Tacit Collusion by Pricing Algorithm with Rule-Based Rivals*

Yongzhi Xu†

Abstract

Pricing algorithms, especially reinforcement learning algorithms, have been widely used in the past decades for firms in competitive markets, which are very helpful for firms to capture more information of the market and their rivals. Existing literature has shown that, however, reinforcement learning algorithms could lead to supracompetitive prices without any communication between firms. In a framework of price competition between two firms both starting with a rule-based algorithm, we show evidence that (1) if one firm is competing using a reinforcement learning algorithm with a rule-based rival, the price is always going to weakly increase, and (2) if both firms are adopting, the one that adopts later would benefit more, and supracompetitive prices are obtained but the price change is ambiguous compared to the initial case. Our findings contribute to the literature by highlighting the importance of the order of algorithm adoption and the transition from rule-based algorithm to reinforcement learning algorithm, which makes it possible to compare the result with previous steady state in a more general case.

Keywords: Algorithmic Pricing, Collusion, Antitrust, Rule-Based Strategy

JEL Classifications: D21, D43, L13, L44

*All errors are my own. This is a preliminary and incomplete version.

†Department of Economics, Texas A&M University, College Station, TX, 77843 (yz_xu@tamu.edu).

1 Introduction

Reinforcement learning algorithms have been widely used to price goods and services in competitive markets. With the help of learning algorithms, especially reinforcement learning (RL) algorithms, firms are capable of learning and discovering more information about the market demand and pricing rules.

However, recent empirical and simulation studies have shown that algorithms may learn and maintain supracompetitive prices (price that's above the competitive level), but current antitrust policy is not able to regulate this tacit collusion. On the simulation experiments side, Calvano et al. (2020b) demonstrates that algorithmic pricing adopted by competing, profit maximizing firms can lead to substantially higher prices than their competitive counterparts in a repeated simultaneous game. Klein (2021) shows the similar result in a sequential game. On the empirical side, Assad et al. (2024), Asker et al. (2023) provides empirical evidence of algorithmic collusion in Germany's retail gasoline markets. If all firms in a market adopt algorithm, the gas stations' markup is increased by 20 to 30%. They also show that a hybrid market has no significant change. [1]

The recent development in algorithmic pricing raises concerns regarding the possibility of algorithmic collusion by government authorities, but it is still an open question how algorithmic collusion could be regulated. On the one hand, authorities could detect and punish firms that collude with algorithms afterwards. Calvano et al. (2020a) suggests that the authorities identify the collusive pricing rules by checking properties of prices derived from economic theory and studies of human collusion. On the other hand, authorities could set regulations that forbid firms from activities that could highly likely lead to collusion. Harrington Jr (2023) suggests regulation on no communication between third-party algorithm company with several different firms in the same market. Johnson et al. (2023) shows how platform could incentivize sellers so that they will not choose to set prices that are higher. In this paper, we will follow the second way and see whether no agreement on adopting algorithm would help with reducing the probability of collusion.

[1] The empirical evidence is indirect because the time of adoption of the pricing algorithms is not observed but inferred. The main evidence is the change of firms' pricing frequency.

One part of the algorithmic pricing phenomenon that may be relevant for the discussion on how to regulate collusion and that has not been studied is that firms may adopt pricing algorithms at different times. In real life, firms are not able to adopt pricing algorithms at the same time without communication, because current development of computer science is not able to detect the kind of algorithm that the rival is using. This raises the question of whether these supracompetitive prices are still going to be achieved just like the case where the algorithms start at the same time. Current regulation is able to capture this tacit collusion if supracompetitive is not achieved any more when firms adopt at different times, which would make it less of concern.

This paper answers what the equilibrium will be if firms adopt pricing algorithms at different times. We develop a model of pricing and use it to give predictions of how the pricing algorithms interact with each other. We then introduce Q-learning, which is a widely used reinforcement learning algorithm, in order to simulate firms' pricing in real life. We find that the price will always weakly increase when the first algorithm is adopted and supracompetitive prices do occur in the scenario when firms adopt prices at different times. This suggests that current regulation cannot address this tacit collusion.

We begin by developing an economic model to show theoretical predictions of our simulations. We start with a small market with small firms that have limited information about their rival and the market. In real life, for example, many third-party sellers on platforms like Amazon are not able to know the other sellers who are selling the identical product with him and how they set their prices. Therefore, they are not able to respond optimally without the help of a learning algorithm and set prices according to a simple rule. This simple rule could refer to platform preset rules, such as Amazon, as discussed in Wang et al. (2023), Brown and MacKay (2023) and Chen et al. (2016). Traditional competing strategies that are widely used, such as price trigger and grim trigger, could also be considered as a simple rule that could be written as a function of rival's price. We will see then if there is one firm in the market that could completely learn the rival's rule, then there will never be a price drop.

For our simulations of how prices are set, we use Q-learning as the algorithm adopted by firms. Q-learning is designed for the Markov decision process with finite state and action sets, where in a single agent problem described in our model, the Markov process is stationary and convergence is maintained. Q learning also links perfectly with the dynamic programming in Economics, which gives the result a very good economic interpretation.

Using Q learning and following the setup of our model, we observe a weak increase in price for any rule in our simulation, which is consistent with our model. We choose three rule-based strategies that are most popular in real life and research, which all satisfy assumptions in our model. We showed that when the first firm starts adopting an algorithm, the price will weakly increase given enough time periods. If the second firm also adopts, however, the price change becomes ambiguous.

The main contribution of our paper to the literature on algorithmic pricing is to investigate how the order of adoption matters and how algorithm interact with rules. As we discussed above, the existing literature has focused on algorithmic collusion when algorithms start at the same time (see Calvano et al. (2020b), Asker et al. (2023), Klein (2021)). We are also able to compare the supracompetitive price with the steady states before any algorithm enters instead of the static Bertrand Nash equilibrium. Another contribution of our paper is to show the transition of firms' pricing strategies from rule-based rival to algorithm rival. Chen et al. (2016), Wang et al. (2023) have compared the Rule-RL versus RL-RL but fail to capture how algorithms' learning path could be affected by the history.

