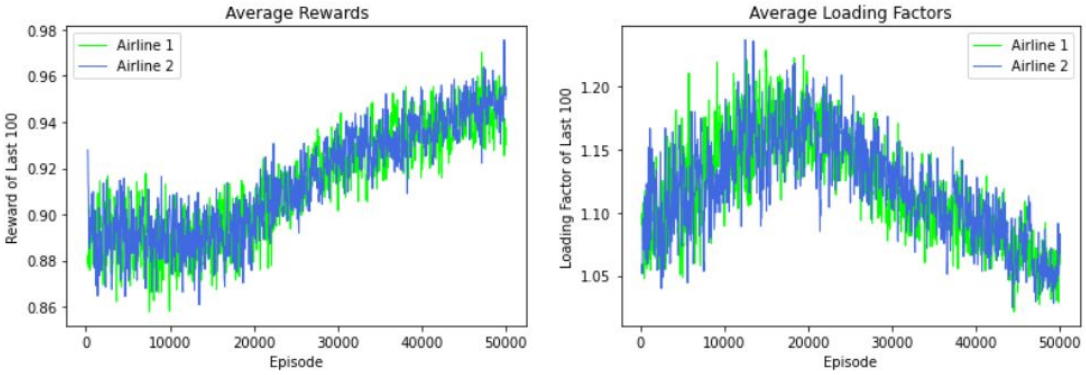

Figure 7: Rewards and Loading Factors: Mean Arrivals = $(60, 70)$, Cancellation = 5%

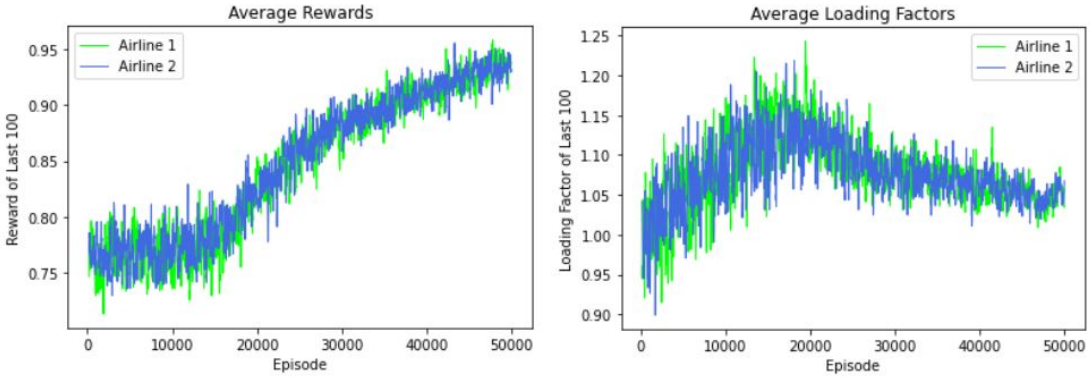

Figure 8: Rewards and Loading Factors: Mean Arrivals = $(20, 110)$, Cancellation = 5%

Table 3: Performance Comparison in the Duopoly Market: EMSR-b vs. EMSR-b

Mean Arrivals	Airline 1		Airline 2	
	Reward	**LF**	**Reward**	**LF**
[40, 90]	0.894	0.818	0.890	0.815
	(0.035)	(0.027)	(0.038)	(0.029)
[60, 70]	0.942	0.919	0.950	0.923
	(0.030)	(0.039)	(0.027)	(0.031)
[20, 110]	0.942	0.922	0.927	0.916
	(0.030)	(0.021)	(0.029)	(0.018)

[a] Cancellation rate is assumed to be 5%.
[b] Standard deviations are reported in the parentheses.
[c] For each case, the comparisons are based on a common test sample including 500 episodes.

variances. Compared with RL agents, EMSR-b heuristics do not have learning ability (i.e., which requires refreshing the model frequently used in demand forecasting) and thus lack the ability to adapt to the new source of uncertainty.