The outline of the remaining paper is as follows. Section 3 summarizes the Q-learning, a reinforcement learning method. Section 2 presents a model to describe competition of firms and theoretical evidence on how algorithms would respond to rules. Section 4 discusses the simulation results, and Section 5 concludes.

2 Economic Model

In this section, we develop an economic model to show theoretical predictions of our simulations. We focus on a small market with small firms that have limited information about their rival and the market. We show that if there is one firm in the market that could completely learn the rival's rule, then there will never be a price drop.

We start with the case where no firms are using RL algorithms. Consider a infinitely

Firm 1:	Rule	RL	RL
Firm 2:	Rule	Rule	RL
	0		time

Table 1: The timing of firms' adoption of algorithm

repeated incomplete information game in which $n = 2$ symmetric firms act simultaneously. Time periods are indexed discretely by $t \in \{1, 2, \dots, \}$. In each period t, firm i earns a profit

$$\pi_i(p_{i,t}, p_{j,t}) = (p_{i,t} - c) * q_{i,t}(p_{i,t}, p_{j,t}),$$

where $c = 1$ is the constant marginal cost, and $q_{i,t}$ is the logit demand

$$q_{i,t}(p_{i,t}, p_{j,t}) = \frac{e^{\frac{\gamma_i - p_{i,t}}{\mu}}}{\sum_{j=1}^{n} e^{\frac{\gamma_j - p_{j,t}}{\mu}} + e^{\frac{\gamma_0}{\mu}}}.$$

In a small market, firms are not able to respond optimally since they have limited information about their rival and the market without the help of an RL algorithm. Before time 0, firms were competing with an exogenous simple rule $p_{i,t} = p_i(p_{j,t-1})$, which is a pure strategy. Since firms are symmetric, we also assume that their strategies are symmetric, i.e., $p_i = p_j$. At time 0, without loss of generality, firm 1 adopts a reinforcement learning (RL) algorithm. Firm 2 has no information about this and keeps the same rule until t_2. We call $t < 0$ as pre period, $0 \leq t < t_2$ as peri period, and $t \geq t_2$ as post period. The process is shown in Table 2.

Peri period In the absence of RL algorithms, assume firms were in a steady state (p_0, p_0) before time 0, i.e., in the pre period. Given the assumption above, a necessary condition is

$$p_1(p_0) = p_2(p_0) = p_0.$$

Suppose that firm 1 instead has complete information about firm 2. If there exists a p_1^* such that, for any available price p', we have

$$\pi_1(p_1^*, p_2(p_1^*)) \geq \pi_1(p', p_2(p')),$$

and equality holds if and only if $p_1^* = p'$. Then, the optimal price of firm 1 will be p_1^* instead of p_0 and the new steady state is $(p_1^*, p_2(p_1^*))$. The relationship between p_1^* and p_0, however, is ambiguous. To see more about this relation, we impose the following assumptions.

Assumption 1. Define p^N to be the static Bertrand Nash equilibrium price, and p^M to be the static monopoly price, i.e.,

$$p^N = \arg\max_p \pi_1\left(p, p^N\right)$$
$$p^M = \arg\max_p \pi_1(p, p)$$

For any $p \in [p^N, p^M]$

i. $p_2(p) \leq p$.

ii. $\pi_1(p, p)$ is non-decreasing.

iii. $p_2(p)$ is non-decreasing.

The assumptions above are satisfied in most real life examples and widely used theoretical models. Assumption i. means that the strategy is "competitive". In real life, it's usually irrational to choose a price that would be higher than the rival in a steady state if the firm has no information about it's rival's price and the rival's price is rational, i.e., $p \in [p^N, p^M]$. Notice that this assumption only works when firm has limit information about the rival's strategy. For example, it would happen that $p_1^* > p_2(p_1^*)$ because firm 1 has complete information.

Assumption ii. is a weaker condition of concave monopoly profit, which would work for most of the commonly used demand functions. Assumption iii. means that since firms have limited information, it'd be better to follow the rival and raise price when they increase.

Proposition 1. Given Assumption i. and ii., $p_1^* \geq p_0$. Further with Assumption iii., $p_2^* \geq p_0$.

Therefore, if the algorithm is able to fully learn the rival's strategy, then there must be a price increase in peri period compared to pre period.

We take three typical rules as example: myopic, price undercut and price trigger. The rules are defined as follows.

- Myopic

$$p_i(p_{j,t-1}) = \arg\max_{p_{i,t}} \pi_{i,t}(p_{i,t}, p_{j,t-1})$$

- Undercut

$$p_i(p_{j,t-1}) = \begin{cases} p_{j,t-1} - \Delta p & \text{if } p_{j,t-1} > p^N \\ p^N & \text{o.w.} \end{cases}$$

- Price trigger

$$p_i(p_{j,t-1}) = \begin{cases} p^M & \text{if } p_{j,t-1} = p^M \\ p^N & \text{o.w.} \end{cases}$$

where p^N is the static Bertrand price, p^M is the static monopoly price. All three could lead to steady state where both firms price the same as static Bertrand price. Price trigger could also lead to static monopoly price, if firms start at some specific state. These steady states are not guaranteed to be equilibrium, especially with logit demand. Firms, however, are not able to deviate since they can only observe their rival's action but not strategy.

3 Q-learning

3.1 Single Agent Problems

The reinforcement learning (RL) algorithm we use is Q-learning. Due to the learning procedure, Q-learning is not as fast as other RL, but since it's motivated by the dynamic programming problem, Q-learning has a good economic interpretation.

Consider first a stationary Markov decision process. In each period $t = 0, 1, 2, \ldots,$ an agent observes a state $s_t \in S$ and then chooses an action $a_t \in A$. Both S and A are time-invariant and A is state-independent. Agent receives a payoff $\pi_t = \pi(s_t, a_t)$, which could be random, at each period t, then the system moves on to the next state s_{t+1}.