6 Robustness

Triopoly Market: When the number of airlines increases to three, the simulation results show that they can still achieve the monopoly outcome (i.e., about 95%) and each airline with a capacity ($K = 33$) can get 1/3 of the theoretical monopoly profit. Although the increase of the joint action space (i.e., from a 2×2 matrix to a 2^3 matrix) brings more complexities in defining the environment, the symmetric assumption can lead the algorithm to converge to a consistent behavior. In other words, these three players still share their experiences with the central controller and can learn from each other's experience and use the common Q-value functions (i.e., neural network) to guide their policies.

Arrival Patterns: Another test checks the model performances by changing the Beta shape parameters used to define the customer arrivals process. As shown in Figure 9, changing them can change the distribution mean position (i.e., customer arrival timing) and shape

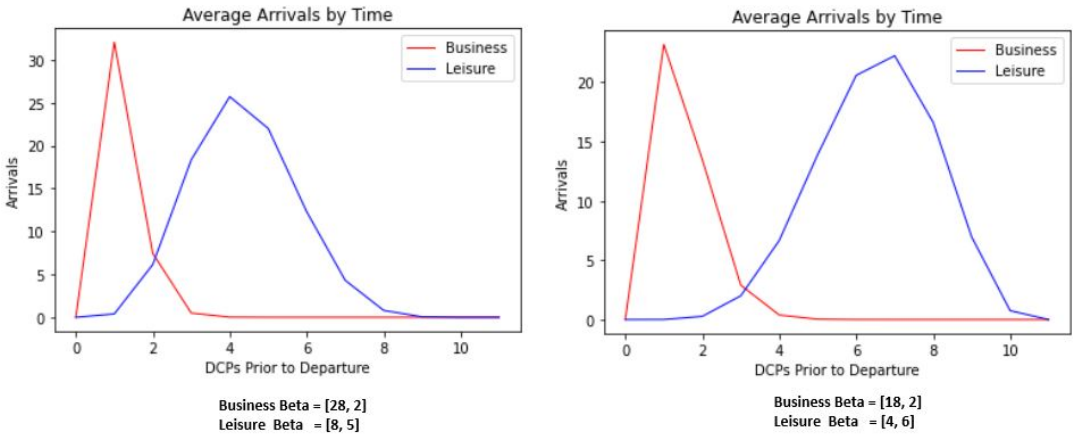

Figure 9: Arrival Curves with Different Beta Parameters: Mean Arrivals = $(40, 90)$

of the distribution (i.e., variances). For example, when changing leisure customers' beta values to $[8, 5]$, the mean booking times of business and leisure customers become closer to each other. Table 4 shows the comparison results: duopoly airlines' revenues are still similar to monopoly revenue, and the slight difference might come from one or two more overbookings in the duopoly case (e.g., 101.9% *vs.* 103.1% for a flight with capacity $K = 100$). The results and conclusions are robust to this change.

Different Cancellation Rates: Additional check is performed by changing the current 5% cancellation rate to 0% and 10%. In either case, although the monopoly agent does not have any prior knowledge, it can learn about the uncertainty around the cancellation process by keeping the loading factor slightly higher than 100%. In the duopoly market, similar learning curves are observed (as shown in Figure 6): while airlines adopt an aggressive booking strategy first increasing their loading factor to approximately 120% and then gradually lower the loading rate to about 107%. Compared with the monopoly case, agents might slightly overbook to offset the uncertainty around the customer choice process.

Spiral-Down Effect: The customer choice model assumes that a customer only buys the ticket according to their class and there are some segmentation fences that prevent them to buy-down. In this study, the WTP of leisure customers is assumed to be smaller than the business class fare, and thus there is no buy-up effect. Relaxing this assumption would allow

Table 4: Robustness Check: Beta Distribution and Cancellation Rate

Beta	Cancellation	Monopoly		Duopoly-Airline 1		Duopoly-Airline 2	
		Reward	LF	Reward	LF	Reward	LF
[28, 2], [4, 6]	**5%**	0.967	1.026	0.957	1.036	0.955	1.031
[28, 2], [8, 5]	**5%**	0.965	1.019	0.958	1.025	0.952	1.029
[18, 2], [4, 6]	**5%**	0.968	1.019	0.960	1.031	0.958	1.024
[28, 2], [4, 6]	**5%**	0.967	1.026	0.957	1.036	0.955	1.031
[28, 2], [4, 6]	**0%**	0.969	1.027	0.952	1.098	0.949	1.101
[28, 2], [4, 6]	**10%**	0.965	1.027	0.955	1.078	0.951	1.084

[a] Mean arrivals are assumed to be $[40, 90]$.

a business customer to buy a leisure class ticket (i.e., they no longer consider the business class ticket) when it is available on the market. The simulation results show that doing so would lower revenue by approximately 5% in either the monopoly or duopoly case (i.e., for the mean arrival setting $[40, 90]$).

7 Conclusions

This study shows that autonomous deep Q-learning neural network algorithms can learn to collude in the airline market. Even though the algorithm agents do not have any prior knowledge of the stochastic demand distribution, timing of the arrivals, customer mix, or, potential cancellation, they can learn a policy through sufficient trials and errors and achieve optimal revenue. It is also found that DQN agents outperform the biased EMSR-b heuristic by 5% on average, whereas EMSR-b heuristics rely heavily on accurate demand forecasting, DQN agents are more adaptive to learning new sources of uncertainties. Contributing to the same knowledge pool (i.e., the central controller), airline DQL agents can learn from each other's experience. Given the capacity constraint and commitment, their behaviors converge, which allow them to collude and achieve monopoly profits.

During this process, airlines do not need to communicate with one another. The collusive pricing algorithm requires limited information about the market, such as the total number

of bookings at each time step, the total capacity of their rivals, prices, and classes offered in the market, which are often publicly available through historical databases such as DB1B (Data Bank 1B), T100, MIDT (Marketing Information Data Tapes from various Global Distribution Systems), OAG (Official Airline Guide), PODS (Passenger Origin/Destination Simulator), or can be inferred by the industry practitioners, given the fact that airlines play tons of repeated games on each route day-to-day. This provides new challenges to antitrust interventions, as no information sharing is explicitly evident. On the other hand, antitrust practitioners are required to build a counterfactual model or benchmark agent showing the what-if outcomes in a fully competitive setting.

This study also provides some new perspectives to further study the tacit collusion problem in the airline industry. For example, to test the existence of collusion outcomes, the algorithm makes a capacity constraint and commitment assumption, which simplifies the question by excluding the incentive for airlines to deviate from the collusion level, as described in theoretical models. Therefore, it is of particular interest to study the relationship between capacity constraints and possible collusion behaviors. It is also interesting to investigate whether airlines converge to some consistent behaviors (i.e., either competitive or cooperative) throughout the booking period.

More research is needed to advance the understanding of dynamic pricing algorithm collusion problems, including but not limited to the airline industry. First, the current algorithm assumes that firms are symmetric. It is worth introducing some asymmetries into the multi-agent reinforcement learning algorithm (e.g., low-cost-carriers or maverick firms). Second, this study assumes two or three players in the market. To extend it to a general case with more players, the neighbor rivals or mean rivals algorithm might be worth trying. In addition, in the current setting, the state vector only considers the time and total number of bookings; this can be extended to include more variables when actual data are available. Finally, this study assumes that the central controller has complete information. In other industries, this may not hold. In such cases, algorithms with communication mechanisms

are required to study the collusion problem.

References

Abreu, D. 1986. "Extremal Equilibria of Oligopolistic Supergames," *Journal of Economic Theory*, 39(1), 191-225.

Aryal, G., F. Ciliberto, B. Leyden. 2022. "Coordinated Capacity Reduction and Public Communication in the Airline Industry," *Review of Economic Studies*, 89(6), 3055-3084.

Asker, J. and V. Nocke. 2022. "Collusion, Mergers, and Related Antitrust Issues," in K. Ho, A. Hortascu., and A. Lizzeri (eds), *Handbook of Industrial Organization*, Vol.5, New Holland: Elsevier, 177-280.

Bao W, and X. Liu. 2019. "Multi-agent Deep Reinforcement Learning for Liquidation Strategy Analysis," arXiv:1906.11046.

Belobaba, P. P., 1987. "Air Travel Demand and Airline Seat Inventory Management," MIT Flight Transportation Laboratory Report R87-7, Cambridge, MA.