Let $a^*(s)$ represent an optimal policy. The decision-maker's problem is to maximize the expected present value of discounted payoff:

$$\mathbb{E}\left[\sum_{t=0}^{\infty} \delta^t \pi_t\right] = \mathbb{E}\left[\sum_{t=0}^{\infty} \delta^t \pi(s_t, a^*(s_t))\right],$$

where $\delta < 1$ represents the discount factor. Let $V(s)$ denote the value of being in state s

$$V(s) = \max_{a \in A} \Big\{ \mathbb{E}[\pi|s,a] + \delta \, \mathbb{E}[V(s')|s,a] \Big\},$$

which represent the maximum discounted payoff in state s, and $Q(s,a)$ be the choice specific value function

$$Q(s,a) = \mathbb{E}[\pi|s,a] + \delta \, \mathbb{E}\left[\max_{a' \in A} Q(s',a')|s,a \right],$$

which represent the future discounted payoff of taking action a at state s and choose optimal policy function $a^*(s)$ in the future. Notice that Q-function is related to the value function by

$$V(s) = \max_{a \in A} Q(s,a).$$

Q-learning could deal with the case where both state and action are finite. In such a case, Q-function collapses to a matrix. If the Q-matrix were known, the optimization problem could be solved by searching the maximizer of the specific row of Q-matrix corresponding to state s, or

$$a^*(s) = \arg\max_{a \in A} Q(s,a).$$

Therefore, as long as the Q-matrix is known, without knowing any underlying model, the agent is able to solve the optimization problem. Q-learning is the algorithm that estimates the Q-matrix using the following iterative procedure. Starting from an arbitrary initial matrix Q_0, the algorithm chooses an action a_t at state s_t for each time period. After observing the payoff π_t, the algorithm will update one cell of Q-matrix according to the following learning rule:

$$Q_{t+1}(s,a) = (1-\alpha)Q_t(s,a) + \alpha \left[\pi_t + \delta \max_{a \in A} Q_t(s',a) \right],$$

where the weight $\alpha \in [0,1]$ is a step-size parameter, which influences the learning rate. For all other cells $s \neq s_t$ and $a \neq a_t$, the Q-value does not change. Since α is constant, the algorithm actually puts more weight on recent observations.

If the agent doesn't have a good expectation of Q-matrix, it's very likely that the initial matrix Q_0 is very different from Q. For state s and an action $a' \neq a^*$, such that $Q(s,a') < Q(s,a^*)$, so it's not the optimal action at state s. If there exist s, a such that $Q_0(s,a^*) < Q(s,a')$ and $Q_0(s,a^*) < Q_0(s,a')$, so $Q_0(s,a^*) < Q_t(s,a')$ and $Q_t(s,a^*) = Q_0(s,a^*)$ for any

t, because it never updates $Q_t(s, a^*)$. In this case, the algorithm will never learn that a^* is the optimal action at state s.

In order to estimate a^* and Q-matrix from an arbitrary initial matrix Q_0, the algorithm should be allowed to "make mistakes", or to explore non-optimal actions. The method we use in our analysis is the ε-greedy model, works as follows. The algorithm exploits (chooses the currently optimal action) with probability $1 - \varepsilon$ and to explore (randomize uniformly across all actions) with probability ε. The probability ε decays with time and is assumed to be $\varepsilon = e^{-\beta t}$, with $\beta > 0$.

3.2 Repeated Game

In a repeated game, stationarity is usually not satisfied. The state transition depends on the previous or current action of all players. Although convergence is not guaranteed ex ante, it can be verified ex post. In our simulation, convergence is achieved if for all players, their optimal policy function does not change for 10^5 consecutive periods. It diverges if it does not converge after 10^8 periods.

The initial Q_0 is set at the discounted payoff that would accrue to player i if opponents randomized uniformly and there's no antitrust agency:

$$Q_{i,0}\left(s, a_i\right) = \frac{\sum_{a_{-i} \in A^{n-1}} \pi_i\left(a_i, a_{-i}\right)}{(1-\delta)|A|^{n-1}}.$$

The visualized procedure of Q-learning is shown in Table 2.

4 Simulation Results

4.1 Parametrization

In our simulation, we follow the assumption that the firms are symmetric, and share a constant marginal cost $c = 1$. The vertical differentiation $\gamma_i = 2, \gamma = 0$, and horizontal differentiation $\mu = 1/4$. The demand function is therefore

$$q_{i,t}(p_{i,t}, p_{j,t}) = \frac{\exp(4(2 - p_{i,t}))}{\sum_{j=1}^{n} \exp(4(2 - p_{j,t})) + 1}.$$

Table 2: Q-learning: pseudo code

Initialize $Q_i(s, a)$

$t = 1, s^1 = \text{random}$

record actions $Opt_i(s)$

while $t < 10^9$

$$a_i^t = \begin{cases} \arg\max_p Q_i(s^t, a) & \text{with prob. } 1 - \varepsilon \\ \text{random} & \text{with prob. } \varepsilon = e^{-\beta t} \end{cases}$$

$$s^{t+1} = (a_1^t, \ldots, a_n^t)$$

$$Q_i(s^t, a_i^t) = (1 - \alpha)Q_i(s^t, a_i^t) + \alpha \left[\pi^t + \delta \max_{a \in A} Q_i(s^{t+1}, a_i^t)\right]$$

if $a_i^t == Opt_i(s^t)$ for 10^5 continuous periods

break

else

$Opt_i(s^t) = a_i^t$

end if

end while

The initial Q-matrix is set as the discounted payoff that would accrue to player i if rivals randomized uniformly:

$$Q_{i,0}(s, a_i) = \frac{\sum_{a_{-i}\in A^{n-1}} \pi_i(a_i, a_{-i})}{(1-\delta)|A|^{n-1}}.$$

We also have different setup for α and β, such that $\alpha \in \{0.05, 0.15, 0.25\}$, $\beta \in \{1 \times 10^{-5}, 2 \times 10^{-5}\}$. For each pair, we have 1000 sessions.

The action set A is defined uniformly distributed as $\{p^1, \ldots, p^{15}\}$, where $p^2 = p^N$ and $p^{14} = p^M$. In this setup, the Bertrand price is about 1.469, monopoly price is about 1.925.