Belobaba, P. P., 1992. "Optimal vs. Heuristic Methods for Nested Seat Allocation," Presentation at ORSA/TIMS Joint National Meeting.

Bilotkach, V. 2011. "Multimarket Competition and Intensity of Competition: Evidence from an Airline Merger," *Review of Industrial Organization*, 38 (1), 95-115.

Bondoux, N., A.Q. Nguyen, T. Fiig, and R. Acuna-Agost. 2020. "Reinforcement Learning Applied to Airline Revenue Management," *Journal of Revenue and Pricing Management*, 19, 332-348.

Busoniu, L., R. Babuŝka, and B. De Schutter. 2008. "A Comprehensive Survey of Multiagent Reinforcement Learning," *IEEE Transactions on Systems, Man, and Cybernetics*, 38, 156-172.

Calvano, E., G. Calzolari, V. Denicolò, and S. Pastorello. 2019. "Artificial Intelligence, Algorithmic Pricing and Collusion," *American Economic Review*, 110, 3267–3297.

Calvano, E., G. Calzolari, V. Denicolò, J.E. Harrington, and S. Pastorello. 2020. "Protecting Consumers from Collusive Prices due to AI," *Science*, 370 (6520), 1040-1042.

Ciliberto, F., and J. Williams. 2014. "Does Multimarket Contact Facilitate Collusion? Inference on Conduct Parameters in the Airline Industry," *RAND Journal of Economics*, 45 (4), 764-791.

Ciliberto, F., E. Eatkins., and J.W. Williams. 2019. "Collusive Pricing Patterns in the US Airline Industry," *International Journal of Industrial Organization*, 62, 136-157.

Coldren, G.M., F.S. Koppelman, K.Kasturirangan and A. Mukherjee. 2003. "Modeling Aggregate Air-travel Itinerary Shares: Logit Model Development at a Major US Airline," *Journal of Air Transport Management*, 9, 361-369.

Evans W., and I. Kessides. 1994. "Living by the Golden Rule: Multimarket Contact in the U.S. Airline Industry," *Quarterly Journal of Economics*, 109 (2), 341-366.

Friedman, J. 1971. "A Non-cooperative Equilibrium for Supergames," *Review of Economic Studies*, 38 (1), 1-12.

Garrow, L. 2012. "Customer Modeling," in C. Barnhart and B.C. Smith (eds.), *Quantitative Problem Solving Methods in the Airline Industry: A Modeling Methodology Handbook*, New York, NY: Springer, 1-34.

Gayle, P. 2008. "An Empirical Analysis of the Competitive Effects of the Delta/Continental/ Northwest Code-share Alliance," *Journal of Law and Economics*, 51 (4), 743-766.

Gosavi, A., N. Bandla, and T.K. Das. 2002. "A Reinforcement Learning Approach to a Single Leg Airline Revenue Management Problem with Multiple Fare Classes and Overbooking," *IIE Transactions*, 34, 729-742.

Gu, C. 2022. "Market Segmentation and Dynamic Price Discrimination in the U.S. Airline Industry," *Journal of Revenue and Pricing Management*, forthcoming.

Harrington, J. 2008. "Detecting Cartels," in Buccirossi P. (ed.), *Handbook in Antitrust Economics*, Cambridge, MA: MIT Press, 213-258.

Harrington, J. 2018. "Developing Competition Law for Collusion by Autonomous Price-Setting Agents," *Journal of Competition Law and Economics*, Vol.14, 331–363.

Harrington, J. 2022. "The Effect of Outsourcing Pricing Algorithms on Market Competition," *Management Science*, 68, 6889-6906.

Jin J, C. Song, H. Li, K. Gai, J Wang, W Zhang. 2018. "Real-time Bidding with Multi-agent Reinforcement Learning in Display Advertising," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management.* ACM, New York, 2193–2201.

Kastius, A. and R. Schlosser. 2021. "Dynamic Pricing under Competition Using Reinforcement Learning," *Journal of Revenue and Pricing Management*, 21, 51-63.

Klein, T. 2021. "Autonomous Algorithmic Collusion: Q-learning under Sequential Pricing," *RAND Journal of Economics*, 52(3), 538-599.

Levine, S., C. Finn, T.Darrell, and P.Abbeel. 2016. "End-to-End Training of Deep Visuomotor Policies," *Journal of Machine Learning Research*, 17, 1-40.

Lowe, R., Y. Wu, A. Tamar, J. Harb, P. Abbeel, I. Mordatch. 2017. "Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments," in *Advances in Neural Information Processing Systems*, V30.

McGill, J., and Van Ryzin, G. 1999. "Revenue Management: Research Overview and Prospects," *Transportation Science*, 33(2), 233-256.

Mnih, V., K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. 2015. "Human-level Control through Deep Reinforcement Learning," *Nature*, 518: 529-533.

OpenAI. 2019. "Dota 2 with Large Scale Deep Reinforcement Learning," arXiv:1912.06680.

Sallab, AE., M. Abdou, E. Perot, and S. Yogamani. 2017. "Deep Reinforcement Learning Framework for Autonomous Driving," *Electron Imaging*, 19, 70–76.

Shihab, S., and P. Wei. 2022. "A Deep Reinforcement Learning Approach to Seat Inventory Control for Airline Revenue Management," *Journal of Revenue and Pricing Management*, 21(2), 183-199.

Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. 2016. "Mastering the Game of Go with Deep Neural Networks and Tree Search," *Nature*, 529, 484-489.

Singal, V. 1996. "Airline Mergers and Multimarket Contact," *Managerial and Decision Economics*, 17(6), 559-574.

Tan, M. 1993. "Multi-agent Reinforcement Learning: Independent versus Cooperative Agents," *Proceedings of the Tenth International Conference on International Conference on Machine Learning*, 330–337.

Talluri, K.T. and G.J. Van Ryzin. 2004. *The Theory and Practice of Revenue Management*. New York, NY: Springer.

Vinod, B. 2021. *The Evolution of Yield Management in the Airline Industry: Origins to the Last Frontier*. Switzerland: Springer Nature.

Vinyals O., I. Babuschkin, W.M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, DH. Choi, R. Powell, T. Ewalds, P. Georgiev., 2019. "Grandmaster Level in Starcraft II using Multi-agent Reinforcement Learning," *Nature*, 575(7782), 350–354.

Walczak, D., A. Boyd., and R. Cramer. 2012. "Revenue Management," in C. Barnhart and B.C. Smith (eds.), *Quantitative Problem Solving Methods in the Airline Industry: A Modeling Methodology Handbook*, New York, NY: Springer, 101-161.

Waltman L., and U. Kaymak. 2008. "Q-learning Agents in a Cournot Oligopoly Model," *Journal of Economic Dynamics and Control*, 32 (10), 3275-3293.

Zhang L., Z. Yang and T. Başar. 2021. "Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms," in K. Vamvoudakis, Y. Wan, F. Lewis, D. Cansever (eds), *Handbook of Reinforcement Learning and Control*, New York, NY: Springer, 321–384.

Weatherford, L.R., S.E. Bodily, and P.E. Pfeifer. 1993. "Modeling the Customer Arrival Process and Comparing Decision Rules in Perishable Asset Revenue Management Situations," *Transportation Science*, 27(3), 239-251.

Appendix A.

Table 5: Deep Q-Network Hyperparameters

Hyperparameter	Value	Description
Max episodes	[30000, 60000]	The maximum number of training samples generated
Min batch size	4000	Number of training samples per update
Replay size	20000	Size of the experience replay dataset
Learning rate	0.01	Rate used by the optimizer
Update rule	ADAM	The parameter update rule used by the optimizer
Initial exploration	1.0	Initial ϵ value in ϵ greedy exploration policy
Min exploration	0.01	Minimum ϵ value in ϵ greedy exploration policy
ϵ decay rate	$(1.0 - 0.01)/max\_episodes$	Rate at which ϵ decreases
Discount rate	0.995	Discount factor on future rewards
Target network update	25	Frequency of updating the target network
Policy network update	5	Frequency of updating the policy network