4.2 Results

4.2.1 Peri Period

We first compare the difference between peri period and pre period, or if firm 2 never adopts algorithm. We can see from Table (3) that almost all results satisfy Proposition 1. There are only 6% in the Ceiling rule that we don't observe a price increase. Among those, half is such that $p_0 = p^{15}$ which does not satisfy the assumption. The others are when $p_0 = p^{14}$, because $\pi_1(p^{14}, p^{14})$ and $\pi_1(p^{13}, p^{13})$ have only 0.3% difference so the algorithm is not always able to differentiate that.

In the case when the rule is myopic or price undercut, the pre steady states are both the same with static Bertrand Nash Equilibrium. We observe different peri steady states, but they are all supracompetitive, which means there are always a price increase. As for the price trigger strategy, we always see that the peri steady state is the same with static monopoly equilibrium. Therefore, whatever the pre steady state is, there is always a price increase.

Rule-based	Pre s-s	Peri s-s	Price increase
Myopic	(p^2, p^2)	(p^8, p^5)	100%
Undercut	(p^2, p^2)	(p^{11}, p^{10})	100%
Price trigger	(p^{14}, p^{14}) or (p^2, p^2)	(p^{14}, p^{14})	100%
Ceiling	any (p, p) s.t. $p < p_0$	(p_0, p_0)	94%

Table 3: Steady states in peri period

4.2.2 Post period

To see the result in post period, we would compare it with peri period. Consider that firm 2 adopts algorithm at t_2 periods when firm 1 did at time 0. We try different t_2 such that $t_2 = \{1 \times 10^5, 2 \times 10^5, \dots, T\}$ such that T is the time when the algorithm of firm 1 converged. Any $t_2 \geq T$ should show a very similar result with $t_2 = T$.

To have a clearer view of result, we will use the price grid instead of the absolute price value.

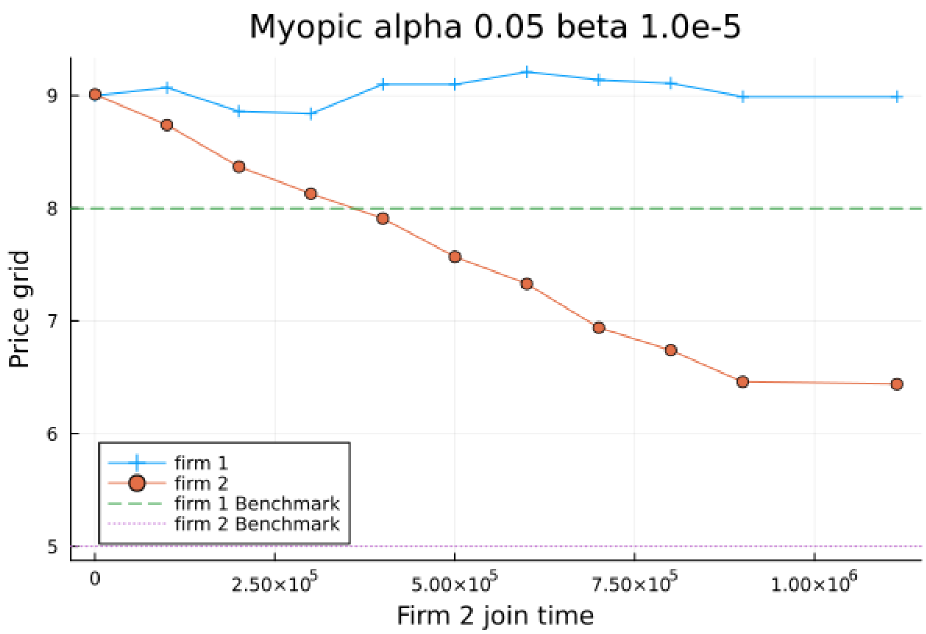

Figure 1: The converged prices when firm 2 adopts later

In Figure 1, the dashed lines are the benchmark or the steady state in peri period. The horizontal axis is the time that firm 2 adopts the algorithm, i.e., t_2. Each point in the figure represent the converged price for each firm given t_2. We can see that in this case, because myopic strategy is very competitive, the later firm 2 adopts, the lower price it would set. In any cases, the price is going to be higher than the benchmark. We would observe a very similar result when α and β are such that the algorithm learns faster in Figure 2.

For price undercut and price trigger, since the strategy is pretty cooperative, the prices doesn't change much when the t_2 increases. We also can't see a price increase compare to peri period. Detail figures are shown in Appendix 6.2.

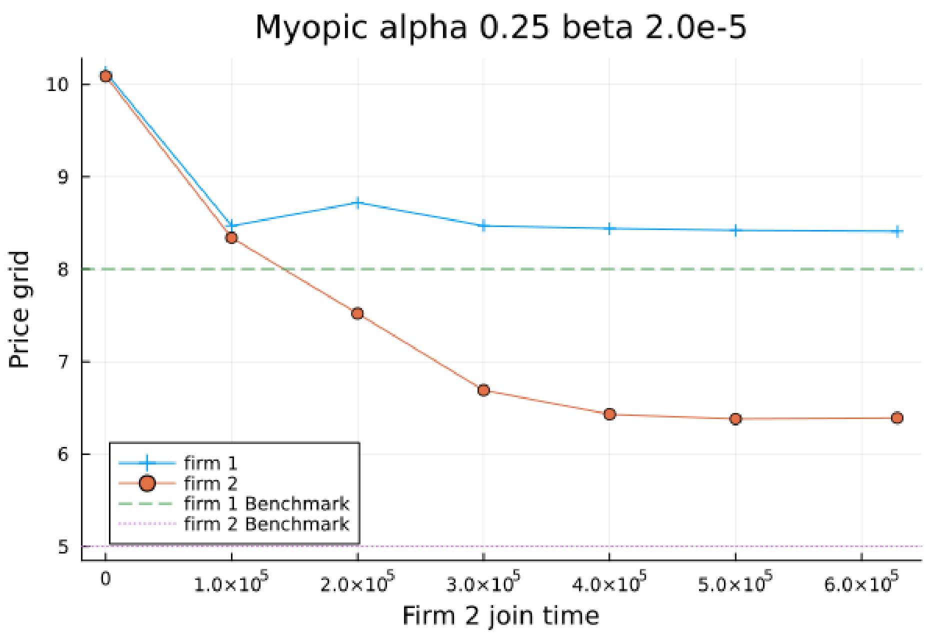

Figure 2: The converged prices when firm 2 adopts later

As for the profits, we first normalize the profit to be

$$\Delta = \frac{\bar{\pi} - \pi^N}{\pi^M - \pi^N}$$

The results follow the intuition that in the same market, the one with lower price would earn more. We would observe that in any cases, firm 1 will earn a lower profit compare to firm 2 in the equilibrium. However, both firms are earning higher payoff compare the static Bertrand equilibrium, which is the pre steady state.

Do firms benefit from the adoption of algorithm? It depends. In the myopic case, firm 1 always earns profit by adopting an algorithm, but firm 2 doesn't. We can see from Figure 3 that when firm 2 adopts the algorithm too early, it's actually not as good as not adopting algorithm at all (the benchmark). For firms, this is a prisoner's dilemma. If both firms don't adopt algorithms, they will earn a low profit that $\Delta = 0$. But if they want to earn better profit, it's better for them to wait until the rival adopts but they don't. If we consider the price of an algorithm, the profit of adopting could be negative.

Since the results follow Proposition 1, we can see that algorithm is always going to rise prices, not only when firms were in static Bertrand Nash equilibrium. But if both firms are using algorithms, then in some cases it's not as high as the Rule-Rule case.

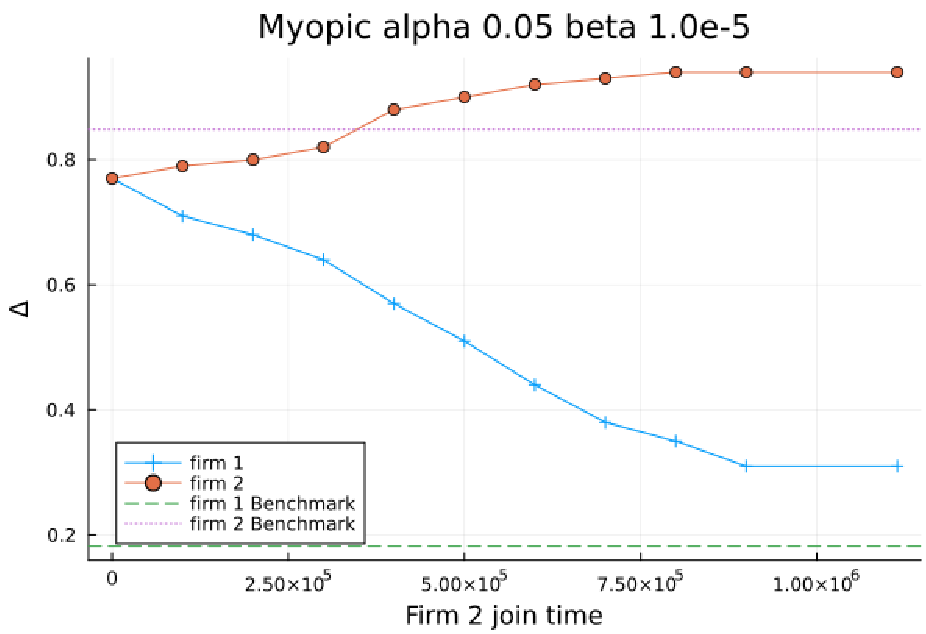

Figure 3: The converged profits when firm 2 adopts later

4.3 Deviation

To see if algorithms are in an equilibrium or not, we test by forcing a firm to deviate. Figure 4 shows how firms would response if firm 1 is forced to deviate at time 6. We would observe that firm 2 would punish firm 1 for this deviation. This is the reason why they could end up with supracompetitive price and an evidence that they are in an equilibrium. Figures of price undercut and price trigger is in Appendix 6.3.

5 Conclusion

This research sheds light on how the order of pricing algorithm adoption affects tacit collusion. We show by simulation experiments that algorithm does help firms to earn more profit, but the firm that adopts later would earn more.

We also show that algorithm could learn completely about the rival's strategy and response optimally in the case that the strategy does not change across time. Reinforcement learning would learn very well in the stationary Markov process where the profit of each action is constant. We also show by both theory and simulation that there must be a price

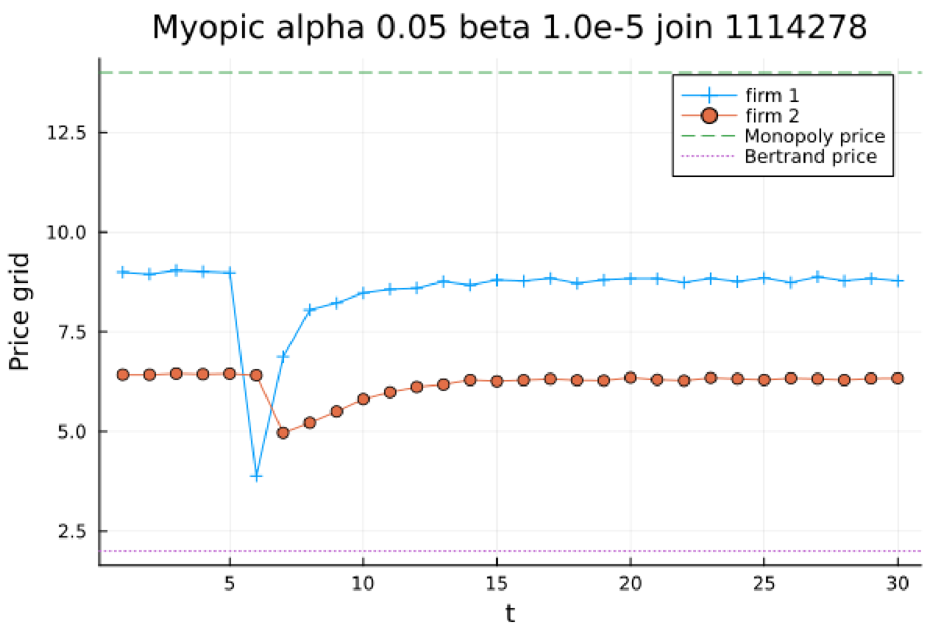

Figure 4: The price pattern when firm 1 deviates after convergence

increase in this case for any previous strategy that satisfy the assumptions. These assumptions are testable and also work for most of the cases in empirical work and theoretical work.

We could also see that not like no communication regulation works in traditional antitrust policy, no communication of the adoption of pricing algorithm is not going to help to keep the price low. Further discussion could be how to detect and punish firms that collude with each other using algorithm.

References

Asker, J., Fershtman, C., and Pakes, A. (2023). The impact of artificial intelligence design on pricing. *Journal of Economics & Management Strategy*.

Assad, S., Clark, R., Ershov, D., and Xu, L. (2024). Algorithmic pricing and competition: Empirical evidence from the german retail gasoline market. *Journal of Political Economy*, 0(0):000–000.

Brown, Z. Y. and MacKay, A. (2023). Competition in pricing algorithms. *American Economic Journal: Microeconomics*, 15(2):109–156.

Calvano, E., Calzolari, G., Denicolò, V., Harrington Jr, J. E., and Pastorello, S. (2020a). Protecting consumers from collusive prices due to ai. *Science*, 370(6520):1040–1042.

Calvano, E., Calzolari, G., Denicolo, V., and Pastorello, S. (2020b). Artificial intelligence, algorithmic pricing, and collusion. *American Economic Review*, 110(10):3267–3297.

Chen, L., Mislove, A., and Wilson, C. (2016). An empirical analysis of algorithmic pricing on amazon marketplace. In *Proceedings of the 25th international conference on World Wide Web*, pages 1339–1349.

Harrington Jr, J. E. (2023). An economic test for an unlawful agreement to adopt a third-party's pricing algorithm. *Available at SSRN 4520245*.

Johnson, J. P., Rhodes, A., and Wildenbeest, M. (2023). Platform design when sellers use pricing algorithms. *Econometrica*, 91(5):1841–1879.

Klein, T. (2021). Autonomous algorithmic collusion: Q-learning under sequential pricing. *The RAND Journal of Economics*, 52(3):538–558.

Wang, Q., Huang, Y., Singh, P. V., and Srinivasan, K. (2023). Algorithms, artificial intelligence and simple rule based pricing. *Available at SSRN 4144905*.

6 Appendix

6.1 Optimal Price of Logit demand

Firms are maximizing the per period profit

$$\pi_i = (p_i - c_i)q_i = (p_i - c_i) \cdot \frac{\exp(\frac{\gamma_i - p_i}{\mu})}{\exp(\frac{\gamma_i - p_i}{\mu}) + \exp(\frac{\gamma_j - p_j}{\mu}) + \exp(\frac{\gamma_0}{\mu})}.$$

The optimal price derived by the first order condition is

$$
\begin{aligned}
p_i^* = c_i + \frac{\mu}{1-q_i} &= c_i + \mu \frac{\exp(\frac{\gamma_i - p_i}{\mu}) + \exp(\frac{\gamma_j - p_j}{\mu}) + \exp(\frac{\gamma_0}{\mu})}{\exp(\frac{\gamma_j - p_j}{\mu}) + \exp(\frac{\gamma_0}{\mu})} \\
&= c_i + \mu \left(1 + \frac{\exp(\frac{\gamma_i - p_i}{\mu})}{\exp(\frac{\gamma_j - p_j}{\mu}) + \exp(\frac{\gamma_0}{\mu})} \right)
\end{aligned}
$$

We can then get

$$
\begin{aligned}
\frac{\gamma_i - p_i}{\mu} &= \frac{\gamma_i - c}{\mu} + 1 + \frac{\exp(\frac{\gamma_i - p_i}{\mu})}{\exp(\frac{\gamma_j - p_j}{\mu}) + \exp(\frac{\gamma_0}{\mu})} \\
\frac{\exp\left(\frac{\gamma_i - c_i}{\mu} - 1\right)}{\exp(\frac{\gamma_j - p_j}{\mu}) + \exp(\frac{\gamma_0}{\mu})} &= \exp\left(\frac{\exp\left(\frac{\gamma_i - p_i}{\mu}\right)}{\exp(\frac{\gamma_j - p_j}{\mu}) + \exp(\frac{\gamma_0}{\mu})} \right) \cdot \frac{\exp\left(\frac{\gamma_i - p_i}{\mu}\right)}{\exp(\frac{\gamma_j - p_j}{\mu}) + \exp(\frac{\gamma_0}{\mu})}
\end{aligned}
$$

The solution is given by

$$
\frac{\exp\left(\frac{\gamma_i - p_i}{\mu}\right)}{\exp\left(\frac{\gamma_j - p_j}{\mu}\right) + \exp\left(\frac{\gamma_0}{\mu}\right)} = W\left(\frac{\exp\left(\frac{\gamma_i - c_i}{\mu} - 1\right)}{\exp\left(\frac{\gamma_j - p_j}{\mu}\right) + \exp\left(\frac{\gamma_0}{\mu}\right)} \right)
$$

where W is the LambertW function. Using the logarithmic property of the LambertW function $ln(W(x)) = ln(x) - W(x)$, we get

$$
\frac{\gamma_i - p_i}{\mu} = \frac{\gamma_i - c_i}{\mu} - 1 - W\left(\frac{\exp\left(\frac{\gamma_i - c_i}{\mu} - 1\right)}{\exp\left(\frac{\gamma_j - p_j}{\mu}\right) + \exp\left(\frac{\gamma_0}{\mu}\right)} \right)
$$

$$
p_i^* = c_i + \mu + \mu W\left(\frac{\exp\left(\frac{\gamma_i - c_i}{\mu} - 1\right)}{\exp\left(\frac{\gamma_j - p_j}{\mu}\right) + \exp\left(\frac{\gamma_0}{\mu}\right)} \right)
$$

6.2 Simulation Results: Price Undercut and Price Trigger

6.3 Deviation: Price Undercut and Price Trigger

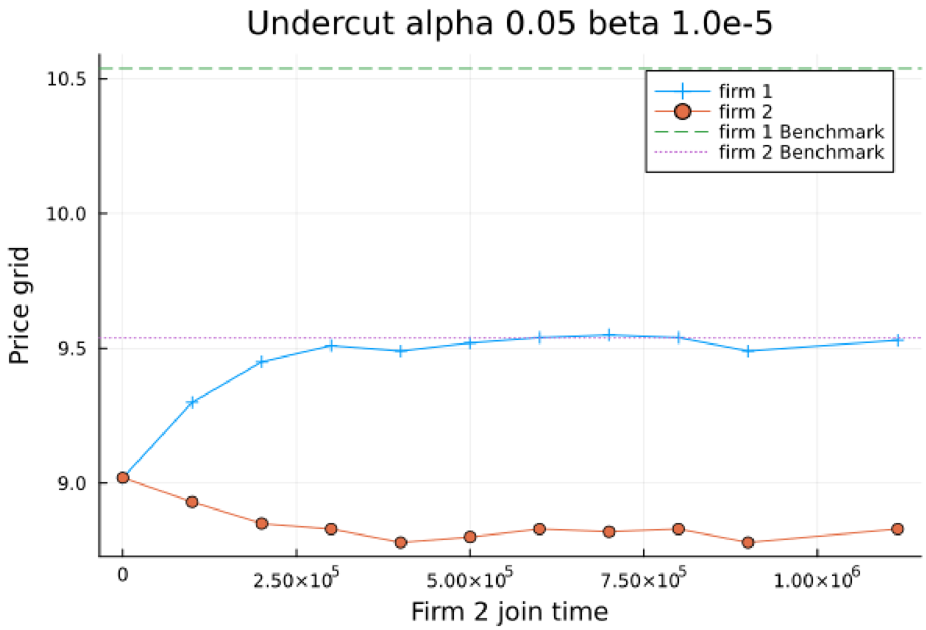

Figure 5: Price Undercut: Price

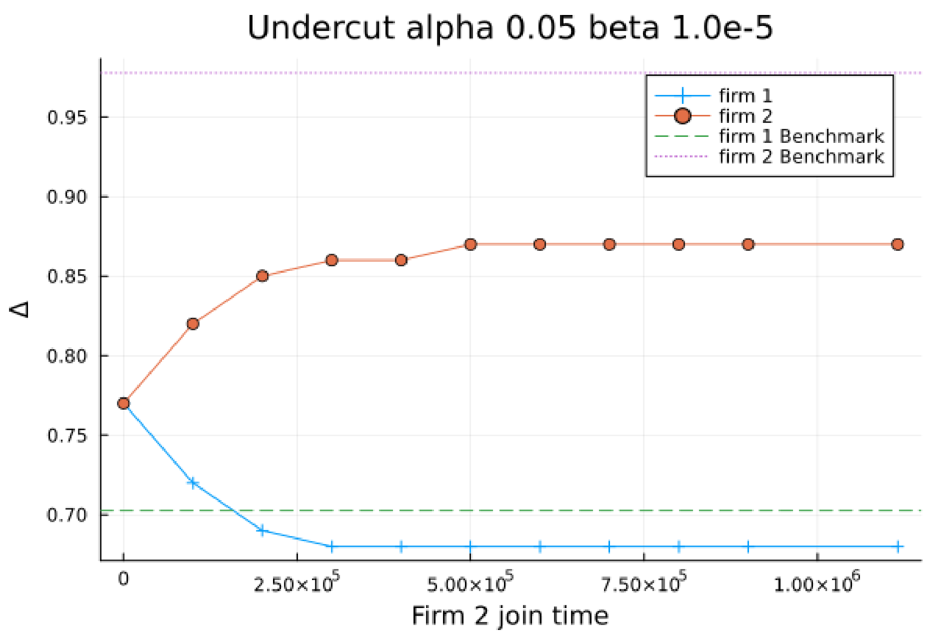

Figure 6: Price Undercut: Profit

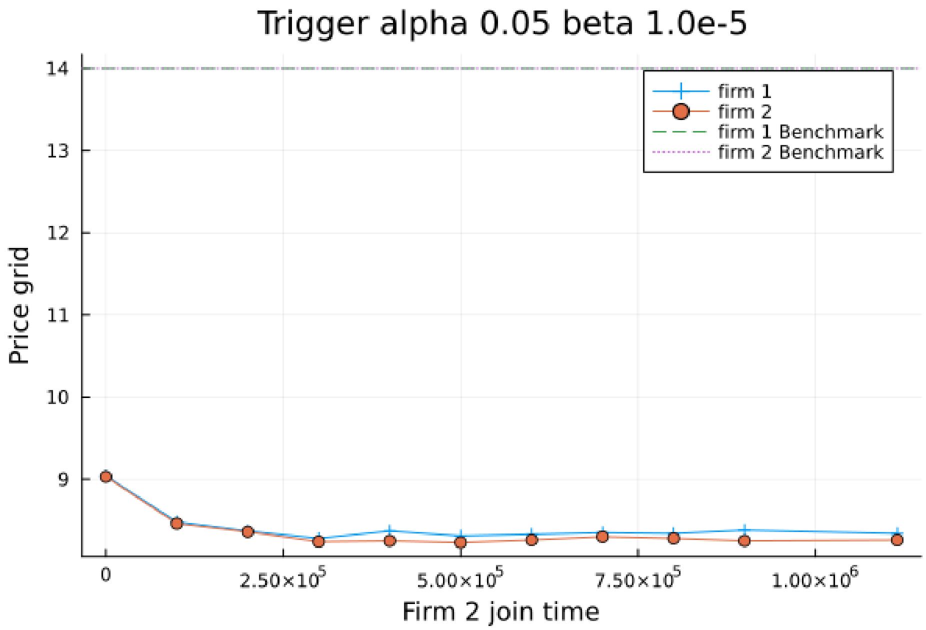

Figure 7: Price Trigger: Price

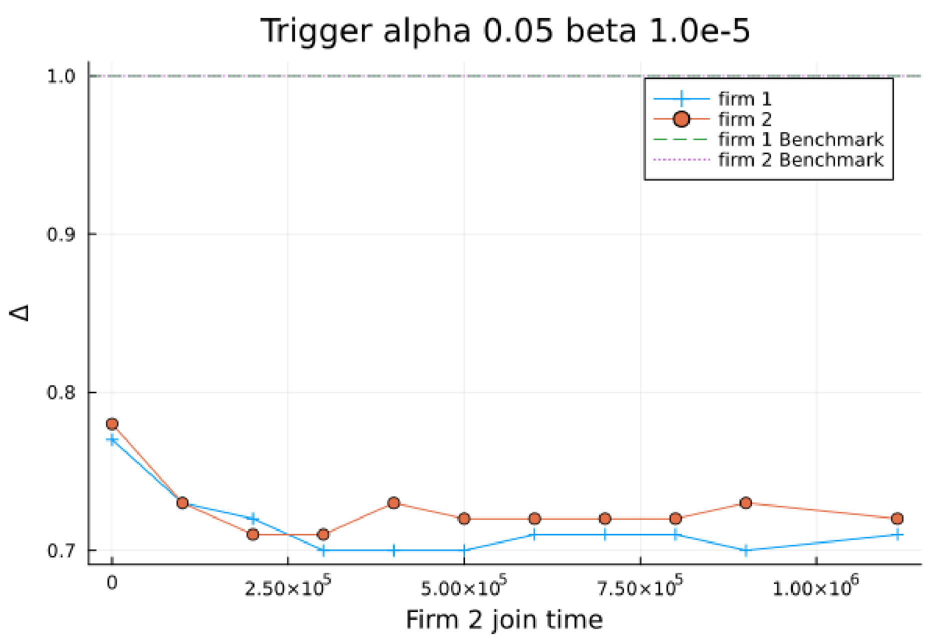

Figure 8: Price Trigger: Profit

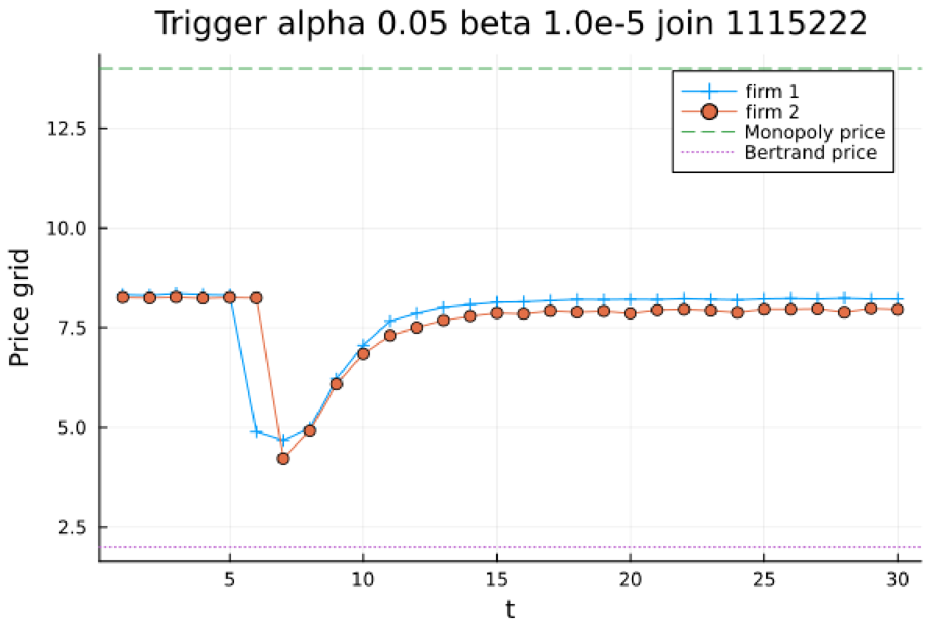

Figure 9: Price Trigger: Deviation

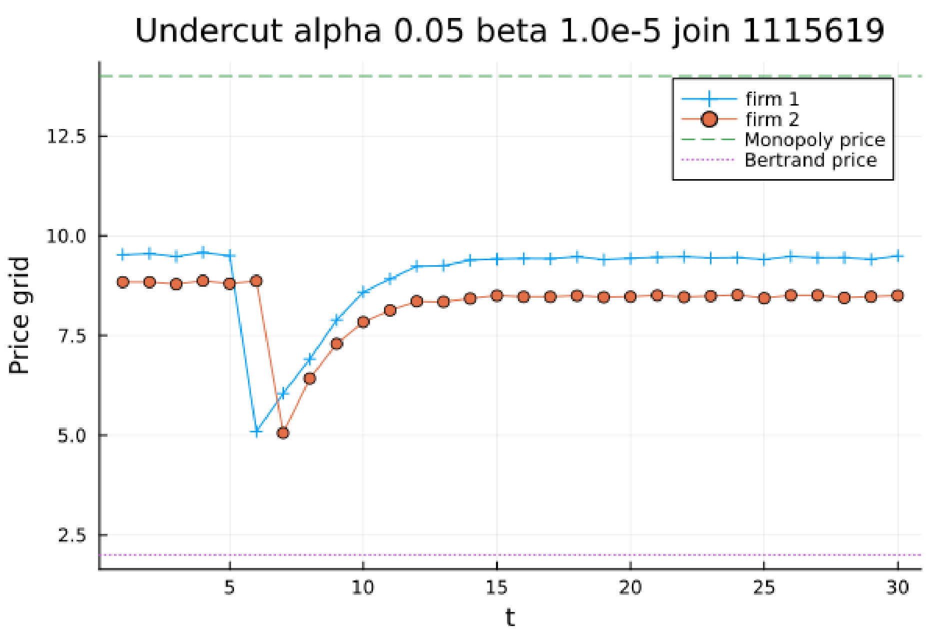

Figure 10: Price Trigger: Deviation